САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

СПИСОК-2014

МАТЕРИАЛЫ ВСЕРОССИЙСКОЙ НАУЧНОЙ КОНФЕРЕНЦИИ ПО ПРОБЛЕМАМ ИНФОРМАТИКИ

23–25 апреля 2014 г.,
Санкт-Петербург

Санкт-Петербург
ВВМ
2014

УДК 004(063)
С72

Печатается по рекомендации
кафедры системного программирования
Санкт-Петербургского государственного университета

С72 Список-2014: Материалы всероссийской научной конференции по проблемам информатики. 23–25 апр. 2014 г., Санкт-Петербург. — СПб.: ВВМ, 2014. — 592 с.

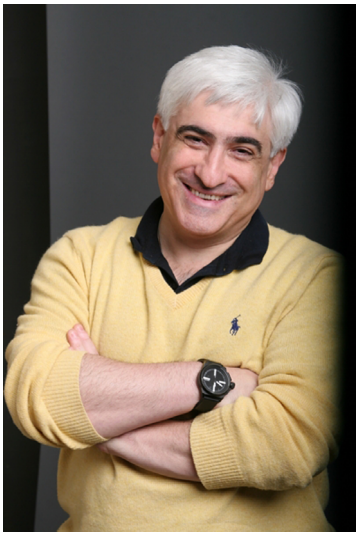
ISBN 978-5-9651-0872-5 ISSN 2310-4724

Тематика сборника затрагивает широкий круг актуальных проблем теоретической и прикладной математики и информатики. Слово «СПИСОК», ставшее названием конференции, — это не только обозначение фундаментальной структуры данных, но и сокращение от названий трех направлений исследований: Системное Программирование, Интеллектуальные Системы, Обеспечение Качества. Результаты исследований в этих областях являются значительной частью того знания, которое в настоящее время можно назвать словом «информатика».

Для студентов и аспирантов естественно-научных специальностей.

ISSN 2310-4724

ISBN 978-5-9651-0872-5



Научные исследования и разработки в сфере информатики сегодня оказывают одно из самых ощутимых и быстрых воздействий на окружающую нас жизнь, именно поэтому они являются одними из наиболее востребованных продуктов интеллектуального труда. Серьезным вызовом, разделяющим исследования и реальную жизнь по-прежнему остается проблема взаимодействия научного и делового сообществ. Без понимания обоюдных интересов больших успехов не достичь. Желаю участникам конференции идентифицировать точки приложения своих научных интересов в реальной экономике и найти правильных партнеров для реализации этих планов.

Александр Туркот
создатель IT-кластера
инновационного центра «Сколково»,

управляющий партнер и основатель
Maxfield Capital Partners

Нейроинформатика и мультиагентное управление

Тимофеев
Адиль Васильевич

д.т.н.
профессор
заведующий лабораторией информационных технологий
в управлении и робототехнике СПИИРАН

Памяти
Адиля Васильевича ТИМОФЕЕВА

Глубоко скорблю вместе с вами по поводу безвременной кончины А. В. Тимофеева. Пусть его имя, образ, работы и ученики навсегда останутся в нашей памяти.

В. О. Сафонов

Адиль Васильевич был моим научным руководителем начиная с руководства дипломной работой и кончая научным консультированием докторской диссертации. Он всегда предоставлял своим ученикам полную свободу выбора направления и методов научной работы, внимательно следя за общей тенденцией научных исследований. Нам будет его очень не хватать.

Т. М. Косовская

Умер родной, доброжелательный человек, с которым было всегда тепло и приятно общаться! Скорблю! Будет сильно его не хватать! Очень жалко!

А. А. Шалыто

Адиль Васильевич Тимофеев, закончив МВТУ имени Баумана, поступил в аспирантуру математико-механического факультета ЛГУ. Его научным руководителем был выдающийся ученый В. А. Якубович. Такой выбор Адиля Васильевича был вполне естественным. Владимир Андреевич Якубович, являясь сильным математиком и обладая широким научным кругозором, начал в то время исследования по распознаванию образов, адаптивным системам и робототехнике. Адиль Васильевич сразу включился в эти исследования, получив в этих направлениях весьма значительные результаты, которые были отражены в ряде солидных публикаций. После этого у Адиля Васильевича появились ученики, которые продолжали и развивали исследования профессора А. В. Тимофеева.

Следствием высокого научного авторитета Адиля Васильевича явилось приглашение работать на кафедре информатики СПбГУ, и долгое

время Адиль Васильевич совмещал работу заведующего лабораторией СПИИРАН с обязанностями профессора кафедры информатики СПбГУ.

Добрая память об Адиле Васильевиче Тимофееве — выдающемся ученом и замечательном человеке навсегда сохранится в наших сердцах.

Г. А. Леонов

Для меня — это большая и невосполнимая потеря. Я работал с этим выдающимся учёным и добрым и отзывчивым человек около 18 лет. Адиль Васильевич всегда отличался большим научным опытом и новыми интересными идеями. Кроме того, он был превосходным лектором и интересным собеседником. Широта его научных интересов всегда поражажала любого, с кем он общался в научных дискуссиях. Он был интеллектуалом-преподавателем для своих студентов, которых никогда не ругал и относился к ним с уважением и пониманием. Нам всем будет его не хватать. Особенно мне, мы с ним глубоко уважали друг друга и понимали друг друга с полуслова.

А. М. Бакурадзе

В середине семидесятых годов Лаборатория системного программирования ВЦ ЛГУ, в которой я тогда работал, размещалась на 14-й линии В. О., дом 29. Там я познакомился с Адилем Васильевичем Тимофеевым, который работал в Лаборатории теоретической кибернетики. Уж не знаю почему, но, несмотря на некоторую разницу в возрасте, мы сблизились, у нас были общие темы для разговора, мы часто пересекались за теннисным столом, на каких-то семинарах и конференциях. Потом наши пути разошлись. Честно говоря, я даже не знал, что он на некоторое время покинул мат-мех, но когда мы через много лет встретились в коридоре мат-меха уже в Петергофе, мы встретились как старые друзья. Оказалось, что у нас много научных пересечений. Сотрудники нашей кафедры выступали у него на семинарах в СПИИРАН. Особо хочу вспомнить нашу совместную работу в Ученом совете по защите диссертаций на мат-мехе. Сейчас многие порицают традицию совместных посиделок после успешной защиты аспирантов, но мне всегда они очень нравились. Адиль Васильевич выступал прекрасным рассказчиком, ему было, что вспомнить, и он умел это интересно рассказать. С грустью понимаю, что времена моих старших товарищей потихоньку уходят. К сожалению, Адиль Васильевич не первый человек из моего круга, который ушел из жизни. Но, видимо, так устроена жизнь. Я всегда буду его помнить улыбающимся.

А. Н. Терехов

КЛАССИФИКАТОР ДЛЯ СТАТИЧЕСКОГО ОБНАРУЖЕНИЯ КОМПЬЮТЕРНЫХ ВИРУСОВ, ОСНОВАННЫЙ НА МАШИННОМ ОБУЧЕНИИ[1]

А. В. Тимофеев

главный научный сотрудник Санкт-Петербургского института информатики и автоматизации Российской академии наук, Заведующий базовой кафедры нейроинформатики и робототехники Санкт-Петербургского государственного университета аэрокосмического приборостроения, Профессор кафедры информатики математико-механического факультета Санкт-Петербургского государственного университета, доктор технических наук, профессор, Заслуженный деятель науки РФ,

199178, Россия, Санкт-Петербург, 14-я линия, д. 39, СПИИРАН

E-mail: *tav@iias.spb.su*

Е. О. Путин

студент V курса математико-механического факультета Санкт-Петербургского государственного университета,

199178, Россия, Санкт-Петербург, 14-я линия, д. 29, математико-механический факультет СПбГУ

E-mail: putin.evgeny@gmail.com

Аннотация: СПИСОК (Системное Программирование, Информационные Системы, Обеспечение Качества) — периодическая научная конференция по проблемам информатики.

Данный документ представляет собой образец оформления тезисов конференции и содержит базовый набор стилей структурированного текста, рекомендованных к использованию.

Введение

В настоящее время использование глобальной сети Интернет является неотъемлемой частью нашей повседневной жизни. Через браузеры можно скачивать различный контент, в том числе программное обеспечение. Сегодня многие компьютерные системы становятся уязвимыми к «зловредным» программам, т. е. программам, нацеленным на нанесение вреда конечному пользователю, или компании. Зловредные программы могут быть классифицированы на несколько групп:

- **Вирусы** — компьютерные программы, которые размножают себя и внедряется в файлы пользователя или операционной системы;

[1] Работа выполнена при поддержке грантов РФФИ № 14-08-01276 и № 12-08-01167-а.

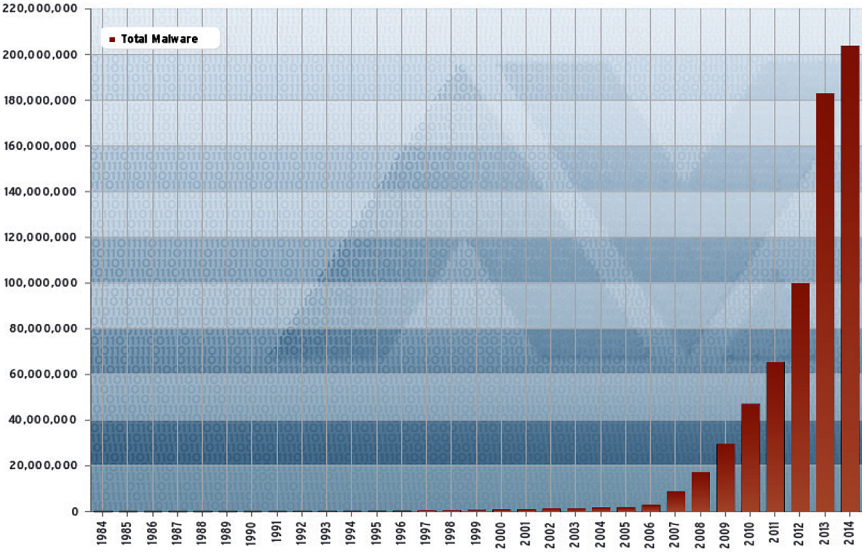


Рис. 1. Гистограмма роста количества вирусов во времени

- **Черви** — саморазмножающиеся компьютерные программы, которые способны посылать себя на другие компьютеры по локальной сети или Интернету;
- **Трояны** — программы, которые маскируют себя под желанную функциональность, но на самом деле реализуют другие «невидимые» операции, такие как неавторизированные доступ.
- **Шпионы** — программы, установленные в компьютере без осведомления пользователя для того, чтобы собирать о нем нужную информацию.

Каждая из указанных групп имеют свою уникальную специфику, но всех их объединяет операционная система Windows (в нашем случае 32-х битная),

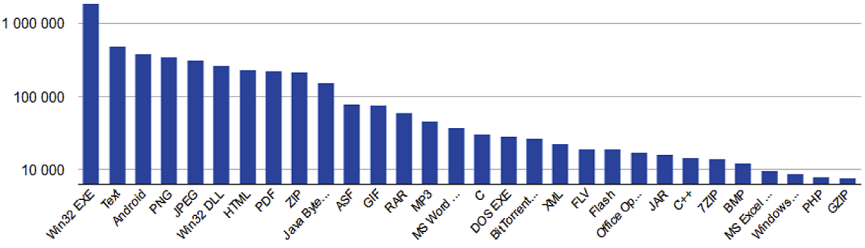


Рис. 2. Гистограмма проверяемости файлов на зараженность и инфицируемость (по оси Y — количество проверяемых файлов за последние 7 дней (30.03.14), по оси X — типы проверяемых файлов)

а значит и единый формат представления исполняемых файлов. Поэтому в дальнейшем под компьютерным вирусом будем просто понимать любую программу (win32), которая причиняет вред.

На Рис. 1 представлена гистограмма роста компьютерных вирусов [1].

Гистограмма проверяемости файлов на заражённость и инфицируемость представлена на Рис. 2 [2].

Как видно из рис. 2, самый проверяемый тип файлов Win32 Exe, что не удивительно из-за множества уязвимостей в Windows OS.

1. Подходы к обнаружению вирусов и актуализация

Методы обнаружения вирусов могут быть разделены на 2 класса :

1. Методы, основанные на сигнатурах.
2. Методы, основанные на выявлении аномалий.

В большинстве современных антивирусах центральное место занимает сигнатурный подход. Он даёт 100% точность обнаружения на уже известных вирусах. Но сигнатурный анализ бесполезен на тех вирусах, которые не известны антивирусу, т. е. не известны их сигнатуры. Будем называть такие вирусы, вирусами нулевого дня (zero-day viruses). Аномальный подход, наоборот, позволяет обнаруживать вирусы нулевого дня. Чаще всего аномальные методы строят классификаторы или решающие правила. Существует немало таких методов [3], но в последнее время наибольший потенциал представляют методы, использующие машинное обучение.

Сигнатурные методы обнаруживают вирус при помощи поиска сигнатур уже известных вирусов в специальном словаре — базе сигнатур вирусов.

Аномальные методы обнаруживают вирус, используя знания, правила и спецификации о нормальном поведении той или иной программы.

Преимущества и недостатки сигнатурного анализа:

1. Позволяет определять конкретный вирус с высокой точностью и малой долей ложных срабатываний.
2. Беззащитен перед полиморфными вирусами.
3. Требует регулярного и крайне оперативного обновления базы сигнатур.
4. На неизвестные вирусы требуются эксперты для ручного анализа вирусов и выделения сигнатур.
5. Неспособен выявить какие-либо новые вирусы, атаки.
6. На разные версии одного и того же вируса необходимы разные сигнатуры.
7. С учетом того, что база сигнатур огромна, сигнатурный анализ очень ресурсоемкая операция.

Преимущества и недостатки аномального анализа:

1. Возможность обнаружения ранее неизвестных вирусов (вирусов нулевого дня (zero-day viruses)).

2. Высокая вероятность ложных срабатываний, т. е. таких при которых доброкачественная программа была распознана как вирус.
3. Высокая сложность обучения системы.
4. «Лечение» неизвестного вируса практически всегда является невозможным.
5. Для уже обученной системы анализ выполняется сравнительно быстро (нужно лишь извлечь признаки).

Сигнатурные и аномальные методы внутри себя могут использовать три различных подхода для обнаружения вирусов:

1. Статический подход. Используя этот подход, подозрительная программа анализируется статически (т. е. без запуска самой программы) как обычный файл.
2. Динамический подход. При этом подходе, подозрительная программа анализируется динамически, т. е. во время ее выполнения в реальном времени.
3. Гибридный подход. Объединение статического и динамического подходов в разных частях анализа «зловредной» программы.

Ниже на рис. 3 приведена «карта ума» (mind map) по подходам к обнаружению вирусов.

В современном мире разрабатываются таргетированные вирусы и совершаются таргетированные атаки, т. е. такие атаки, которые нацеленные на конкретную компанию, организацию или страну.

К примеру, по опубликованным данным газеты New York Times [4] известный компьютерный червь Stuxnet, который по некоторым предположениям был специально разработан против компьютерной сети ядерного проекта Ирана, использовал 4 уязвимости нулевого дня и 3 известных. Более детальный обзор вируса Stuxnet и уязвимостей нулевого дня можно найти в последнем сообщении [5]. Также очень хорошо цифровая актуализация описана в отчете Лаборатории Касперского [6].

Из всего вышесказанного следует необходимость развития, усовершенствования и создания новых методов обнаружения вирусов в области аномального анализа, в том числе и в статическом подходе.

Рис. 3. «Карта ума» по методам обнаружения вирусов

Обнаружение вирусов при помощи методов машинного обучения (МО) представляет огромный потенциал в статистическом обнаружении вирусов, потому что все методы МО призваны обобщать решение конкретной задачи на общий класс аналогичных задач, и показывают практически лучшие результаты во всех задачах классификации и прогнозирования.

2. Общий подход к созданию классификатора

В начале исследований формируется и определяется выборка вирусов и чистых фалов.

Создание классификатора для обнаружения вирусов условно можно разделить на три стадии:

1 Формирование признакового описания файла (features extraction). Результатом этой стадии является вектор, содержащий признаковые характеристики рассматриваемого объекта. В задаче построения классификатора статического обнаружения вирусов признаками могут выступать следующие объекты:
   - Строки — исполняемый файл рассматривается как обычная строка или последовательность строк. Признаки — числовые характеристики строк (например, частота нулей в подстроке).
   - Структурные элементы Portable Executable файла. Эта специальная информация встроенная во все файлы Win32 и Win64. Она необходима для загрузчика операционной системы Windows и для самого приложения. Подробная документация доступна (см. например,[7]). Признаки, извлеченные из структурной информации PE файлов могут быть следующими: сертификат, date/time stamp, файловый указатель — позиция внутри файла информация компоновщика, тип CPU, логическая информация (выравнивание секций, размер, секции кода, отладочные флаги), информация об импорте — список тех DLL, которые использует исполняемый файл, и экспорте — те функции которые предоставляет другим приложениям, таблицу релокаций (перемещений) директории ресурсов — иконки, кнопки и прочее.
   - N-граммы на уровне байт. Сегменты последовательных байт из разных мест внутри исполняемого файла длины N. Каждая N-грамма рассматривается как признак. К примеру, количество байтовых биграмм — 256 * 256 = 65536 штук. Таким образом, будем рассматривать 65536 признаков.
   - N-граммы на уровне опкодов. Опкод (opcode — operation code) — специфичный для CPU операционный код, который выполняет специальную машинную команду (например, mov, push, add).
2. Выбор признаков (feature selection). В течение этой фазы вектор, созданный на стадии 1 вычисляется, а избыточные и нерелевантные признаки выбрасываются из рассмотрения. Выбор признаков имеет много преиму-

ществ: увеличение выполнения обучающейся модели за счет сокращения количества необходимых операций и, как следствие, увеличение скорости обучения, повышение обобщающей способности за счет сокращение размерности пространства признаков, удаление «выбросов», лучшая интерпретируемость и т. п. Задача этой стадии заключается в том, чтобы из уже имеющихся признаков выбрать наиболее значимые (информативные). Существует несколько подходов к выделению информативности признаков. Наиболее популярными являются корреляционные и фильтровые методы [4].

3. Построение математической модели, классификатора, который использует разряженный вектор, полученной стадии 2. Для построения классификатора могут использоваться следующие математические модели:
   - Деревья решений (decision trees, DT),
   - Случайный лес (random forest, RF),
   - Градиентный бустинг (gradient boosting machines, GBM),
   - Логистическая регрессия и ее оптимизации (logistic regression, LR),
   - Метод опорных векторов (support vector machines, SVM),
   - K-ближайших соседей (K-nearest neighbor, kNN),
   - Adaboost.
   - Наивный Байес (Naive Bayes, НБ, NB),
   - Нейронные сети (НС, NN).

3. Анализ исследований в области создания классификатора статического обнаружения вирусов

По результатам анализа проделанных ранее исследований можно сделать вывод, что не во всех работах присутствуют описанные выше фазы. Например, часто выбор признаков опускается. При этом авторы полагают, что большое количество признаков существенно не ухудшит результирующую модель, так как существуют устойчивые к «передозированным» признакам модели.

Следует сказать, что анализ и сравнение результатов исследований является сложной задачей, поскольку разные авторы используют различные обучающие множества. Кроме того, сравнение моделей происходит по различным метрикам.

В своей работе 2001 года [8] автор предложил следующий метод:

1. Для каждого исполняемого файла рассматриваются три различных признака:
   - Список DLL, использующихся внутри исполняемого файла.
   - Список системных вызовов DLL,
   - Количество различных системных вызовов внутри каждой DLL.
2. Выбор признаков, осуществлялся специальным индуктивным алгоритмом Ripper [9] для нахождения паттернов в данных DLL.

3. В качестве математической модели выступал Наивный Байесовский (NB) классификатор, который использовался для нахождения паттернов в строковых данных, а N-граммы последовательностей байт были использованы, как вход для Мультиномиального Наивного Байеса.

Обучающее множество состояло из 4266 файлов, из которых было 3265 вирусов и 1001 чистых файлов. Точность классификации для такого множества, подхода и модели была 97,11%.

В работе за 2006 год [10] использовался N-граммный по байтам подход. Их обучающее множество состояло из 1971 чистых файлов и 1651 зловредных. Выбор признаков проходил с использованием Information Gain. Были выбраны 500 наиболее часто встречаемых N-грамм. Ученые пробовали множество различных моделей (Naïve Bayes, SVM, KNN и др.), но лучшая из них — Adaboost дала 0.98 точности с 0,05% ложных срабатываний.

В работе [11] авторы использовался близкий к подходу, описанному в [8], а именно:

1. Из каждого исполняемого файла извлекались:
   - Структурные признаки формата PE: вся информация о заголовках PE и секциях,
   - Список всех используемых DLL,
   - Список всех функций внутри каждой DLL.
2. Все эти признаки были отфильтрованы по Information Gain (IG) и были выбраны 20 наиболее информативных признаков.
3. Рассматривалось несколько моделей (SVM, NB, DT), причём лучшей оказалась DT (C4.5 реализация) которая показала результат в 99,6% точности с 2,7% ложных срабатываний.

Исследователи использовали открытую вирусную коллекцию VX heaven [12] (которая на 2010 год насчитывала ~ 230 тысяч PE вирусов, червей и агентов) и собранную коллекцию чистых файлов в 10592. При этом много внимания уделялось выбору значимых признаков и построению моделей на различных подмножествах признаков, в том числе уменьшению размерности (пространства признаков), что дало чуть худший результат, чем ранее приведенный.

В фундаментальной работе [13] исследовался подход, основанный на опкодах. В качестве обучающего множества была взята VX heaven коллекция. В работе [13] дается детальный анализ использования и выделения опкодовых N-граммов. Перебором различных N лучший результат достигается на $N = 2$. При этом производилось построение множества моделей (RF, DT, NB, KNN, SVM) с оптимизациями, и сравнение моделей. Была выбрана лучшая модель, а именно SVM, с полиномиальным ядром для биграмм дала 95,90% точности с 0,03% ложных срабатываний.

В работе [14] использовался строковый подход. Для большей масштабируемости подхода использовались ансамбли SVM с бэгингом (Bagging,

Bootstrap Aggregating). Обучающее множество состояло из 39838 исполняемых файлов, из которых 8320 было чистых файлов и 31518 вирусов, червей, троянов и агентов фалов. Идея подхода состояла в том, чтобы извлекать из файлов интерпретируемые строки и использовать их в качестве прецедентов для обучения ансамбля SVM с последующей агрегацией. К примеру строка "<html> <script language = «javascript»> window.open(«readme.eml»)» всегда присутствует в червях «Nimda». Авторы работы [12] извлекли 13 448 строк из всего обучающего множества, а для селекции признаков воспользовались алгоритмом Max-Relevance, который отранжировал признаки, выбрав таким образом 3000 значимых признаков. Итоговой результат с примененной моделью — 92,22% точности.

4. Краткое изложение авторского исследования

За основу в проведённых исследованиях была взята VX Heaven коллекция вирусов, в которой насчитывается ~ 230 тысячи PE вирусов. Чистые файлы собирались с различных версий операционной системы Windows (95, XP, Vista, 7), свободного проекта Cygwin и другого свободного программного обеспечения взятого с сайта download.com. Выбирались исключительно файлы формата Poratable Executable, т. е. файлы с расширением exe, dll, sys. Всего было собрано 17746 чистых файлов.

Таким образом все обучающее множество представляло 248 тысяч исполняемых файлов.

Целью работы было:

- Исследовать существующие подходы статического анализа обнаружения вирусов с применением машинного обучения,
- Провести собственные независимые эксперименты,
- Разработать собственный классификатор, способный агрегировать различные признаковые описание одних и тех же файлов,
- Сравнить результаты приведенных исследований с собственными наработками.

Заметим, что у каждого из ранее описанных подходов есть свои принципиальные недостатки:

- У подхода с выделением строк недостаток в том, что на разные классы вирусов приходятся разные строковые описатели, а в целом необходимо формировать общее представление о всех рассматриваемых объектов. По этой причине он дает, худший результат, чем остальные подходы, но лучше может классифицировать конкретный тип зловредных программ.
- У подхода с N-граммами байт, хотя он дает хорошие результаты, очень слабая интерпретация с точки зрения исследователя. Например, трудно заключить, почему последовательность 0000 0000 0000 0001 более зна-

чима чем 0000 0000 0000 0010 и совершенно неочевидно, какой будет точность при увеличении исходного множества исполняемых файлов.

- У подхода с N-граммами опкодов хорошая интерпретация (за счет того что есть потенциально опасные последовательности опкодов которые известны), но не самый лучший результат по точности и дизассемблированию кода.
- У подхода со структурными признаками, хорошие результаты. Однако выделенную структурную информацию можно подделать. Если разработчик вируса пользуется стандартными средствами разработки (к примеру, Visual Studio), то структурная информация вирусного файла не сильно отличается от чистого, так как по умолчанию используются стандартные средства компиляции и линкера.

Из приведённого следует, что единственно верного решения нет, и имеет смысл рассматривать комбинации подходов. В предлагаемом подходе рассматривается комбинации N-грамм опкодов и структурных признаков PE файла.

Следует отметить, что весь статический анализ очень чувствителен к запакованным исполняемым файлам и применению обфускации на бинарном уровне. Запакованный исполняемый файл — это исполняемый файл внутри которого находится подпрограмма запаковывающая его и делающая недоступным правильное чтение его структурных элементов. Также бессмысленно дизассемблировать запакованный файл, потому что сначала будет дизассемблироваться сам пакер, а это не то, что требуется. Применение обфускации на бинарном уровне портит дизассемблирование файла, но не отражается на структурных элементах. Поэтому в исследовании рассматриваются только незапакованные исполняемые файлы. Для этих целей была специально разработана программа, которая и распаковывает все файлы.

Условно исследование можно разделить на 3 части:

1. Исследование структурного подхода и построение классификаторов для него.
2. Исследование N-грамм для опкодов и построение классификаторов для них.
3. Агрегация классификаторов с 1 и 2 стадии в один общий стековый классификатор.

Для структурного подхода рассматривались 173 признака — это вся информация о заголовках PE: Optional, File, Dos headers , отдельно поля DataDirectory и 8 основных секций: text(code), data, idata, edata, reloc, rsrc, debug, bss и tls. Поля секций, которых не было в исполняемом файле, обнулялись. Затем, выбрасывая из рассмотрения признаки с малым количеством уникальных значений (<10), строилось множество моделей более или менее устойчивых к передозировке признаками. Признаки, у которых был большой разброс в значениях, факторизовались, т. е. происходила перенумерация с присвоением. Лучшие результаты в 99,6% и 99,2% точности показали моде-

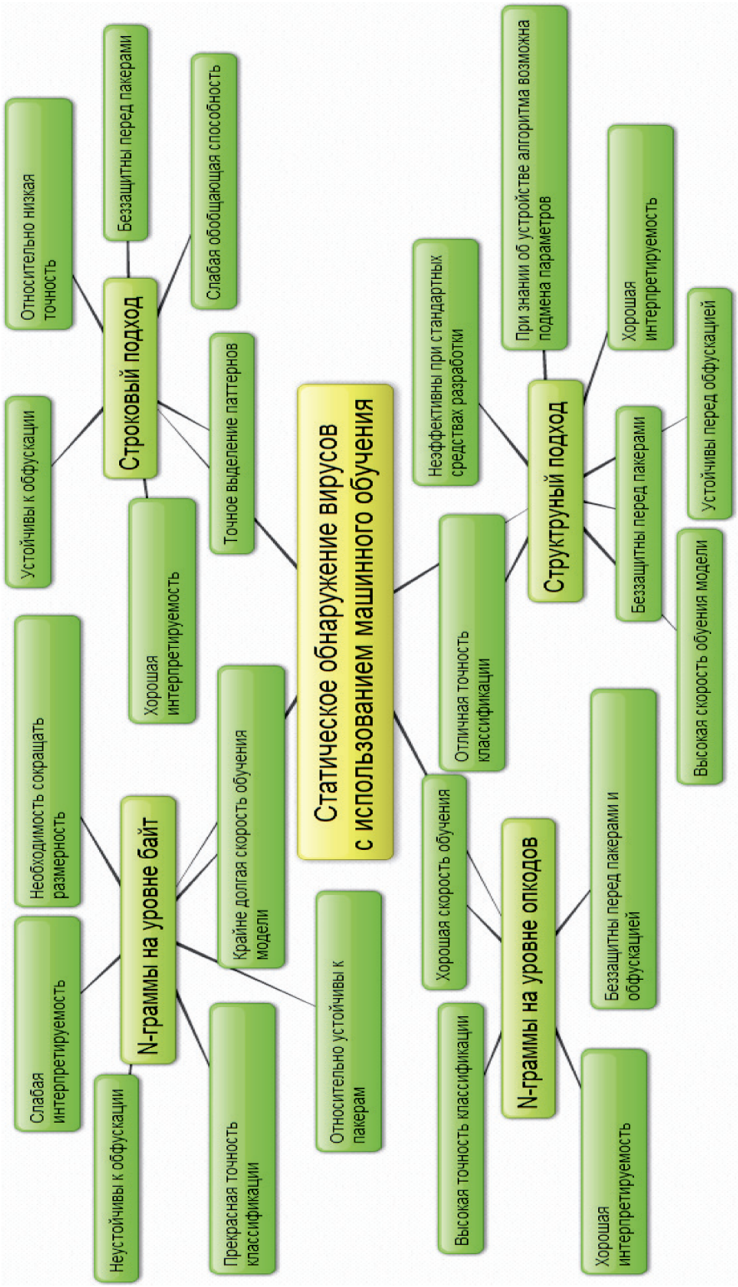


Рис. 4. «Карта ума» подходов машинного обучения для статического обнаружения вирусов

ли RF и GBM, соответственно. Также проводились исследования связанные с уменьшением размерности, в частности с использованием PCA анализа. После PCA осталось 17 признаков и обучение проводилось гораздо быстрее, но результат был хуже — 98,93% и 98,87% соответственно на тех же моделях.

Для N-граммного подхода на опкодах, все файлы дизассемблировались и брались 1-, 2-, 3-, 4-граммы (больше брать бессмысленно из-за затрат на скорость обучения и ухудшения общей точности) опкодов и для каждой граммы независимо строились модели.

На первой стадии фильтрации проводился корреляционный анализ, на выявление линейных зависимостей в данных. Они устранялись и далее удалялись признаки, которые имели меньше 20–30 уникальных значений. Это гарантировало отсев выбросов и нулевых или близких к ним незначащих признаков. Все признаки шкалировались на один интервал для ускорения работы градиентных методов. Для 1-грамм уменьшение размерности не требуется, а для 2-, 3- и 4-грамм проводился PCA анализ. Полученная точность в 97,38% для RF и 96,21 для GBM очень близка к исследованиям, проведённым в работе [13].

Также для каждого из подходов анализировалась значимость признаков для конкретной модели методом случайного перемешивания.

Суть метода заключается в следующем: обучаем какую-то модель, считаем общую эталонную точность модели на тренировочном множестве, далее в цикле по каждому признаку случайным образом перемешиваем каждый следующий признак, а другие оставляем в нормальном состоянии, смотрим на точность с перемешенным признаком и если эта точность мало отличается от эталонной, то признак для модели незначимый.

Таким образом, можно выделить признаки, главенствующие при обучении для конкретной модели, оставить их, а на всех остальных построить PCA и использовать как дополнительные признаки. Этот подход является оптимальным с точки зрения общей точности и затрат на обучение, но дает результат точности хуже, чем подход по всем признакам сразу (95,40% и 94,90% для RF и GBM, соответственно).

Настройка гиперпараметров каждой модели происходила при помощи кросс-валидации с 5 шагами.

Далее строился агрегированный классификатор. На вход ему подается матрица из предсказаний других классификаторов. Суммарно было построено 20 классификаторов (по 10 на каждый подход) и поэтому столбцов у матрицы предсказаний было 20.

Суть агрегированного классификатора заключается в том, чтобы стабилизировать и сгладить предсказания разных классификаторов натренированных на разных признаковых подмножествах описывающих одну и ту же задачу. Такой прием в машинном обучении называется Stacking или StackedGeneralization [15]. Он позволяет улучшить обобщающую способность классификатора, в некоторых случаях улучшая общую точность

до 5%, во многом благодаря сглаживанию и уменьшению количества ложных срабатываний. С учетом того, что данные с предсказаний классификаторов распределены близко к линейному, то чаще всего для стекового классификатора выбирается линейная модель, что и происходит в нашем случае.

В итоге стековый классификатор дал общую точность в 99,87% с 0,01% ложных срабатываний.

Заключение

Итоговый стековый классификатор, как и ожидалось, дал лучшую общую точность и меньшее число ложных срабатываний, за счет оптимального объединения построенных на предыдущих шагах классификаторов.

В дальнейшем планируется:

- Провести исследование с целью адаптации классификатора статического обнаружения вирусов с использованием машинного обучения на случаи запакованных, обфусцированных файлов. Такая цель может быть достигнута более глубокой предфильтрацией исходных файлов, к примеру можно выделить общие сегменты (байт, опкодов) присущие чистым файлам и соответственно вирусам и рассматривать в качестве признаков распределение этих общих сегментов внутри каждого файла.
- Провести исследование с большим количеством используемых моделей и автоматической селекцией признаков. К примеру, нейронные сети, а также глубокие нейронные сети представляют огромный потенциал в автоматической селекции признаков.
- Провести исследование с целью распознавания конкретных классов вирусов.

Литература

1. http://www.av-test.org/en/statistics/malware/
2. https://www.virustotal.com/en-gb/statistics/
3. *Nwokedi Idika, Aditya P. Mathur.* A Survey of Malware Detection Techniques. 2007.
4. http://www.nytimes.com/2011/01/16/world/middleeast/16stuxnet.html?pagewanted=all
5. http://go.eset.com/us/resources/white-papers/Stuxnet_Under_the_Microscope.pdf
6. http://media.kaspersky.com/documents/business/brfwn/ru/Advanced-persistent-threats-not-your-average-malware_Kaspersky-Endpoint-Control-white-paper-ru.pdf
7. http://msdn.microsoft.com/en-us/library/gg463119.aspx
8. *Schultz M. G., Eskin E., E. Z.,* and *Stolfo S. J.* Data mining methods for detection of new malicious executables // Proceedings of the IEEE Symp. on Security and Privacy. Pp. 38–49. 2001.
9. *Cohen W.* Fast effective rule induction // Proc. 12th International Conference on Machine Learning. Pp. 115–23. San Francisco, CA: Morgan Kaufmann Publishers, 1995.
10. *Kolter J. Z.* and *Maloof M. A.* Learning to detect and classify malicious executables in the wild // The Journal of Machine Learning Research. 7:2721–2744, 2006.

11. *Usukhbayar Baldangombo, Nyamjav Jambaljav,* and *Shi-Jinn Horng*. A Static Malware Detection System Using Data Mining Methods // International Journal of Artificial Intelligence & Applications. Jul. 2013. Vol. 4. Issue 4. P. 113.
12. vxheaven.org
13. *Igor Santos, Felix Brezo, Xabier Ugarte-Pedrero, Pablo G. Bringas*. Opcode Sequences as Representation of Executables for Data-mining-based Unknown Malware Detection // *Information Sciences: an International Journal*. Vol. 231. May, 2013. Pp. 64–82.
14. *Ye Y., Chen L., Wang D., Li T., Jiang Q.* and *Zhao M*. Sbmds: an interpretable string based malware detection system using svm ensemble with bagging // Journal in Computer Virology. 5 (4):283–293. 2009.
15. *Georgios Sigletos Georgios Paliouras Constantine D. Spyropoulos*. Combining Information Extraction Systems Using Voting and Stacked Generalization // Journal of Machine Learning Research. 6 (2005). Pp. 1751–1782.

ИНТЕЛЛЕКТУАЛЬНЫЕ РОБОТЫ-АССИСТЕНТЫ НА ЗЕМЛЕ И В КОСМОСЕ

С. Э. Чернакова
младший научный сотрудник СПИИРАН
E-mail: s_chernakova@rambler.ru

Нечаев А.И.
руководитель проекта НПП «Таир»
E-mail: a_nechaev@rambler.ru

А. А. Иванов
директор НПП «Таир»
E-mail: info@tair-npp.com

Аннотация. Исследоание посвящено разработке проекта «Интеллектуальный робот-ассистент».

Введение

Цель проекта — внедрение информационной технологии телеуправления интеллектуальными роботами-ассистентами на Земле и в Космосе.

Робот-ассистент предназначен для помощи космонавтам в отсеках МКС и других КА, в том числе: перспективных грузовых и сборочных КА, лунных орбитальных станциях, лунных и других напланетных станций.

Основные варианты информационной и технической помощи космонавтам:

— информационная поддержка (поиск документации, подсказки во время операций);
— вспомогательные операции (освещение, видеосъемка, целеуказание, сбор мусора);
— навигационная поддержка, поиск грузов и местоположения предметов в отсеках;
— помощь в обслуживании жизнедеятельности (гигиена, прием пищи, уборка помещений);
— круглосуточное наблюдение за ходом космических экспериментов;
— поддержка технологических операций (ремонта, замены, настройки оборудования);
— транспортировка грузов, контейнеров из грузовых отсеков КА и обратно;
— дистанционное управление оборудованием в нештатных ситуациях (задымление и пр.);
— автономное выполнение рутинных операций под контролем космонавтов или с Земли.

1. Аналоги и конкуренты

1.1. Аналоги роботов и систем управления:

— телеуправляемые андроидные роботы, многозвенные шагающие роботы, транспортные роботы и космические манипуляторы, работающие в копирующем, супервизорном и командном режимах;
— автономные программируемые и очувствленные роботы и мехатронные системы, группы роботов и мехатронных систем;
— самообучающиеся роботы и мехатронные системы, реконфигурируемые механические системы, ассоциации и распределенные системы микро- и нано-роботов;

1.2. Основные конкуренты на участие в наземных стендовых экспериментах, космических экспериментах, штатной эксплуатации на МКС, других орбитальных и напланетных станциях:

— «Robonaut-2» андроидный робот (находится на МКС демонстрация и реклама);
— «Киробо», японский андроидный робот-собеседник (используется на МКС);
— «SAR-400» андроидный робот (Россия) (проходит испытания в ЦПК);
— «DORES» универсальный технологический манипулятор ЦНИИРТК (опытные образцы);
— «Lemur-2» — шестиногий робот-ассистент (в разработке).

2. Основные решаемые проблемы

2.1. Надежность и устойчивость телеуправления при задержках, помехах, узкой полосе каналов связи.

2.2. Простота и естественность процессов обучения и контроля результатов обучения роботов, уверенность в адекватности действий автономного робота.

2.3. Надежность и достоверность функционирования роботов в автономном режиме в условиях частичной неопределенности задач и внешних условий.

2.4. Организация коллективной деятельности людей и роботов на основе естественного диалога, понимания задач и оценки ситуаций, планирования и реконфигурации ресурсов, оперативного взаимодействия экипажей и групп роботов.

3. Некоторые характеристики предложенной технологии

3.1. Многозвенные манипуляторы повышенной надежности, с избыточностью степеней подвижности и возможностью реконфигурации, наличие распределенной системы управления и силомоментного очувствления.

3.2. Единый многофункциональный интерфейс роботов для взаимодействия с внешней средой, информационными системами, людьми и другими роботами.

3.3. Интегрированная модель внешней среды, внутреннего состояния и поведения робота с возможностью обучения движениям, поведению и взаимодействию робота с людьми и другими роботами.

3.4. Информационная распределенная среда телеуправления роботами с возможностью накопления и систематизации знаний, формирования задач и прогнозов, организации взаимодействия распределенных групп людей и роботов.

4. Наши конкурентные преимущества

4.1. Модульность конструкции манипулятора, компактность парковки, большая рабочая зона, сменные полезные нагрузки и захватные устройства.

4.2. Возможность манипуляций с полезной нагрузкой в условиях земной гравитации (дополнительные устройства «разгрузки» не требуется).

4.3. Комплексное решение безопасности, навигации и человеко-машинного интерфейса в мультимодальной ассистивной информационной системе (многофункциональном интерфейсе).

4.4. Интеллектуальная система телеуправления на основе комплексных моделей внешней среды и внутреннего состояния робота, обучения показом движений и демонстрации поведения робота.

4.5. Информационная технология телеуправления, обеспечивающая надежность и достоверность поведения робота, в том числе при наличии задержек и сокращения полосы пропускания каналов связи.

5. Научно-технический задел участников проекта

5.1. Опытные образцы и действующие макеты роботов-манипуляторов, многозвенные мехатронные системы и функциональные модули, математическое обеспечение и конструкторская документация, результаты испытаний и опытной эксплуатации.

5.2. Разработки, образцы и результаты внедрения аппаратно-программных средств мультимодального интерфейса для диалога с оператором, информационных систем и систем телеуправления повышенной надежности и достоверности.

5.3. Теоретические методы и алгоритмы, аппаратно-программная реализация сило-моментного и визуального взаимодействия роботов с человеком и внешней средой, навигации и обучения методом показа движений.

5.4. Методы и алгоритмы реализации обобщенных образно-семантических структурированных моделей знаний, информационные системы формирования и передачи знаний и данных, методы построения информаци-

онной среды телеуправления распределенными группами роботов и групп операторов.

6. Ближайшие планы

6.1. Формирование предложений по концепции проекта как альтернативы известным технологиям и проектам телеуправляемых роботов-ассистентов.

6.2. Обсуждение вариантов развития проекта для решения актуальных социальных, технических и научных задач.

6.3. Создание рабочей группы и организация кооперации для реализации проекта.

Литература

1. *Kulakov F. M., Chernakova S. E*. Intelligent method of robots teaching by show // Robotic and Automation... (IDAACS+ River Publishing project).
2. *Kulakov F. M., Chernakova S. E*. Intelligent method of robots teaching by show // 7th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS-2013). 11–15 September 2013. Berlin. Germany.
3. *Kulakov F. M., Shmyrov A. S., Shymanchuk D. V*. Supervisory Remote Control of Space Robot in unstable Libration Point // 7th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS-2013). 11–15 September 2013. Berlin. Germany.
4. *Кулаков Ф. М., Чернакова С. Э*. Телеуправление космическими роботами с помощью тренажёра-интерфейса // Электронный сборник трудов 2-го российско-германского семинара по космической робототехнике. 25 ноября 2013 г. Звёздный городок. Россия.
5. *Кулаков Ф. М*. Тренажёр-интерфейс для дистанционного управления космическими роботами // Электронный сборник трудов 10-й международной научно-практической конференции «Пилотируемые полеты в космос». 27–28 ноября 2013 г. Звёздный городок. Россия.

ПРИНЦИПЫ МУЛЬТИФРАКТАЛЬНОГО ПРОЕКТИРОВАНИЯ ГЛОБАЛЬНЫХ СЕТЕЙ НОВОГО ПОКОЛЕНИЯ

А. М. Бакурадзе
Ведущий электроник СПИИРАН
E-mail: klauzert@yandex.ru

Аннотация. Описываются принципы многофрактального проектирования глобальных сетей нового поколения. Даётся описание фрактальной декомпозиции глобальных сетей на автономные подсети. Показано мультифрактальное представление глобальных сетей нового поколения.

Введение

Большая сложность, определяемая многомерностью и многосвязностью глобальных телекоммуникационных сетей (ТКС), делает задачу централизованного сетевого управления практически необозримой и трудно разрешимой. Однако на практике задача сетевого управления упрощается вследствие того, что глобальная ТКС большого масштаба естественным образом или, исходя из расчётных соображений распадается на автономные подсети меньшего масштаба (региональные, корпоративные и локальные ТКС) с типовой или смешанной топологией связей.

1. Фрактальная декомпозиция глобальных сетей на автономные подсети

Эффективным принципом разделения глобальной телекоммуникационной сети (ТКС) на автономные подсети является принцип **фрактальной** или **мультифрактальной декомпозиции** [1–3]. Использование этого принципа наиболее целесообразно на этапе концептуального проектирования глобальной ТКС и особенно её РТС и САУ.

Согласно этому принципу сложная глобальная ТКС разделяется на множество непересекающихся (т. е. не имеющих общих узлов) взаимосвязанных подсетей меньшего масштаба, архитектура которых топологически подобна архитектуре глобальной ТКС. В результате основные (базисные) компоненты глобальной ТКС — коммуникационная, информационная, управляющая и транспортная системы приобретают распределённый (по подсетям) характер и имеют меньшую сложность, определяемую масштабом подсетей.

При необходимости каждая из автономных подсетей в свою очередь может быть декомпозирована на непересекающиеся локальные (локализован-

ные) ТКС меньшего масштаба и сложности. Распределённая архитектура этих локальных (локализованных) подсетей останется топологически подобной архитектуре автономных и глобальных ТКС. Это значит, что топология подсетей в процессе декомпозиции глобальной ТКС относится к классу типовых сетевых топологий («звезда»,»кольцо», «полносвязная» и т. п.). В частности, она может совпадать с топологией глобальной ТКС.

В этом проявляется **фрактальность** (самоподобие) локальных подсетей и глобальной ТКС. Принцип самоподобия (self-similarity) тесно связан с быстро развивающейся в последние годы теорией фракталов.

Термин «фрактальный» происходит от латинского слова «fractus» и означает «дробный» или «ломанный». В современной математике фракталами называются геометрические или графические объекты (кривая Пеано, снежинка Коха, ковёр Серпинского, множество Мандельброта и т. п.), обладающие свойствами определённого самоподобия.

Самоподобие как основная характеристика фрактала означает, что он устроен более или менее единообразно в широком диапазоне изменения масштабов или размера.

Теоретическая значимость и всеобъемлющий характер свойства самоподобия была отмечена М. Шредером: «Самоподобие представляет собой понятие, объединяющее фракталы, хаос и степенные законы. Самоподобие или инвариантность относительно изменений масштаба или размера является отличительной чертой многих законов природы и бесчисленных явлений в окружающем нас мире. Самоподобие в действительности представляет собой одну из решающих симметрий, формирующих нашу Вселенную и влияющих на наши попытки понять её.»

Структура или процесс, обладающие свойством самоподобия, имеет одинаковый вид или одинаково ведёт себя при их рассмотрении с разной степенью «увеличения» или в разном масштабе. В роли масштабирующего параметра может выступать геометрическая величина (длина, ширина, высота) или время.

Применительно к глобальным ТКС свойство самоподобия может проявляться в их топологической структуре, в трафике данных или в ошибках, возникающих в каналах связи. В последние годы появилось много исследований и результатов, свидетельствующих о самоподобной (а не пуассоновской) природе трафика данных в глобальных ТКС.

Интерпретация глобальной ТКС как фрактала основана на том, что ей присуща определённая топологическая симметрия, т. е. некоторая неизменность топологии подсетей при изменении масштаба в процессе декомпозиции. При этом максимальный масштаб (количество узлов N) глобальной ТКС ограничен сверху некоторой величиной N^*. С другой стороны, минимальный масштаб локальных подсетей не может быть меньше некоторого числа N_*, при котором свойство самоподобия теряется. Аналогичные ограничения, задаваемые числами M_* и M^*, имеют место для числа каналов связи M.

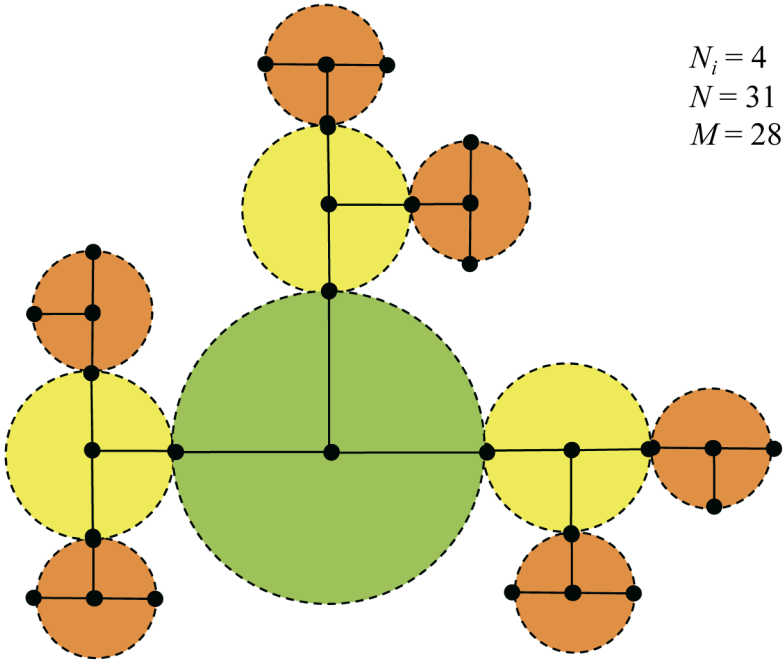


Рис. 1. Фрактальное представление однородной глобальной ТКС с топологией «звезда»

Свойство полного самоподобия характерно лишь для так называемых регулярных фракталов, характеризующихся только одним параметром — фрактальной размерностью. К таким фракталам можно отнести только идеальные гомогенные ТКС с однородной (одинаковой) топологией подсетей.

На Рис. 1 представлена топология однородной (регулярной) глобальной ТКС типа «звезда» при её фрактальной декомпозиции на одну магистральную, три автономные и шесть локальных подсетей с числом узлов 4 и диаметром, кратным 2. Общее число узлов этой ТКС N = 31, а общее число каналов связи M = 28.

Другой пример фрактальной декомпозиции однородных ТКС, состоящих из иерархически связанных подсетей с топологией связей типа «кольцо», изображён на Рис. 2. Число узлов (масштаб) каждой подсети r-го уровня иерархии $N_r = 4r$, где $r \geq 1$. Каждая такая подсеть имеет собственный маршрутизатор. Эти маршрутизаторы подсетей связаны между собой и с центральным маршрутизатором ТКС хотя бы одним каналом связи.

В тех случаях, когда подсети k-го уровня однородной глобальной ТКС имеют топологию типа «кольцо», но их реальный масштаб не превышает

расчётный (например, реальный масштаб k-ой подсети меньше 4r), можно ограничиться исследованием только реальных узлов, считая остальные узлы фиктивными (виртуальными). Тогда каналы связи соединяют только соседние реальные узлы ТКС.

Задачи сетевого управления такими регулярными или квазирегулярными ТКС легко декомпозируются и вследствие этого значительно упрощаются.

Реальные ТКС большого масштаба, как правило, не бывают однородными. Поэтому их следует отнести к **мультифрактальным**, т. е. неоднородным (гетерогенным) фрактальным объектам. Это связано с тем, что при фрактальной декомпозиции глобальной ТКС на подсети меньшего масштаба их топология может быть самоподобной только в заданном классе типовых сетевых топологий, т. е. с точностью до принадлежности к этому классу. Хотя класс типовых сетевых топологий на практике достаточно ограничен (обычно число типовых топологий не превышает 5), в процессе фрактальной декомпозиции глобальной ТКС возникает неопределённость (энтропия) и связанная с ней неоднозначность.

Пример мультифрактального представления гетерогенной глобальной ТКС при её декомпозиции на одну магистральную, четыре автономные и семь локальных подсетей представлен на Рис. 3.

На рис. 3 изображена декомпозиция гетерогенной глобальной ТКС со смешанной топологией связей и числом узлов в подсетях, равном 4. Эта

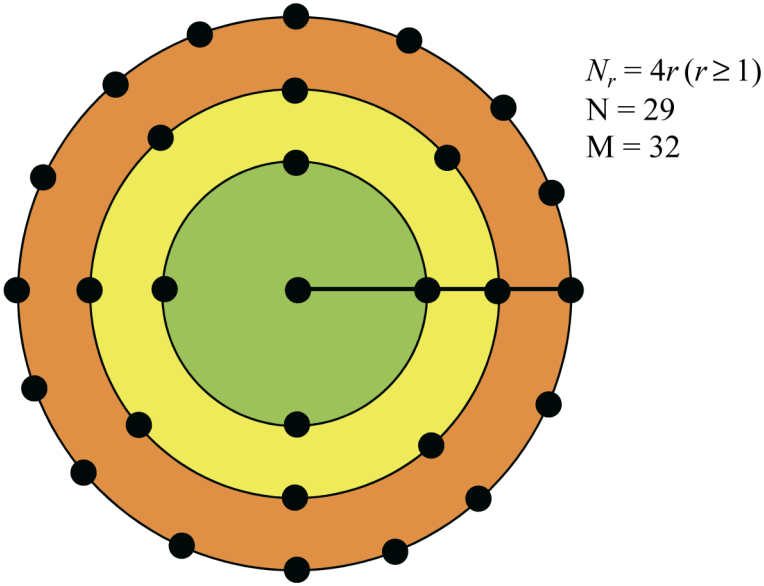


Рис. 2. Фрактальная декомпозиция однородной ТКС с иерархической топологией типа «кольцо»

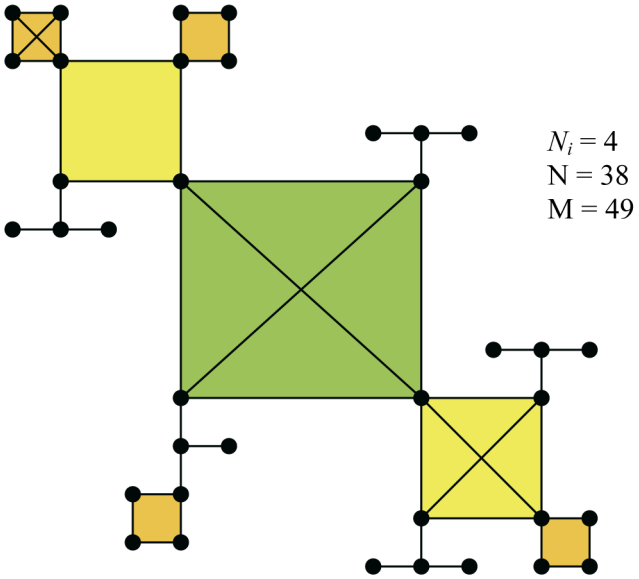


Рис. 3. Мультифрактальное представление глобальной ТКС со смешанной топологией связей

ТКС состоит из одной магистральной, одной автономной и одной локальной подсети с топологией «полносвязная», двух автономных и трёх локальных подсетей с топологией «звезда» и одной автономной и трёх локальных подсетей с топологией «кольцо». Общее число узлов такой гетерогенной ТКС N = 38, а общее число каналов связи M= 49.

Задача синтеза топологической структуры (облика) глобальной ТКС и её подсетей является одной из центральных проблем концептуального проектирования ТКС нового поколения. Она заключается в рациональном (в частности, оптимальном) выборе числа узлов и топологии связей между ними для всех подсетей и объединяющей их глобальной ТКС в целом.

Важную (а, может быть, центральную) роль при решении этой задачи играет принцип фрактальности или мультифрактальности глобальных ТКС новых поколений. Согласно этому принципу такие ТКС должны, по возможности, состоять из множества взаимосвязанных подсетей меньшего масштаба и сложности, которые являются топологически самоподобными.

Главное достоинство фрактального или мультифрактального подхода к проектированию глобальных ТКС новых поколений заключается в возможности **стандартизации** и **унификации** методов и средств сетевого управления, приёма, обработки и передачи информационных потоков в самоподобных подсетях разного масштаба. Благодаря этому значительно упростятся

и унифицируются методы проектирования основных (базовых) компонентов ТКС, а именно: сетевой системы управления и распределённых транспортных, коммуникационных и информационных систем.

Литература

1. *Тимофеев А. В.* Мульти-агентное управление и интеллектуальный анализ потоков данных в компьютерных сетях. СПб.: Анатолия, 2012. 282 с.
2. *Тимофеев А. В., Остюченко И. В.* Мульти-агентное управление качеством в телекоммуникационных сетях // Труды 11-й Всероссийской научно-методической конференции «Телематика-2004» (Санкт-Петербург, 7–10 июня 2004 г.). Т. 1. С. 177–179.
3. *Сырцев А. В., Тимофеев А. В.* Модели и методы маршрутизации потоков данных в телекомму-никационных системах с изменяющейся динамикой. М.: Новые технологии, 2005. 82 с.

ПРЕДВАРИТЕЛЬНАЯ ОБРАБОТКА ВИДЕО ДАННЫХ ДЛЯ РАСПОЗНАВАНИЯ В СИСТЕМАХ УПРАВЛЕНИЯ ДВИЖЕНИЕМ

А. А. Алексеев
докторант СПИИРАН
E-mail: alexeev3@yahoo.com

Аннотация. Описываются принципы обработки видео данных для распознавания в системах управления движением

Введение

Для реализации алгоритмов распознавания требуется предварительная обработка изображений видео потока с помощью Левенберга — Марквардта.

1. Основные идеи

Предварительная обработка включает в себя:

- **Устранение искажений на изображениях с камер, вносимых не идеальностью оптического тракта.**

Предполагается, что для каждой камеры известны ее внутренние параметры и коэффициенты искажений, а также то, что изображения с камер откалиброваны, искажения устранены, а ректификация изображений не требуется.

- **Построение матрицы расстояний до каждой точки наблюдаемого изображения.**

Анализ механизмов распознавания человеком показал, что трехмерная информация о пространстве позволяет более обоснованно проводить последующее распознавание отдельных объектов, формировать управляющую траекторию движения самоходной установки. Распознавание объектов на плоскости не использует информации об объемном характере видео сцен, поэтому является существенно менее эффективным. Для задач построения расстояний до каждой точки наблюдаемого изображения используются методы широко известной технологии фотограмметрии. Фотограмметрия позволяет построить карту расстояний до снимаемого объекта при совмещении одновременно полученных снимков с двух отстоящих друг от друга на известное расстояние видео камер. Пространственные координаты точек объекта определяются путём измерений, выполняемых по двум или более фотографиям, снятым из разных положений. При этом на каждом изображении отыскиваются общие точки. Затем луч зрения проводится от местоположения фотоаппарата до точки на объекте. Пересечение этих лучей и определяет рас-

положение точки в пространстве. Алгоритмы, фотограмметрии, имеют целью минимизировать сумму квадратов множества ошибок, решаемую с помощью алгоритма Левенберга — Марквардта (или метода связок), основанного на решении нелинейных уравнений методом наименьших квадратов [1]. При использовании технологии фотограмметрии обеспечивается высокая точность измерений, а стоимость аппаратуры измерений имеет невысокую стоимость по сравнению с методами радарграмметрии и лазерного сканирования.

- **Разделение трехмерных объектов и выделение их контуров или границ.**

Осуществляется выделение объектов и изображений на плоскостях, выделение контуров, границ и текстуры соответствующими методами сегментирования изображения. Среди них можно выделить методы: основанные на кластеризации, выделении краёв, разрастания областей, разреза графа, сегментирования с помощью модели, многомасштабной сегментации [2].

Литература

1. *Лобанов А. Н.* Фотограмметрия. М.: Недра, 1984. 553 с.
2. *Форсайт Дэвид А., Понс Жан*. Компьютерное зрение. Современный подход / Пер. с англ. М.: Издательский дом “Вильямс”, 2004. 928 с.

СРАВНИТЕЛЬНЫЙ АНАЛИЗ МЕТОДОВ КОЛЛЕКТИВНОГО ИНТЕЛЛЕКТА В ЗАДАЧАХ КЛАСТЕРИЗАЦИИ И ОПТИМИЗАЦИИ

А. И. Кукушкин

аспирант кафедры компьютерных систем и программных технологий

E-mail: polladin@rambler.ru

Е. Н. Бендерская

к.т.н., доц. кафедры компьютерных систем и программных технологий

E-mail: helen.bend@gmail.com

Санкт-Петербургский государственных политехнический университет

Аннотация: В данной работе проведён обзор коллективных алгоритмов применительно к задачам кластеризации и оптимизации. Рассматривается применение коллективных алгоритмов к комбинаторным задачам, оптимизации непрерывных функций и кластеризации. Проведена модификация одного из представленных алгоритмов кластеризации.

Ключевые слова: коллективный интеллект, кластеризация, оптимизация, самоорганизация, параллельные вычисления, вероятностные алгоритмы

Введение

Коллективный интеллект описывает коллективное поведение децентрализованной самоорганизующейся системы. Системы коллективного интеллекта, как правило, состоят из множества агентов локально взаимодействующих между собой и с окружающей средой. Сами агенты обычно довольно просты, но все вместе, локально взаимодействуя, могут решать сложные задачи. Примером в природе может служить колония муравьев, рой пчел, стая птиц и т. д. Глобальная динамика коллективного интеллекта позволяет выполнять задачи, которые никак не смог бы выполнить один отдельный представитель коллектива.

Существует большое количество алгоритмов основанных на поведении различных биологических видов, но далеко не все эти алгоритмы пригодны или актуальны для решения прикладных задач. В данном обзоре приведены те алгоритмы, которые можно использовать для задач оптимизации и кластеризации.

Методы коллективного интеллекта относятся к вероятностным алгоритмам. Скорость выполнения таких алгоритмов выше, чем детерминированных, но решение не всегда будет оптимальным. Коллективные алгоритмы

делят задачу на несколько параллельно выполняемых потоков, в связи с быстрым развитием аппаратной базы для параллельных вычислений эта область становится всё более актуальной.

В данной работе будут рассмотрены алгоритмы на основе колоний муравьёв. Они не содержит централизованного управления и каждый рабочий имеет доступ к очень ограниченной информации. Метод взаимодействия между муравьями основан на оставлении тропы из феромонов, так муравьи могут находить оптимальный путь от еды до гнезда. В процессе передвижения, муравьи оставляют фермоны на земле, и идут по вероятностно выбранному пути, который выбирается на основе феромон оставленных предыдущими муравьями [3].

Описанное выше поведение муравьёв вдохновило на создание алгоритма, в котором искусственные муравьи для решения задачи взаимодействуют друг с другом через оставления феромон на ребрах графа. Такая муравьиная система может быть применена для решения комбинаторных оптимизационных задач, таких как задача коммивояжёра.

Решение дискретной задачи оптимизации

Применение метода муравьиных колоний для решения задачи коммивояжёра (Transport Salesman Problem (TSP))

TSP является NP-полной, т. е. сложность алгоритма поиска оптимального пути не полиномиальная, и для того, чтобы гарантированно найти оптимальный путь, в общем случае надо будет перебрать все возможные варианты. Для некоторых задач критичным является время поиска пути, и менее критичным его оптимальность. В этом случае можно использовать алгоритм на основе муравьиных колоний [1, 2].

Описание алгоритма:

1) Необходимо разместить N муравьёв в K вершинах.
2) На каждой итерации, муравьи, независимо друг от друга пытаются пройти все вершины, при этом правило, по которому каждый муравей выбирает следующий город, выглядит следующим образом:

$$p_k(r, s) = \begin{cases} \dfrac{[\tau(r, s)][\eta(r, s)]^{\beta}}{\sum_{u \in J_k(r)} [\tau(r, u)][\eta(r, u)]^{\beta}} & \text{if} \quad s \in J_k(r) \\ 0 & \text{otherwise} \end{cases},$$

где τ – это феромоны, а η – величина обратно пропорциональная расстоянию между городами, J_k – набор достижимых городов из пункта r. β — параметр, насколько важнее феромоны, по отношении к расстоянию межу городами.

1) После того, как все муравьи прошли свой путь, обновляем ферономы. Различные алгоритмы отличаются процессом обновления феромон, некоторые из возможных вариантов описаны ниже.
2) Запускаем алгоритм определённое количество итерации, или до тех пор, пока все муравьи не будут ходить по одному пути.

   Некоторые из возможных вариантов обновления феромон:

   - Помечается путь всех муравьёв, в зависимости от длинных пройденного пути:

$$\tau(r, s) \leftarrow (1-\alpha)\,\tau(r, s) + \sum_{k=1}^{m} \Delta\tau_k(r, s),$$

где

$$\Delta\tau_k(r, s) = \begin{cases} 1/L_k & \text{if } (r, s) \in \text{tour done by ant } k \\ 0 & \text{otherwise} \end{cases},$$

где α – от 0 до 1 параметр испарения феромонов, L_k – длина пути.

   - Учитывать только один путь — минимальный:

$$\tau(r, s) \leftarrow (1-\alpha)\,\tau(r, s) + \alpha\Delta\tau(r, s),$$

где

$$\Delta\tau_k(r, s) = \begin{cases} 1/L_{gb} & \text{if } (r, s) \in \text{globalbest tour} \\ 0 & \text{otherwise} \end{cases}.$$

   - Также возможен вариант локального обновления феромон, когда после каждого шага муравья обновляются феромоны (такой вариант плохо подходит при параллельном выполнении алгоритма):

$$\tau(r, s) \leftarrow (1-\rho)\,\tau(r, s) + \rho\Delta\tau(r, s).$$

Решение задачи коммивояжёра для полносвязного графа, т. е. каждый город соединён с каждым:

Для полносвязного графа, чтобы найти гарантированно оптимальное решение, необходимо рассмотреть все возможные варианты, количество которых равно $(n-1)!$ для ассиметричной TSP, для симметричной комбинаций будет в 2 раза меньше.

К примеру, для 16 городов необходимо перебрать 1 307 674 368 000 различных вариантов, такое количество перебора, даже у современных компьютеров, займёт значительное время.

Матрица смежности для такой задачи представляет собой матрицу эвклидовой дистанции размерностью 16×16. По горизонтали и по вертикали расстояние между городами одинаково.

На рисунке показан найденный путь:

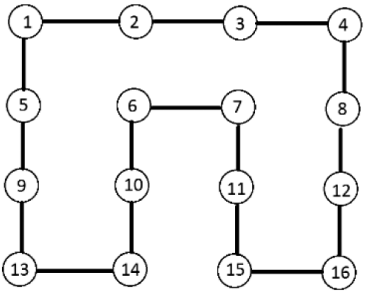


Рис. 1. Найденный оптимальный путь для 16 городов

Время работы алгоритма в MatLab не более минуты. Приведённое решение соответствует оптимальному случаю. Но при запуске с меньшим количеством итерации или меньшим количеством агентов не гарантирует нахождение оптимального пути. Следовательно, найденный путь с помощью данного алгоритма можно расценивать только как близкий к оптимальному, а не гарантированно оптимальный, и чем больше у нас итераций и агентов, тем более вероятней, что найденный путь будет оптимальным.

Решение непрерывной задачи оптимизации

Для того, чтобы применить муравьиный алгоритм для непрерывной функции необходимо определиться по каким характеристиках будет проводиться поиск решения близкого к оптимальному. В статье [4] такой характеристикой является вероятностная функция распределения. Пример такой функции приведён на рис. 2.

Эта функция характеризуется:

$$P(x) \geq 0 \, \forall x,$$

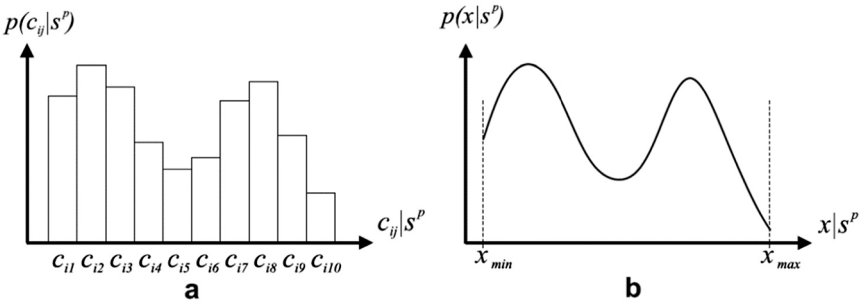


Рис. 2. Вероятностная функция распределения (дискретная и непрерывная)

$$\int_{-\infty}^{+\infty} P(x)\,dx = 1.$$

Распределение строиться на основе функции Гаусса, т. к. функция Гаусса имеет только один минимум, то в данном случае приходится использовать сумму функций Гаусса:

$$G^i(x) = \sum_{l=1}^{k} \omega_l g_l^i(x) = \sum_{l=1}^{k} \omega_l \frac{1}{\sigma_l^i \sqrt{2\pi}} e^{-\frac{(x-\mu_l^i)^2}{2\sigma_l^{i2}}}.$$

Количество функций Гаусса равняется количеству измерений для исследуемой функции.

При оптимизации комбинаторной задачи, феромоны хранились в таблице, и при выборе следующего пути развития выбиралось значение из таблицы и на его основе рассчитывались вероятности для каждого из возможных путей развития. В случае с непрерывной оптимизацией не получиться хранить таблицу феромон, иначе она должна была быть бесконечной. Вместо этого используется другой подход. Обновление таблицы феромонов происходит на основе найденного «хорошего» решения. А вместо испарения выбрасывается самое «плохое» из решений представленных в таблице. Количество значений функции в таблице определяет сложность вероятностной функции распределения.

Пример такой таблица приведён на рис. 3:

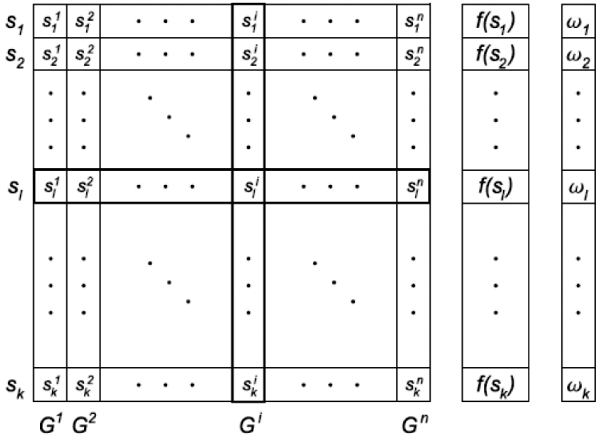


Рис. 3. Таблица найденных решений

где $f(s_1) <= f(s_2) <= \ldots <= f(s_k)$ (при поиске минимума), ω — вес определяющий коэффициент при сложении функций Гаусса, вычисляется по следующей формуле:

$$\omega_l = \frac{1}{qk\sqrt{2\pi}} e^{-\frac{(l-1)^2}{2q^2k^2}}.$$

Алгоритм работы

1) Заполняем таблицу взяв случайные значения для S из допустимого диапазона.
2) Вычисляем

$$\sigma_l^i = \xi \sum_{e=1}^{k} \frac{|s_e^i - s_l^i|}{k-1}.$$

3) Для каждого агента выбираем случайным образом координату для каждого измерения (s).
4) Вычисляем функцию.
5) Обновляем таблицу.
6) После того как все агенты посчитали новые таблицы (или прошли несколько итераций по обновлению таблицы), объединяем в одну таблицу лучшие из полученных решений. После этого у всех агентов на следующей итерации будет одинаковая таблица.
7) Выполнять, до тех пор пока не пройдёт определённое количество итераций или диапазон изменения переменных будет меньше заданной погрешности.

Параметр ξ обозначает то же, что при дискретной оптимизации обозначалось как испарение феромон. Следовательно, при большом ξ будет более медленная сходимость.

Параметр q используется для обозначения стандартного отклонения — q^2 дисперсия. Чем меньше q, тем менее вероятностный будет алгоритм, т. е. будет сходиться к лучшему решению из таблицы, чем больше q тем больше распределение похоже на равномерное.

Использование коллективных алгоритмов в задачах оптимизации

Коллективные алгоритмы в задачах оптимизации можно использовать в тех случаях, когда не критична точность найденного решения, а время поиска этого решения сильно ограничено. С помощью коллективного алгоритма можно найти решение за определённого количество времени, но при этом чем дольше работает алгоритм, чем с большей вероятностью будет найдено оптимальное решений.

Кластеризация

Существует несколько кардинально отличающихся метода кластеризации основанного на ant-based алгоритмах. Один из них основан на использовании центров кластеров, а другой использует двумерную сетку, но из-за сложности реализации и последующего определения кластеров метод с использованием двумерной сетки сложно применить на прикладных задачах.

Описание алгоритма с использованием двумерной сетки [7, 8]:

На двумерной сетке случайным образом располагаются данные. В начале алгоритма, каждый агент берёт по одному объекту. В процессе работы алгоритма агент совершает хаотические передвижения по сетке. При этом с некоторой вероятностью, в зависимости от соседей по клетке, он может бросить объект, после чего, он берёт другой объект вероятностным образом, при том вероятность опять же зависит от соседей по клетке. В процессе такой работы образуются кластеры на двумерной сетке. Пример таких кластеров показал на рис. 4.

Пошаговое описание алгоритма кластеризации с помощью двумерной сетки:

1) Инициализируем данные, т. е. разбрасываем их случайным образом по сетке
2) Каждый агент берёт случайным образом объект данных и ставится на случайную клетку на поле.
3) В цикле выбираем случайным образом агента, который будет делать шаг в случайном направлении, и будет решать с помощью вероятностной функции бросить ли ему объект с данным.
4) Если объект был брошен, то агент должен взять другой объект, также используя вероятностную функцию.
5) Алгоритм заканчивается по достижению заданного количества итераций.

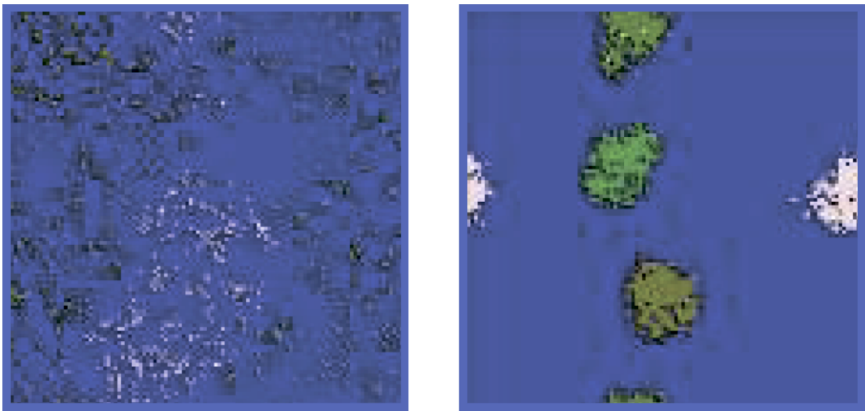


Рис. 4. Улучшенный алгоритм кластеризации

Вероятности описывающие решение взять или бросить объект:

$$p_{pick}(i) = \left(\frac{k^+}{k^+ + f(i)} \right)^2,$$

$$p_{drop}(i) = \left(\frac{f(i)}{k^- + f(i)} \right)^2, \quad f(i) = \max\left(0.0, \frac{1}{\sigma^2} \sum_{j \in L} \left(1 - \frac{\delta(i,j)}{\alpha} \right) \right),$$

$$k^+ = 0.1 \text{ and } k^- = 0.3.$$

Функция $f(i)$ основывается на количестве соседей и расстоянии между данными этих соседей и передвигаемого объекта. Где α параметр, который зависит от расстояния между соседними объектами.

Кластеризация на основе вычисления центров

Второй тип методов кластеризации основанных на муравьиных колониях, осуществляет направленных поиск центров кластеров, по алгоритму представленному ниже [5,6]:

1) В первой части формируются кластеры на основе феромонов и случайной или хаотической последовательности, после этого вычисляются характеристики получившихся кластеров.
2) Во второй части ведётся локальный поиск, т. е. перемещается случайно взятый объект данных из одного кластера в другой случайно выбранный кластер, если при этом характеристики получившихся кластеров улучшатся, то оставляем объект в другом кластере, если же ухудшатся, то возвращаем на исходный кластер.

Функционал, на основе которого оценивается качество кластеризации:

$$D = \min_{1 \le i \le n} \left\{ \min_{\substack{1 \le i \le n \\ i \ne j}} \left\{ \frac{d(c_i, c_j)}{\max\limits_{1 \le k \le n} \{ d'(c_k) \}} \right\} \right\}.$$

Алгоритм реализованный на основе работы [6] не показал приемлемых результатов. Точность определения кластера с помощью такого алгоритма очень мала, поэтому, для работоспособности данного алгоритма были внесены некоторые изменения:

Изменения были внесены во вторую часть алгоритма, теперь будем не случайно выбирать данные а перемещать все данные так, чтобы они принадлежали к ближайшим центрам. При том, если мы переносим одно значение из одного кластера в другой, то центры этих кластеров тоже меняются. Поэтому сравним два алгоритма, в одном из них при каждом переносе пересчитываются центры, а в другом они пересчитываются только после всех возможных перестановок.

Поведём тестирование на выборке из 100 точек и 4-х кластеров, и на 1000 точках и 4-х кластеров:

- Без перестроения центров 32% ошибок (скорость выполнения несколько секунд).
- С перестроением центров 0% ошибок (скорость выполнения несколько минут).
- С увеличением количества точек, точность работы алгоритма без перестроения уменьшилась, и ошибка возросла до 42%.
- Увеличив количество агентов до 5ти ошибка без перестроения уменьшается до 0%.
- Время выполнения алгоритма:
  - с перестроением центров — несколько минут;
  - без перестроения — несколько секунд.

Как видим, при использовании равного количество итераций и агентов, точность определения кластеров с перестроением центров выше, но время работы значительно дольше. Поэтому целесообразней повысить количество итераций и количество агентов для повышения точности без перестроения, чем использовать перестроение центров кластеров.

Сравнение по времени выполнения алгоритмов кластеризации

В таблице 1 приведены времена выполнения на различных тестовых множествах, но, т. к. ant-based и k-means алгоритмы используют для построения кластеров центры, то их время выполнения меньше чем у db-scan и hierarchical, но при использовании центров кластеров в некоторых случаях невозможно правильно произвести кластеризацию.

Как видим из таблицы, ant-base алгоритм работает медленнее чем k-means на данном тестовом множестве. Но в отличие от k-means ant-based алгоритм использует направленных поиск центров кластеров, поэтому он будет всегда медленнее чем k-means, но качество кластеризации будет лучше.

Таблица 1

Время работы алгоритмов кластеризации

	ant-based	**k-means**	**Db-scan**	**Hierarchical**
Elongate	0.01106	0.00396	0.03604	0.88589
Lsun	0.05126	0.01675	0.24006	15.52687
Target	0.30653	0.02418	1.12115	108.82627
EngyTime	0.29226	0.03447	31.54761	–
TwoDiamonds	0.05799	0.00682	0.94960	122.22326
wingNuts	0.093104	0.02028	1.55207	251.06460

Заключение

Все рассмотренные алгоритмы принадлежат к классу коллективного интеллекта. Эти алгоритмы имеют широкую область применения и используются для решения различных как дискретных так и непрерывных задач, в задачах кластеризации и кассификации.

Все алгоритмы принадлежащие к коллективному интеллекту подразумевают параллельную работу множества агентов, и поэтому такие алгоритмы легко разбить на большое количество параллельно выполняющихся подзадач, что при аппаратной поддержке, даёт более высокую производительность [9].

Модифицированный алгоритм кластеризации будет всегда медленнее чем k-means, но в отличие от k-means при использовании нескольких итераций, ant-based алгоритм будет осуществлять направленный поиск центров кластеров, основываясь на данных от предыдущих итераций.

Литература

1. *Marco Dorigo, Luca Maria Gambardella.* Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem Université Libre de Bruxelles Belgium. 1996.
2. *Xinming Zhang, Lirong Wang, Bingyi Huang*. An improved niche ant colony algorithm for multi-modal function optimization. IEEE. 2012.
3. *Deneubourg J.-L., Aron S., Goss S.* and *Pasteels J. M.* The Self-Organizing Exploratory Pattern of the Argentine Ant. March 1989.
4. *Krzysztof Socha, Marco Dorigo*. Ant colony optimization for continuous domains. June 2006.
5. *Xiaoyong Liu*. An Effective Clustering Algorithm With Ant Colony. Department of Computer Science. Guangdong Polytechnic Normal University. China. 2010.
6. *Xiao-Hua, Yi-Xin*. An adaptive ant colony clustering algorithm. Department of Computer Science. Yangzhou University. China. IEEE. 2004.
7. *Handl J*. Knowles Ant-Based Clustering and Topographic Mapping. School of chemistry. The University of Manchester. 2006 Massachusetts Institute of Technology.
8. *Deneubourg J. L., Goss S., Franks N., Sendova Franks A., Detrain C., Chrétien L*. The dynamics of collective sorting robot-like ants and ant-like robots // Proceedings of the first international conference on simulation of adaptive behavior on From animals to animats (1990). Pp. 356–363
9. *Robin M*. GPU-Accelerated Data Mining with Swarm Intelligence. Weiss Honors Thesis Department of Computer Science Macalester College. May, 2010.

МОДЕЛЬ ФОРМИРОВАНИЯ ЛОГИКО-ПРЕДИКАТНОЙ НЕЙРОННОЙ СЕТИ[1]

Т. М. Косовская

профессор кафедры информатики СПбГУ,
ст.н.с. СПИИРАН

E-mail: kosovtm@gmail.com

Аннотация. Рассматривается модель сети с нейронами, реализую-щими предикатные формулы, имеющие вид элементарных конъюнкций. В отличие от классических нейронных сетей предлагаемая модель имеет два блока: блок обучения и блок решения.

При ошибках, возникающих при использовании блока решения, подключается блок обучения. Кроме того, конфигурация сети не фиксируется заранее, а меняется каждый раз после работы блока обучения.

Базой для создания логико-предикатной нейронной сети является логико-предметный подход к решению задач искусственного интеллекта.

Введение

Элемент классической искусственной нейронной сети [1] представляет из себя сумматор взвешенных входов, после которого находится передаточная функция, приводящая значение выхода сумматора в промежуток [0,1]. Конфигурация нейронной сети заранее фиксируется и в процессе обучения меняются только значения весов входов сумматора.

Ниже предлагается модель логико-предикатной нейронной сети, имеющей два блока — блок обучения и блок распознавания. Каждый из блоков в качестве своих элементов имеет предикатную формулу в виде элементарной конъюнкции. Входами элемента сети являются значения переменных для соответствующей элементарной конъюнкции.

Конфигурация блока обучения формируется в процессе обучения сети. После предварительного обучения в этом блоке определяется конфигурация блока распознавания. Блок обучения — это «долго работающий» блок. В отличие от него блок распознавания — это «быстро работающий» блок. Несмотря на то, что блок обучения работает действительно долго (решается NP-трудная задача), это соответствует тому, что человек обучается годами, чтобы потом решать многие задачи в течение секунд.

Формирование предлагаемой сети основано на построении многоуровневого описания классов при логико-предметном подходе к решению задач искусственного интеллекта [2]. Используется способ построения такого многоуровневого описания, изложенный в [4].

[1] Работа поддержана грантом РФФИ 14-08-01276.

Основные понятия логико-предметного подхода

Пусть исследуемый объект представлен как множество своих элементов $\omega = \{\omega_1, ..., \omega_t\}$. На ω задан набор предикатов $p_1, ..., p_n$, характеризующих свойства элементов ω и отношения между ними. Логическим описанием $S(\omega)$ объекта ω называется множество всех атомарных формул или их отрицаний, истинных на ω. Множество всех объектов разбито на классы $\Omega = \cup_{k=1}^{K} \Omega_k$. Логическим описанием класса называется формула $A_k(\boldsymbol{x})$, заданная в виде дизъюнкции элементарных конъюнкций, такая что если $A_k(\boldsymbol{\omega})$ истинна, то $\omega \in \Omega_k$. (Здесь и далее обозначение $\boldsymbol{x}$ используется для списка, состоящего из элементов множества x.)

С помощью построенных описаний предлагается решать следующие задачи. **Задача идентификации:** *проверить, принадлежит ли объект* ω *или его часть классу* Ω_k. **Задача классификации.** *Найти все такие номера классов k, что* $\omega \in \Omega_k$. **Задача анализа сложного объекта.** *Найти и классифицировать все части* τ *объекта* ω, *для которых* $\tau \in \Omega_k$.

Решение задач идентификации, классификации и анализа сложного объекта сведено к доказательству соответственно формул $S(\omega) \Rightarrow \exists\, \boldsymbol{x}_{\neq} A_k(\boldsymbol{x})$, $S(\omega) \Rightarrow \vee_{k=1}^{K} A_k(\boldsymbol{x})$, $S(\omega) \Rightarrow \vee_{k=1}^{K} \exists\, \boldsymbol{x}_{\neq} A_k(\boldsymbol{x})$. Решение всех этих задач основано на решении задачи $S(\omega) \Rightarrow \exists\, \boldsymbol{x}_{\neq} A(\boldsymbol{x})$, где $A(\boldsymbol{x})$ — элементарная конъюнкция.

В [3] доказаны оценки числа шагов алгоритмов, решающих сформулированные задачи. Доказана NP-трудность рассматриваемых задач.

Понятие неполной выводимости

Рассматривается задача проверки того, что из истинности всех формул множества $S(\omega)$ следует истинность $A(\boldsymbol{x})$ или некоторой её максимальной подформулы $A'(\boldsymbol{x}')$ на наборе различных констант из ω.

Пусть a и a' – количества атомарных формул в формулах $A(\boldsymbol{x})$ и $A'(\boldsymbol{x}')$, m и m' – количества переменных формул в формулах $A(\boldsymbol{x})$ и $A'(\boldsymbol{x}')$ соответственно. Параметры q и r определяются соответственно по формулам $q = a'/a$, $r = m'/m$. В этом случае формула $A'(\boldsymbol{x}')$ называется *(q, r)*-фрагментом формулы $A(\boldsymbol{x})$.

Построение многоуровневого описания классов

В [2] описано построение многоуровневого описания классов, позволяющее существенно уменьшить число шагов алгоритмов, решающих каждую из трёх сформулированных задач. Такое построение основано на выделении «часто» встречающихся в описаниях классов подформул $P_i^1(\boldsymbol{y}_i^1)$ «небольшой сложности» и заменой их на новые предикаты $p_i^1(x_i^1)$, где x_i^1 — новые переменные первого уровня. При повторении этой процедуры с выделенными подформулами можно получить 2-уровневое, 3-уровневое, …, L-уровневое описание вида

$$A_k^L(\boldsymbol{x}_k^L)$$
$$\cdots$$
$$p_i^l(y_i^l) \Leftrightarrow P_i^l(\boldsymbol{y}_i^l)$$
$$\cdots$$
$$p_{nL}{}^L(y_{nL}{}^L) \Leftrightarrow P_{nL}{}^L(\boldsymbol{y}_{nL}{}^L).$$

Понятие неполной выводимости формулы позволяет разработать алгоритм выделения подформул с требуемыми свойствами.

1. Для каждой пары элементарных конъюнкций, входящих в описания классов, посредством применения метода неполной выводимости для $A_i(\boldsymbol{x}_i) \Rightarrow \exists\, \boldsymbol{x}_{j\neq} A_j(\boldsymbol{x}_j)$ выделяем их максимальную подформулу $Q^I{}_{i,j}(\boldsymbol{x}_{i,j})$.
1. Повторяем процесс выделения общих подформул для $Q^{l-1}{}_{i1\ldots il}(\boldsymbol{x}_{i1\ldots il})$ и $Q^{l-1}{}_{j1\ldots jl}(\boldsymbol{x}_{j1\ldots jl})$, получив их общие подформулы $Q^l{}_{i1\ldots il,j1\ldots jl}(\boldsymbol{x}_{i1\ldots il,j1\ldots jl})$ ($l = 2$, …, L). Процесс завершится, так как на каждой итерации длины подформул уменьшаются.
2. Выберем среди подформул $Q^l{}_{i1\ldots il,j1\ldots jl}(\boldsymbol{x}_{i1\ldots il,\,j1\ldots jl})$ такие, которые удовлетворяют требуемым условиям и обозначим их посредством $P_i^1(\boldsymbol{y}_i^1)$ ($i = 1$, …, n_1).
3. Формулы $P_i^{l+1}(\boldsymbol{y}_i^{l+1})$ ($i = 1, \ldots n_{l+1}$, $l = 1, \ldots, L - 1$) строятся из выделенных ранее подформул $Q^l{}_{i1\ldots il,j1\ldots jl}(\boldsymbol{x}_{i1\ldots il,\,j1\ldots jl})$ с учётом требуемых условий.

Формирование логико-предикатной сети

В процессе обучения для формирования обучающего блока сети предлагается обучающая выборка, содержащая описания объектов с указанием классов, которым они принадлежат. В описании каждого объекта различные константы заменяются различными переменными и между атомарными формулами ставится знак &. Описанием класса служит дизъюнкция так полученных элементарных конъюнкций.

Затем используется выделение общих подформул $Q^l{}_{i1\ldots il,j1\ldots jl}(\boldsymbol{x}_{i1\ldots il,\,j1\ldots jl})$ из конъюнкций, соответствующих объектам обучающей выборки.

Полученные в обучающем блоке формулы $P_i^l(\boldsymbol{y}_i^l)$ ($i = 1, \ldots, n_l$, $l = 1, \ldots, L$) служат содержимым нейронов $\boldsymbol{l}$-го уровня решающего блока. Последний уровень составляют формулы $A_k^L(\boldsymbol{x}_k^L)$.

В процессе распознавания используется только решающий блок.

Если в процессе использования построенной сети обнаруживается неправильное распознавание объекта, то возможно дообучение сети посредством добавления описания неправильно классифицированного объекта к первому слою обучающего блока и выделению общих подформул этого описания и уже имеющихся. После этого происходит перестройка решающего блока.

Заключение

В докладе описан подход к формированию самоперестраивающейся нейронной сети с элементами, реализующими вычисление значения элементарной конъюнкции предикатных формул.

Литература

1. *Комашинский В. И., Смирнов Д. А.* Нейронные сети и их применение в системах управления и связи. М.: Горячая линия – Телеком, 2002. 94 с.
2. *Косовская Т. М.* Многоуровневые описания классов для уменьшения числа шагов решения задач распознавания образов, описываемых формулами исчисления предикатов // Вестн. С.-Петербург. ун-та. Сер. 10. 2008. Вып. 1. С. 64–72.
3. *Косовская Т. М.* Некоторые задачи искусственного интеллекта, допускающие формализацию на языке исчисления предикатов, и оценки числа шагов их решения // Труды СПИИРАН, 2010. Вып. 14. С. 58–75.
4. *Косовская Т. М.* Понятие неполной выводимости предикатной формулы и его применения к решению задач искусственного интеллекта // В настоящем сборнике, секция «Теоретические основы информатики».

РАЗРАБОТКА БАЗОВЫХ ПРОГРАММНЫХ МОДУЛЕЙ АНАЛИЗА РАБОЧИХ ДВИЖЕНИЙ ОПЕРАТОРА ПРИ ОБУЧЕНИИ РОБОТА МЕТОДОМ ПОКАЗА

В. А. Перхуров
СПбНИУ ИТМО
E-mail: timsablin@gmail.com

Аннотация. Основной задачей данной работы является разработка алгоритмов и отдельных программных модулей анализа траекторий естественных движений головы и руки человека-оператора. В ходе работы выполняется разработка алгоритмов приема, интерполяции и анализа естественных движений человека, позволяющие автоматически получать семантические описания в виде многоуровневой системы фреймов.

Введение

Специфические условия функционирования мобильных робототехнических систем предъявляют высокие требования, в частности, к ориентации в условиях многообразной и неорганизованной внешней среды. В связи с этим выдача объективной информации о внешней среде в режиме реального времени становится одной из основных проблем.

Восприятие, осмысление подразумевают развитый интеллект, возможности искусственного интеллекта наиболее эффективно реализованы в экспертных системах [1], основанных на компьютерных базах данных. Среди многочисленных проблем искусственного интеллекта исследования по представлению и использованию знаний занимают особое место. В то же время, несмотря на совершенство информационных и компьютерных технологий и достижений в области телевизионной обработки изображений, создания совершенных миниатюрных цветных телекамер, пока еще широко не используются автоматические системы распознавания или описания внешней среды в реальных, не упрощенных условиях, за исключением программ чтения текста, введенного сканерами.

На наш взгляд, а также по данным [2], это в первую очередь связано с отсутствием единой методологии и науки о представлении внешней среды. Мы исходим из того, что восприятие у простейших организмов животного мира, детей на ранней стадии развития достаточно для решения их жизненных задач в реальном мире и реальном времени. Ограничение «вычислительных» мощностей успешно компенсируется адекватной организацией структурных процессов восприятия, методами (инстинктами), заложенными в опыте, и постоянной настройкой всего организма на решаемую задачу.

Результаты проводимых исследований ценны также тем, что ясное представление механизмов восприятия должно существенно ускорить обучение людей, особенно детей, всяким видам формообразующей деятельности, связанной с восприятием формы или ее созданием. А также, резко сократить объём информации и упростить процедуры, необходимые для осмысленного обучения или восприятия совершенно новой информации, захлестывающей людей в настоящем и будущем.

Для получения необходимого опыта практического использования положений теории структурного представления знаний и методов описания формы объектов требуется апробация нескольких общих положений теории восприятия на имеющейся аппаратно-программной среде [аппаратуре], пусть для достаточно упрощенных, но цельных систем восприятия реального мира.

Основной задачей исследований является разработка алгоритмов и отдельных программных модулей анализа траекторий естественных движений головы и руки человека-оператора.

Автоматический анализ формы естественных движений человека с получением их семантического описания позволит выполнять обучение и телеуправление робота-манипулятора методом показа движений.

Регистрация и анализ движений

В качестве устройства регистрации движений использовался трекер движений, представляющий собой систему из камеры и реперных точек за которыми осуществляется слежение.

Диоды крепятся либо на специальном ободе для слежения за движениями головы. Для получения координат трекер должен быть проинициализирован на хост- машине с помощью запуска специального приложения — сервера данных, именно с этим приложением и будет осуществляться обмен пакетами для получения координат.

На начальном этапе осуществлялась лишь регистрация движений головы. Для этого была написана несложная программа. Программа была опро-

Рис. 1. Общий вид трекера и реперного устройства

бована на записи нескольких простых движениях головы: наклоны, утвердительный кивок, отрицание.

Снятые с помощью трекера траектории движения предлагается обрабатывать по следующему алгоритму:

1. На первом этапе последовательность точек, пока еще без учета направления преобразуется в последовательность линий аппроксимирующих отснятую траекторию, этим достигается упрощение количества вычислений при дальнейшем анализе траектории.
2. На втором этапе образовавшиеся линии получают свои собственные идентификаторы — уникальные имена соответствующие элементам траектории. Элементы считаются тождественными друг другу, если их длина, совпадает или лежит в некотором диапазоне относительно эталонной длины элемента. Таким образом, формируется первый уровень знаний.
3. На последующем этапе именованные элементы начинают попарно объединяться. Запоминаются относительная ориентация составляющих пару элементов, их размер, центр тяжести. Им также присваиваются имена, и таким образом происходит формирование следующего уровня знаний.
4. На каждом последующем уровне процесс повторяется подобно третьему этапу, элементы предыдущего уровня знаний объединяются попарно и идентифицируются. Так продолжается до тех пор, пока не будет уровней элементов для объединения.

В ходе работы были выполнены начальные этапы по имплементации предложенного алгоритма. Была проведена разработка первых двух этапов алгоритма: аппроксимации снятой траектории и распознавание линий для формирования первого уровня знаний.

После отработки алгоритма мы получаем файл знаний, содержащий в себе элементы из которых состоит наша траектория. На основе этого файла нашу траекторию можно преобразовать в последовательность идентификаторов элементов, а все числовые параметры черпать из файлов знаний.

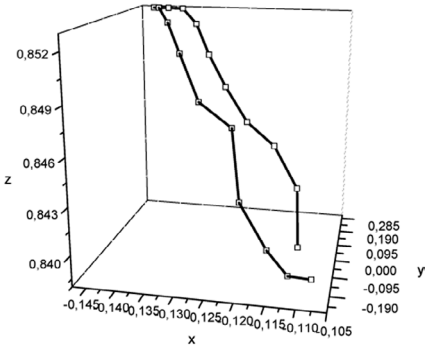


Рис. 2. Пример изначально заснятой траектории

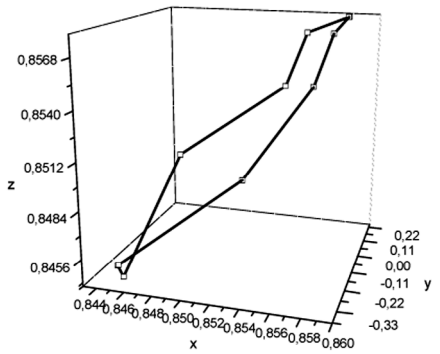


Рис. 3. Пример аппроксимированной траектории

Рис. 4. Общее представление фрейма знаний

Элементы записываются в файлы знаний в следующем формате:

x1, y1, z1, x2, y2, z2, length, c_x, c_y, c_z, id
n — порядковый номер отрезка.
x1, y1, z1 — координаты начала отрезка
x2,y2,z2 – координаты конца отрезка
length – длина отрезка
c_x,c_y,c_z – координаты середины отрезка
id – уникальный идентификатор отрезка

Заключение

В ходе данной работы была выполнена разработка алгоритмов приема, интерполяции и анализа естественных движений человека, позволяющие автоматически получать семантические описания в виде многоуровневой системы фреймов.

Произведена разработка и отладка программных модулей на реальной аппаратуре регистрации движений «Трекер», получены первые результаты экспериментов для типовых движений головы человека.

На основе выполненных работ будет продолжена разработка программных модулей для обучения и телеуправления интеллектуальными роботами-манипуляторами космического и социального назначения.

Отечественные интеллектуальные роботы-ассистенты в ближайшее время будут использоваться для помощи человеку при выполнении сложных технологических операций, обеспечения работ в экстремальных условиях, а также оказывать помощь человеку в повседневной деятельности, в том числе для людей с ограниченными возможностями.

Литература

1. *Уотермен Д.* Руководство по экспертным системам / Пер. с англ. М.: Мир, 1989.
2. *Цыпкин Я. З.* Адаптация и обучение в автоматических системах. Главная редакция физ.-мат. лит-ры изд. «Наука». М., 1968.

Системное программирование

Терехов
Андрей Николаевич

председатель программного комитета конференции
д.ф.-м.н., профессор
заведующий кафедрой системного программирования СПбГУ
директор НИИ Информационных технологий СПбГУ
генеральный директор ЗАО Ланит-Терком

УПРАВЛЕНИЕ ПОЛИТИКАМИ КОНТРОЛЯ ДОСТУПА НА ОСНОВЕ РОЛЕВОЙ МОДЕЛИ

Р. С. Одеров

студент кафедры системного программирования СПбГУ

E-mail: roman.oderov@gmail.com

Введение

Безопасность — один из наиболее важных и сложных аспектов построения и функционирования современных информационных систем. Тема безопасности пронизывает весь жизненный цикл программного продукта: от зарождения идеи до вывода системы из эксплуатации. В IT-сфере под безопасностью (или информационной безопасностью) обычно понимают защищенность информации и ее окружения от целенаправленных или случайных действий, в результате которых может быть нанесен неприемлемый ущерб самой информационной системе или ее пользователям [14].

Предметная область

Существует множество организаций, сотрудники которых имеют дело с различными секретными документами. Требования к системам, работающим с секретной информацией, описаны в ряде стандартов, законов, руководящих документов. Ясно, что чем больше потенциальный ущерб от рассекречивания конфиденциальных сведений, тем серьезнее принимаемые защитные меры. Поэтому в упомянутых выше законах и стандартах требования к безопасности зависят от степени конфиденциальности информации [5, 14].

Большое внимание уделяется описанию политик контроля доступа, определяющих права доступа процессов к данным в рассматриваемой системе. Политики основаны на моделях контроля доступа — математически формализованных системах, включающих в себя субъекты (пользователи/процессы), объекты (данные/документы/файлы) и правила доступа, по которым вычисляется возможность выполнения субъектом набора операций над объектом. Среди моделей контроля доступа можно выделить две наиболее распространенных: MAC (Mandatory Access Control/Мандатный контроль доступа) и DAC (Discretionary access control/Дискреционный контроль доступа) [1, 2, 7, 13, 14].

Система, в которой возможна *одновременная* работа с документами *различных* уровней секретности, обязана реализовывать соответствующие механизмы по обеспечению безопасности этих данных, в том числе поддерживать различные политики доступа (в соответствии с законодательством и стандартами безопасности). В этом кроется серьезная проблема: как без-

опасно работать с разными документами в рамках одной системы и исключить утечки информации.

Сейчас в серьезных (например, оборонных) организациях подобная проблема решается с использованием нескольких физически разделенных машин. На одной производится работа с документами одного уровня секретности, на второй — другого…

Проект MCD, инициированный на кафедре СП Мат-Мех ф-та СПбГУ, призван решить описанную выше проблему в рамках одной системы с помощью современных технологий виртуализации и удаленной доставки приложений путем организации отдельных безопасных «зон» как для хранения документов различного уровня секретности, так и для удаленной работы с ними. Таким образом, в MCD не обойтись без одновременной поддержки нескольких политик контроля доступа, основанных на различных моделях.

При этом возникают серьезные проблемы, в числе которых: определение возможности выражения концептуально разных моделей контроля доступа средствами некоторой «базисной» модели, анализ увеличения сложности управления такой комплексной системой безопасности (ее конфигурирование, визуализация…) и пр.

В данной работе проведено исследование различных моделей контроля доступа [МКД]; показана возможность построения комплексной системы контроля доступа [СКД], в рамках которой могут быть реализованы различные модели и, соответственно, политики. Описано построение прототипов такой СКД на основе ролевой модели и инструмента ее администрирования.

Решение задачи управления различными политиками доступа в рамках системы MCD

В качестве базиса была выбрана RBAC модель [3, 10, 11, 12], на основе которой можно реализовать как MAC, так и DAC модели [4, 6, 8, 9]. Однако стоит отметить, что сложность и размер RBAC, в которой моделируются упомянутые выше MAC и DAC, существенно возрастает, что может повлечь за собой проблемы поддержки и управления. Поэтому необходимо продумать архитектуру и инфраструктуру хранения и работы модели RBAC, а также реализовать удобные инструменты управления, в которых предусмотрены интерфейсы работы как с RBAC, так и с моделируемыми в ней MAC, DAC. Что касается последнего, здесь особенно к месту была бы набирающая популярность концепция визуального программирования, в рамках которой пользователи (в том числе далеко не профессиональные программисты) могли бы в доступной и понятной манере конфигурировать политики доступа путем создания визуальных диаграмм/схем, а затем загружать их в систему управления доступом.

Итак, были поставлены следующие задачи, необходимые для реализации прототипа системы управления политиками контроля доступа для MCD:

- Проектирование архитектуры системы управления политиками контроля доступа.
- Реализация структуры хранения модели RBAC и алгоритмов работы с ней.
- Возможность пакетного внесения данных в модель (например, на основе XML-файлов).
- Возможность удобного/понятного отображения политик доступа и содержимого моделей (визуализация в виде схем, диаграмм и т. п.).
- Возможность визуального программирования политик контроля доступа.

На текущий момент реализовано:

- Структура хранения базовой модели RBAC.
- Возможность пакетного внесения данных на основе файлов XML.
- Управляющая функциональность базовой модели RBAC0 [11].
- Визуализация содержимого модели в виде схем/диаграмм.

В будущем возможны следующие направления развития:

- Реализация расширенных моделей RBAC1 и RBAC2 [11].
- Расширение возможностей по визуализации и визуальное моделирование (управление).
- Шаблоны для моделирования MAC и DAC.
- Контроль состояния системы, проверка неразрешимости задачи определения безопасности системы.
- Интеграция в MCD.

Список литературы

1. Access Control Fundamentals, part 3 [В Интернете] / авт. *Sadeghi Cubaleska*. 2009. — http://www.trust.rub.de/media/ei/lehrmaterialien/232/OSS_chap3.pdf
2. Access Control Models [В Интернете] / авт. *Kantarcioglu Murat*. UT Dallas, 2009. — http://www.utdallas.edu/~muratk/courses/dbsec09s_files/access2.pdf
3. Access rights administration in Role-Based Security Systems [Журнал] / авт. *Matunda Nyanchama Sylvia Osborn*. — [б.м.]: The dep. of CS, The University of Western Ontario, 1994.
4. Configuring Role-Based Access Control to Enforce Mandatory and Discretionary Access Control Policies [Журнал] / авт. *Sylvia Osborn Ravi Sandhu*. QAMAR MUNAWER. 2000.
5. DoD Standard 5200.28-STD «The orange book». — [б.м.]: United States Department of Defense, 1985.
6. How to do Discretionary Access Control Using Roles [Журнал] / авт. *Ravi Sandhu Qamar Munawer*. 1998.
7. Integrity considerations for secure computer systems [Доклад]. — Hanscom Air Force Base, Bedford, Massachusetts: Deputy for command and management systems, Electronic Systems division, Air Force Systems Command, United States Air Force, 1977.

8. RBAC on MLS Systems without Kernel Changes [Журнал] / авт. *Kuhn D. Richard*. 1998.
9. Role Hierarchies and Constraints for Lattice-Based Access Controls [Журнал] / авт. *Sandhu Ravi*. — [б.м.]: George Mason University & SETA Corporation, 1996.
10. Role-Based Access Control (RBAC): Features and Motivations / авт. *David F. Ferraiolo Janet A. Cugini, D. Richard Kuhn*. — [б.м.]: National Institute of Standards and Technology. 1995.
11. Role-Based Access Control Models [Журнал] / авт. *Ravi S. Sandhu Edward J. Coyne, Hal L. Feinstein, Charles E. Youman*. 1995.
12. Role-Based Access Controls [Журнал] / авт. *David F. Ferraiolo D. Richard Kuhn*. — [б.м.]: National Institute of Standards and Technology, 1992.
13. Secure Computer Systems: Mathematical Foundations [Доклад] / авт. *D. Elliott Bell Leonard J. LaPadula*. — [б.м.]: MITRE, 1973.
14. Основы информационной безопасности [В Интернете] / авт. *В. Галатенко* // ИНТУИТ. — 01.04.2003. — http://www.intuit.ru/studies/courses/10/10/info

ОРГАНИЗАЦИЯ ЭФФЕКТИВНОГО ХРАНЕНИЯ ОБРАЗОВ ВИРТУАЛЬНЫХ МАШИН С ВОЗМОЖНОСТЬЮ ИХ МОДИФИКАЦИИ ДЛЯ АВТОЗАПУСКА ПРИЛОЖЕНИЙ

С. А. Серко

студент кафедры системного программирования СПбГУ

E-mail: serko.sergej@gmail.com

Введение

Одна из актуальных проблем в сфере информационных технологий — обеспечение безопасности информационных систем, в частности, организация одновременной работы с данными разных уровней секретности. Именно эту задачу призван решить проект MCD, инициированный на кафедре СП Мат-Мех ф-та СПбГУ.

В основе системы лежат технологии виртуализации и удаленной доставки приложений: безопасность рабочего пространства пользователя достигается путем запуска каждого приложения в отдельной виртуальной машине на сервере и удаленной работой с ним посредством протокола X11. Потенциально большое количество одновременно работающих пользователей и доступных им приложений ставят вопрос об организации эффективного хранения порождаемых образов виртуальных машин, занимающих значительные объемы дискового пространства.

Одним из решений данной проблемы является использование систем хранения данных с большим количеством дисковых накопителей. Однако данный вариант подходит далеко не всем ввиду его дороговизны. Как следствие встает задача оптимального использования доступного дискового пространства.

Сегодня лидирующими технологиями в сфере сжатия данных являются дедупликация и архивация. Они существенно уменьшают объем используемого образами виртуальных машин дискового пространства, но за это приходится расплачиваться дополнительными временными затратами на распаковку/восстановление оригинального файла. Но специфика системы MCD накладывает ограничение на время доступа к файлу — не каждый пользователь готов ожидать до 5 минут для запуска приложения, пусть и в безопасной среде. Исходя из вышесказанного, можно сформулировать задачу организации эффективного хранения образов виртуальных машин, где эффективность понимается как совокупность оптимального использования дискового пространства и высокой скорости доступа к информации.

В данной работе исследованы различные технологии сжатия информации, проведены тестирование и сравнительный анализ решений на основе

этих технологий; доказана возможность построения системы эффективного хранения образов виртуальных машин, а также описано построение прототипа системы эффективного хранения образов виртуальных машин, обладающей функцией внедрения автозапуска приложений.

Организация эффективного хранения образов виртуальных машин с использованием методов хранения/сжатия информации на основе архивации и дедупликации. Внедрение автозапуска приложений

В качестве технологий обеспечения оптимального использования дискового пространства были выбраны дедупликация и архивация ввиду их популярности, большого количества программных решений на их основе, а так же их эффективности при сжатии образов виртуальных машин. Немаловажна и возможность одновременного использования этих техник, причем как в комбинации компрессия + блочная дедупликация, когда файлы предварительно сжимаются архиватором перед тем, как их дедуплицировать, так и в варианте дедупликация + компрессия, когда производится сжатие уже дедуплицированных данных.

Как уже было сказано выше, для модуля хранения данных системы MCD необходимы использование дискового пространства и высокая скорость доступа к информации. Таким образом, ключевыми характеристиками технологий сжатия для нас являются степень сжатия и скорость восстановления файла.

В ходе исследовательской деятельности были поставлены следующие задачи, необходимые для реализации прототипа модуля хранения образов виртуальных машин с функцией внедрения автозапуска приложений:

- Тестирование и сравнительный анализ существующих решений сжатия массивов данных на основе технологий архивации и дедупликации.
- Тестирование и анализ комплексных решений, реализованных на основе существующих.
- Поиск метода хранения/сжатия образов виртуальных машин, отвечающего требованиям высокой скорости доступа к информации и эффективного использования дискового пространства.
- Внедрение автозапуска приложений в образы ВМ.

На текущий момент реализованы:

- Тестирование FUSE-файловых систем со встроенной дедупликацией: SDFS, LessFS.
- Тестирование файловой системы с встроенной дедупликацией ZFS on Linux.
- Тестирование консольных архиваторы, работающих под управление ОС сем-ва Windows: WinRar, WinZip,7-zip.

- Тестирование консольные архиваторы, работающих под управление ОС сем-ва Unix: rar, zip,7-zip, bzip2, gz, lzop.
- Сравнительный анализ протестированных решений.
- Подсистема внедрения автозапуска приложений в ВМ с предустановленной ОС сем-в Windows и Unix.

Среди будущих направлений развития можно выделить первоочередные:

- Тестирование комплексных решений на основе архивации и дедупликации.
- Разработка модуля эффективного хранения ВМ с возможностью модификации для автозапуска приложений.
- Интеграция модуля в MCD.

Список литературы

1. *Ватолин Д., Ратушняк А., Смирнов М., Юкин В.* Методы сжатия данных. Устройство архиваторов, сжатие изображений и видео. М.: Диалог-МИФИ, 2002. 384 с.
2. *Ливак Е. Н.* Алгоритмы сжатия. — http://mf.grsu.by/UchProc/livak/en/po/comprsite/theory_contents.html
3. *Mohammed Al-laham, Ibrahiem M. M. El Emary*. Comparative Study Between Various Algorithms of Data Compression Techniques // World Congress on Engineering and Computer Science. 2007.
4. *Aaron Toponce*. ZFS: The True Cost Of Deduplication/ — https://pthree.org/2013/12/18/zfs-administration-appendix-d-the-true-cost-of-deduplication/
5. *Одеров Р. С.* Безопасное рабочее пространство пользователя Multi-Cloud Desktop. Модуль внедрения автозапуска приложений в виртуальные машины. СПб.: СПбГУ, 2013.

ЭВОЛЮЦИЯ ЯЗЫКОВ ПРИ МЕТАМОДЕЛИРОВАНИИ «НА ЛЕТУ» В DSM-ПЛАТФОРМЕ QREAL

А. И. Птахина

студентка 5 курса кафедры системного программирования СПбГУ

E-mail: alinaptakhina@gmail.com

Аннотация. В статье описывается методика адаптации моделей под новые версии языка и поддержки эволюции визуальных языков. Приводится описание реализации методики в системе QReal — проекте научно-исследовательской группы кафедры системного программирования СПбГУ, в режиме метамоделирования «на лету», позволяющем быстро и легко модифицировать визуальный язык программирования: добавлять новые элементы языка, удалять и изменять существующие непосредственно в процессе разработки.

Введение

В современном мире большой интерес вызывают средства, позволяющие ускорить процесс разработки программного обеспечения. Для этой цели применяются различные технологии, в том числе и визуальное программирование1. При таком подходе программа представляется в виде набора диаграмм. Каждая диаграмма представляет собой модель, описывающую фрагмент функциональности ПО с нужной степенью детализации.

При проектировании часто возникает потребность в создании множества моделей предметной области. Соответственно возникает потребность в языке, который бы позволил упростить процесс описания этих моделей. Язык, используемый для создания других языков моделирования, называется метаязыком. Модели на метаязыке содержат описание всех абстракций визуального языка и правила построения из них визуальных моделей. Модель языка моделирования называется метамоделью.

Довольно часто при визуальном программировании мы имеем дело с визуальными языками моделирования, созданными для использования в рамках конкретной предметной области. Они называются предметно-ориентированными визуальными языками моделирования (DSVL, Domain Specific Visual Language). Эти языки, в отличие от языков общего назначения, разрабатываются для решения определенного круга задач и, благодаря этому, могут привести к более быстрому и эффективному их решению.

Существуют специальные инструментальные средства, позволяющие быстро создавать визуальные языки и редакторы для них, они называются DSM-платформами [2][3]. В данной работе мы будем рассматривать DSM-платформу QReal [4] — проект научно-исследовательской группы кафедры

[1] Подробнее об этом можно прочитать, например, в [1].

системного программирования Санкт-Петербургского государственного университета под руководством профессора А. Н. Терехова.

Метамоделирование «на лету» в QReal

Платформа QReal позволяет автоматически генерировать код произвольных визуальных редакторов по описаниям их метамоделей. Также в данной системе в рамках курсовой работы третьего курса автора [5] был реализован интерпретатор метамоделей, позволяющий эмулировать функциональность сгенерированного редактора. Благодаря ему появилась возможность производить изменения в метамодели без перегенерации кода и пересборки редактора. Было реализовано в рамках курсовой работы четвертого курса автора [6] так называемое метамоделирование «на лету», которое позволило быстро и легко расширять систему.

Метамоделирование «на лету» (см. Рис. 1) предоставило пользователю возможность вносить изменения в визуальный язык программирования прямо в процессе работы с языком: добавлять новые элементы языка, удалять ненужные с его точки зрения элементы и изменять существующие (под этим понимается изменение графического изображения элемента и его свойств). При этом все новые и измененные сущности сразу доступны для построения диаграммы, и в случае возможных конфликтов и некорректности системы пользователю предоставляется соответствующая информация о возможных последствиях. При таком подходе редактирование модели и метамодели объединено, что избавляет пользователя от необходимости мыслить в терминах

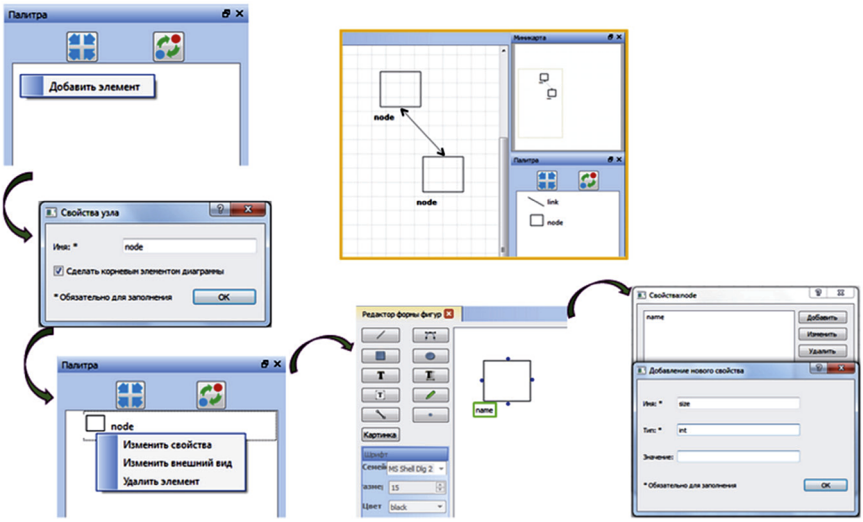


Рис. 1. Метамоделирование «на лету» в QReal

двух абстракций. Уровень метамодели скрыт от пользователя, что позволяет ему полностью сконцентрироваться на задаче, а не на создании инструментария. При метамоделировании «на лету» пользователь может использовать как существующую метамодель, загрузив ее в систему, так и может создать свой собственный язык «с нуля» и тем самым получить быструю реализацию предметно-ориентированного визуального языка.

Эволюция визуальных языков моделирования

В течение жизненного цикла предметно-ориентированного решения используемые визуальные языки могут меняться. Представим ситуацию: пользователь создает визуальный язык для своей предметной области. Задав набор элементов языка, он на их основе создает модель. После этого он сохраняет модель и заканчивает на этом работу. Через некоторое время у него возникает потребность в создании модели, схожей по объектам с предыдущей моделью. Пользователь в режиме интерпретируемой метамодели открывает имеющеюся у него метамодель и производит необходимые изменения: добавляет новые элементы языка, удаляет ненужные, редактирует свойства элементов. После этого он создает нужную в данный момент модель. В такой ситуации произошло изменение визуального предметно-ориентированного языка. Старая модель пользователя стала несовместимой с текущей версией языка. О некоторых элементах, которые используются в ней, система в текущий момент может не иметь информации (они были удалены при изменении визуального языка), также возможны ситуации, когда элементам были добавлены новые свойства, либо были изменены или удалены существующие. Появляется задача адаптации созданных моделей под новые версии языка и поддержки эволюции языков.

Методика обеспечения эволюции визуальных языков

Как упоминалось ранее, визуальные языки в современных DSM-платформах задаются с помощью метамоделей, то есть моделей синтаксиса языка. Они описывают, какие элементы могут находиться на диаграмме, какие свойства есть у этих элементов, и как элементы могут быть связаны друг с другом.

Для реализации поддержки эволюции визуальных языков воспользуемся следующей идеей: будем считать, что метамодель для измененных визуальных языков одна и всегда поддерживает актуальное для всех версий языка состояние. Это означает, что все объекты, созданные на разных этапах моделирования, всегда содержатся в метамодели. Информация об удаленных элементах и свойствах также хранится в метамодели, хотя на диаграмме и в палитре данные элементы не отображаются. Также мы будем хранить все предыдущие имена свойств элементов — это позволит избежать проблем

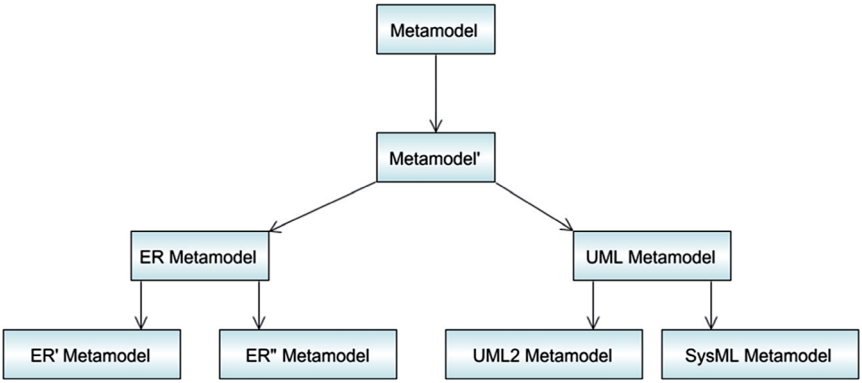


Рис. 2. Пример иерархии эволюции метамоделей

в случае, если свойство было переименовано. И будем производить автоматическое преобразование свойств: автоматически добавлять новые свойства со значением по умолчанию, удалять несуществующие свойства, производить переименование свойств, если оно имело место быть, при смене типа свойства будем приводить, если это возможно, значения к данному типу, а в случае, если невозможно — обнулять.

Предложенный подход позволяет избежать конфликтных ситуаций при загрузке старой модели в версии новой метамодели, предоставляет возможность восстановления удаленных элементов и свойств, а также обеспечивает сохранение информации об иерархии эволюции метамоделей.

Остановимся на иерархии эволюции метамоделей и рассмотрим, что это такое, на примере, приведенном на рисунке 2. Предположим, мы создали метамодель Metamodel, далее мы ее каким-либо образом изменили, допустим, добавили новые элементы в нее — получили другую метамодель Metamodel». Потом мы на основе метамодели Metamodel» создали две другие метамодели для различных целей: ER Metamodel и UML Metamodel, каждую из которых детализировали и получили метамодели следующего уровня: ER» Metamodel, ER»’ Metamodel и UML2 Metamodel, SysML Metamodel соответственно. В данном примере метамодель SysML Metamodel является модифицированной версией метамодели UML Metamodel, и, значит, пользователь должен иметь возможность открывать модели, построенные на основе UML Metamodel, а также на основе метамоделей MetaModel» и Metamodel. Что касается моделей, построенных на основе других метамоделей, приведенных в примере, то никакой информации об их объектах не должно содержаться в метамодели SysML Metamodel, поскольку данные метамодели не являются ее ранними версиями и, по сути, не имеют никакого отношения к ней.

Рассмотрим, как предложенный подход можно применить в DSM-платформе QReal, с помощью диаграммы, представленной на рис. 3. Пусть в ре-

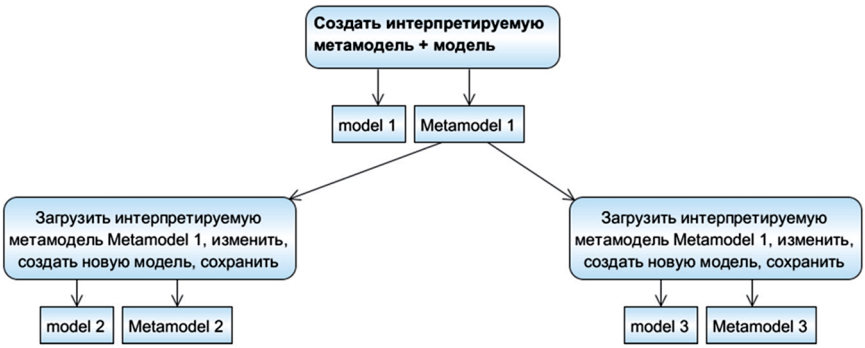


Рис. 3. Иллюстрация к применению методики в QReal

жиме метамоделирования «на лету» мы создали предметно-ориентированный визуальный язык и сохранили полученную метамодель Metamodel 1 и модель model 1. Далее мы открыли созданную метамодель Metamodel 1 и постепенно создали сначала одну измененную версию языка — получили Metamodel 2 и model 2, а потом другую, никак не связанную с предыдущей, версию языка, получив Metamodel 3 и model 3.

В данном примере метамодели Metamodel 2 и Metamodel 3 содержат всю информацию об объектах и свойствах, которые присутствуют в метамодели Metamodel 1. Таким образом, если мы откроем в режиме метамоделирования «на лету» любую из данных метамоделей и попытаемся открыть модель model 1, то после некоторых преобразований, которые будут рассмотрены далее, мы сможем увидеть интересующую нас модель. Однако сведениями об объектах, специфичных для данных метамоделей, и модификациях, которые произошли после того, как мы изменили Metamodel 1, ни одна из данных метамоделей по отношению к другой не владеет. Это означает, что на базе одного визуального языка мы получили две совершенно разные, никак не связанные между собой версии этого языка. А значит модели, созданные для одного визуального языка, не должны поддерживаться в другом.

Особенности адаптации моделей под новые версии языка и поддержки эволюции языков

Рассмотрим подробнее преобразования, которые происходят при адаптации модели под новые версии визуального языка. В качестве предметно-ориентированного визуального языка рассмотрим DSVL язык простой графики с заданием формы фигуры с помощью графических примитивов и отношений над примитивами (см. Рис 4).

В изображенном на рисунке визуальном языке имеются четыре сущности, представляющие простые геометрические фигуры: квадрат, эллипс

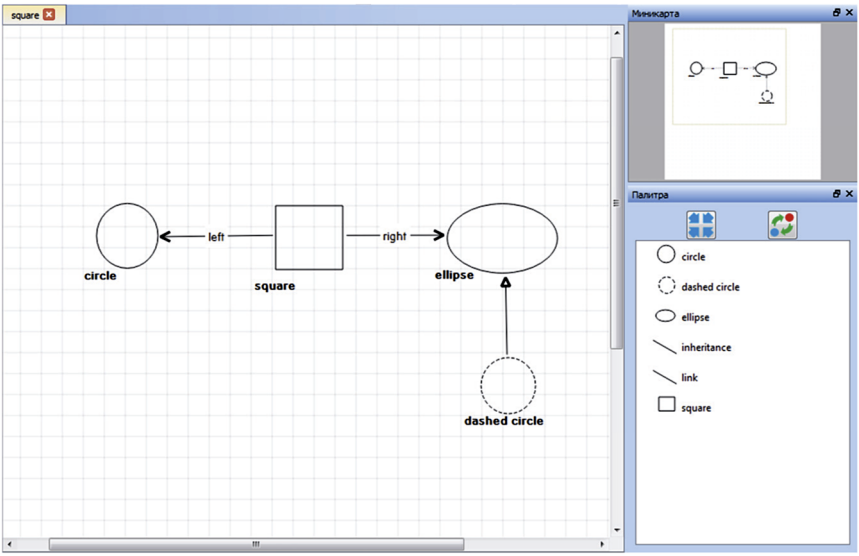


Рис. 4. Визуальный язык простой графики

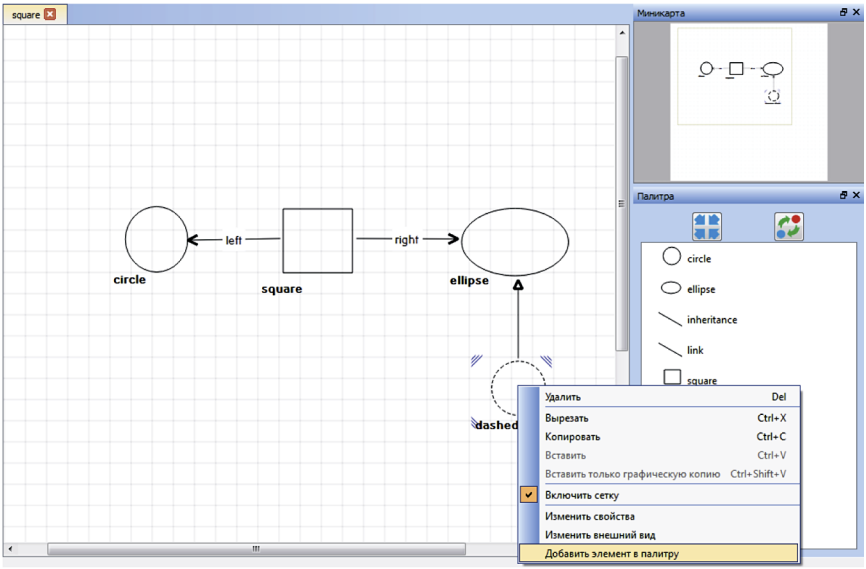


Рис. 5. Открытие модели в измененном визуальном языке простой графики

и 2 круга, отличающихся типом границы. Также в визуальном языке присутствуют 2 типа связи: связь, отвечающая за позиционирование объектов относительно друг друга, и связь, показывающая то, что один элемент находится внутри другого элемента. Модель на рисунке 3 задает фигуру, в центре которой расположен квадрат, слева от него круг, а справа эллипс, внутри которого находится пунктирный круг.

Предположим, мы создали простую модель, описанную выше, сохранили ее и решили удалить один из кругов, а именно сущность «пунктирный круг» из нашего визуального языка. В результате этого действия наша метамодель изменилась — в ней данный элемент отмечен, как удаленный. Мы получили другую версию визуального языка. Однако, несмотря на все произошедшие изменения, мы можем открыть модель, созданную нами ранее (см. Рис. 5). Как мы видим, удаленная сущность «пунктирный круг» отсутствует в палитре, однако она присутствует на диаграмме, и мы можем добавить ее в палитру и использовать далее в процессе моделирования через контекстное меню, выбрав пункт «Добавить элемент в палитру».

Рассмотрим, что происходит в случае, если в измененной версии языка меняется набор свойств элементов. В такой ситуации, чтобы избежать возможных конфликтов при открытии модели, созданной на основе ранней

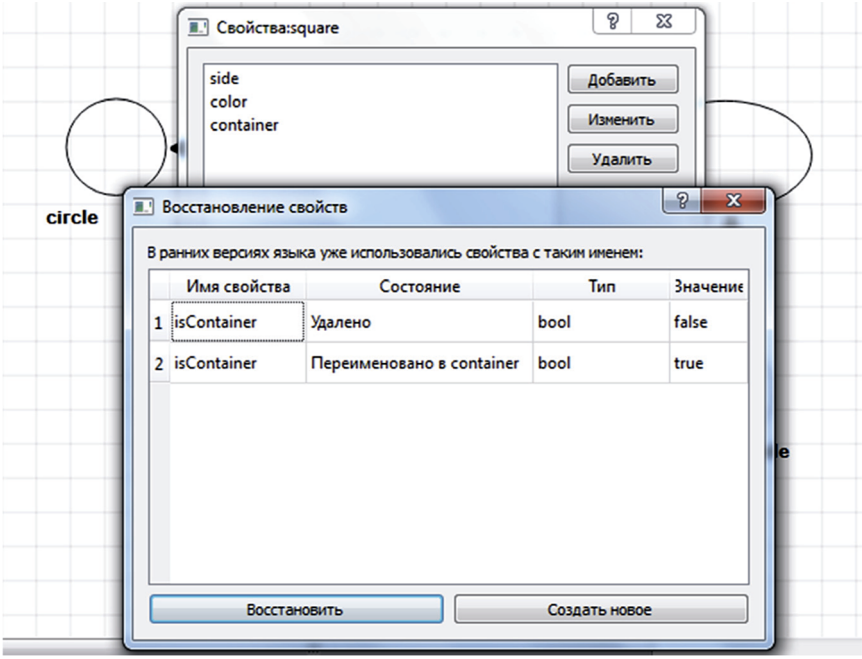


Рис. 6. Диалоговое окно восстановления свойств

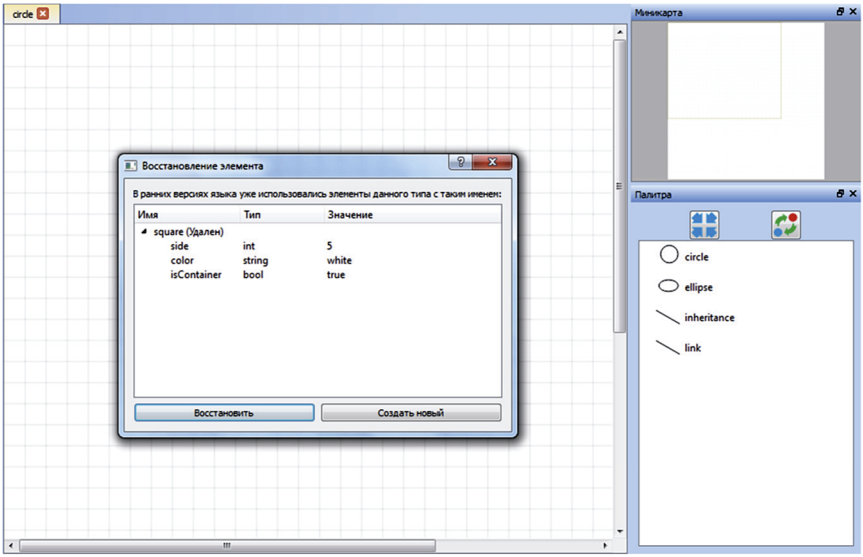


Рис. 7. Диалоговое окно восстановления элемента

версии языка, необходимо производить преобразование свойств. В данной работе был предложен следующий интуитивно понятный и логичный способ преобразования: при открытии старой модели в измененной версии языка необходимо элементам этой модели автоматически добавлять новые свойства со значением по умолчанию, удалять несуществующие свойства и производить переименование свойств, если оно имело место быть.

Для поддержки эволюции языков было также реализовано восстановление удаленных и переименованных свойств элементов. Таким образом, в случае, если пользователь попытается добавить свойство с именем, использовавшимся ранее, то ему будет предоставлен список всех удаленных и переименованных свойств с этим именем . В диалоговом окне восстановления свойств, представленном на рисунке 6, для каждого элемента списка одноименных свойств указывается вся необходимая информация о нем: состояние (удалено/переименовано в [текущее имя свойства]/используется), тип свойства и значение по-умолчанию. Пользователь по желанию может либо восстановить прежнее свойство (при этом, в случае переименованного свойства произойдет смена его текущего имени), либо создать одноименное новое.

Существует также возможность восстановления удаленных элементов языка, в случае если пользователь пытается создать одноименную сущность или связь. В диалоговом окне восстановления элемента, представленном на рисунке 7, для каждого одноименного элемента отображается информа-

ция о всех его свойствах с типом и значением по умолчанию. Аналогично диалоговому окну восстановления свойств, пользователю предоставляется возможность либо восстановить один из элементов приведенного списка со всеми его свойствами, либо создать свой новый элемент с таким именем.

Сравним предложенный подход с методами, которые применяются для поддержки эволюции языков в других DSM-платформах. Расмотрим систему MetaEdit+ [7] — среду для создания и использования предметно-ориентированных языков, разработанную в рамках научно-исследовательского проекта MetaPHOR. При построении метамодели в MetaEdit+ используется язык метамоделирования GOPPRR (Graph-Object-Property-Port-Relationship-Role), получивший свое название от основных понятий, которыми он оперирует. MetaEdit+ использует подход, основанный на интерпретации метамоделей. При этом при внесении изменений в описание метамодели все необходимые преобразования в соответствующих моделях DSM-платформа производит сама. Следует отметить, что созданный в системе объект невозможно удалить, но информацию о нем можно скрыть от пользователя: если явно не указывать, что данный объект используется в графе, то в палитре редактора диаграммы он отображен не будет. В случае же, если мы захотим добавить удаленный элемент из старой модели в палитру, нам придется явно добавить его в граф с помощью инструмента Graph Tool. Такое поведение близко к тому, которое мы хотели реализовать, однако при таком подходе пользователь вынужден работать отдельно с моделью и метамоделью. Нашей же задачей была возможность поддержать совместную эволюцию модели и метамодели и производить все необходимые преобразования автоматически, чтобы избавить пользователя от необходимости мыслить в терминах двух абстракций.

Заключение

В рамках данной работы была разработана методика эволюции визуальных предметно-ориентированных языков и реализовано средство адаптации моделей под новые версии языка и поддержки эволюции языков при метамоделировании «на лету» в DSM-платформе QReal. Это позволило обеспечить совместную эволюцию модели и метамодели. Была произведена апробация предложенной методики и выполнено тестирование реализованного средства на примере простого DSVL-языка.

Литература

1. *Кознов Д.* Основы визуального моделирования // Бином. Лаборатория Знаний, Интернет-Университет Информационных Технологий-2008.
2. *Павлинов А., Кознов Д., Перегудов А., Бугайченко Д., Казакова А., Чернятчик Р., Фесенко Т., Иванов А.* О средствах разработки проблемно-ориентированных ви-

зуальных языков / Под ред. А. Н. Терехова и Д. Ю. Булычева // Системное программирование. Вып. 2. Спб.: Изд. СПбГУ, 2006. С. 121–147.
3. *Kelly Steven, Tolvanen Juha-Pekka*. Domain-Specific Modeling: Enabling Full Code Generation. Wiley-IEEE Computer Society Pr., 2008.
4. *Терехов А. Н., Брыксин Т. А., Литвинов Ю. В.* и др. Архитектура среды визуального моделирования QReal // Системное программирование. Вып. 4. СПб.: Изд-во СПбГУ. 2009, С. 171–196.
5. *Птахина А. И.* Интерпретация метамоделей в metaCASE-системе QReal (курсовая работа). СПбГУ, кафедра системного программирования, 2012. — http://se.math.spbu.ru/SE/YearlyProjects/2012/YearlyProjects/2012/345/345_Ptakhina_report.pdf [дата просмотра: 21.04.2014]
6. *Птахина А. И.* Разработка метамоделирования «на лету» в metaCASE-системе QReal (курсовая работа). СПбГУ, кафедра системного программирования, 2013. — http://se.math.spbu.ru/SE/YearlyProjects/2013/YearlyProjects/2013/445/445-Ptakhina-report.pdf
7. Domain-Specific Modeling with MetaEdit+. — http://www.metacase.com/

ПРОФИЛИРОВАНИЕ ОПЕРАЦИОННЫХ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

Д. Е. Дерюгин

студент кафедры системного программирования СПбГУ, 3-й курс

E-mail: deryugin.denis@gmail.com

Аннотация. Профилирование помогает понять, как работает та или иная программа. Это может быть полезно во многих случаях, но особенно важным оно становится именно для операционных систем реального времени. В работе рассматриваются методы профилирования таких систем, а также приводится описание реализации профилировщика на основе ОСРВ Embox.

Введение

Профилирование — это сбор различных характеристик работы программы, таких как время выполнения определённых участков кода, объём занимаемой программой памяти и так далее. Использование профилировщиков (инструментов профилирования) важно во многих областях: разработчики оборудования могут узнавать, как ведёт себя старое ПО на новой платформе, разработчики ПО могут находить те самые 5% кода, которые исполняются 95% времени («бутылочные горлышки»), или, например, проверить, насколько эффективна стратегия замещения данных в кэш-памяти.

По мере роста объёма программы всё сложнее становится понимать поведение как отдельных её модулей, так и поведение системы в целом. При отсутствии соответствующих инструментов тестирование, отладка и оптимизация становятся в разы сложнее.

Понимание поведения программ становится особенно важными при изучении работы операционных систем реального времени, ведь для таких ОС важна детерминированность (то есть определённость в поведении), чего сложно достигнуть без соответствующих инструментов. Более того, в таких системах важно собирать информацию не только о работе прикладного ПО как такового, но и о работе ядра.

Практическая часть моей курсовой работы основана на модульной конфигурируемой операционной системе Embox для встроенных систем, так как этот проект имеет открытый исходный код, а также имеется возможность непосредственно контактировать с разработчиками.

Проект Embox существует уже несколько лет, за это время было написано более 150 000 строк кода (преимущественно на языке C), реализовано множество алгоритмов от классики для операционных систем, описанной в литературе, до оригинальных разработок. Имеется поддержка ряда платформ (ARM, x86, MicroBlaze, MIPS, PowerPC, SPARC).

Обзор методов профилирования

Большую часть профилировщиков можно отнести к одной из двух категорий: сэмплирующие (статистические) и инструментирующие.

Сэмплирование

Идея сэмплирования заключается в том, чтобы с некоторой периодичностью приостанавливать работу программы тем или иным образом, затем производить анализ состояния программы (стэк вызовов, объём занимаемой памяти и так далее), а затем на основе собранной информации строить статистическую модель.

Профилировщики данного типа хорошо подходят для программ уровня пользоваталя. Вот некоторые из них, получившие распространение: Intel VTune Amplifier, AMD CodeAnalyst, Apple Inc. Shark, OProfile.

Этот подход хорош тем, что профилируемый код сам по себе не изменяется. Также относительно мало количество накладных расходов.

Сэмплирующие профилировщики могут быть основаны на двух механизмах:

1. *Программное сэмплирование* — производится с помощью системных прерываний. Большинство профилировщиков использует именно этот механизм, так как для этой реализации нет нужды в специфичной аппаратной поддержке.
2. *Аппаратное сэмплирование* — должно поддерживаться платформой. Например, новые MIPS-процессоры содержат специальный регистр PCSAMLE, позволяющий сэмплировать счётчик процессорных тактов. Преимущество этого подхода в том, что накладные расходы становятся ещё меньше, а также нивелируются недостатки реализации системных прерываний.

Впрочем, у статистически профилировщиков есть и недостатки. Ясно, что сам результат будет статистическим, а значит, в специфичных ситуациях замеры могут производиться в неподходящее время, упуская некоторую существенную информацию. Например, таким неподходящим временем может быть исполнение непрерываемого участка кода.

Как правило, конкретные реализации этого подхода имеют свои, менее очевидные недостатки, что было показано в одной из курсовых работ прошлых лет [5].

Инструментирование

Второй подход — это инструментирование. Идея заключается в изменении программного кода вставкой специальных инструкций для сбора необходимой информации. Понятно, что изменение программного кода влечёт за собой изменение поведения программы, однако, данный подход не яв-

ляется статистическим и нечувствителен к непрерываемости кода. Такие инструменты хороши для программ уровня ядра.

Инструментирование может быть разделено на несколько типов:

- *Вручную* — программисту необходимо собственноручно вставлять команды, собирающие нужную информацию.
- *На уровне исходного кода* — специальный инструмент автоматически изменяет код программы. Пример: Parasoft Insure++.
- *На уровне промежуточного кода* — актуально при использовании языков высокого уровня. Пример: OpenPAT.
- *С помощью компилятора* — профилировщик сам компилирует программу. Примеры: gprof, Quantity.
- *Инструментирование бинарного файла* — производится изменение уже скомпилированного исполняемого файла. Во время исполнения — код инструментируется прямо перед исполнением. Исполняемая программа должна находиться под полным контролем профайлера. Примеры: Pin, Valgrind, DynamoRIO.

Реализация

То, что было в Embox раньше

При разработке средства профилирования было принято решение не начинать разработку с самого начала, а отталкиваться от того, что было уже реализовано в системе.

Двумя годами ранее был реализован механизм, позволяющий расставлять trace_point и trace_block — специальные конструкции — для измерения количества исполнений определённой команды и для замера время работы нужных участков кода.

Впрочем, имелся ряд недостатков. Один из ключевых — негибкость, то есть сложность в использовании. Например, в силу специфичности реализации, невозможно, например, создать более одного trace_block в одной области видимости. Ещё один минус заключался в том, что все необходимые конструкции необходимо было расставлять вручную, что, конечно же, затрудняет анализ большого объёма кода.

Устранение недостатков

Конечно же, новый инструментирующий профилировщик должен был по меньшей мере избавиться от недостатков предыдущих наработок.

Невозможность использовать больше одного trace_block»а вытекала из не очень удачной реализации макроса, отвечавшего за объявление блоков.

Также хотелось повысить точность измерений. Ряд платформ поддерживает возможность измерять время в процессорных тактах. В архитектуре

x86 для этого используется TSC (Time Stamp Counter), значение которого хранится в специальном регистре (MSR), для платформы SPARC — это TICK register. Возможность использовать счётчик тактов повысила точность на порядки по сравнению с измерением системного времени.

Впрочем, есть несколько аспектов, которые нужно учитывать при использовании подобных счётчиков.

Out-of-order Execution.

Внеочередное исполнение инструкций (Out-of-order Execution) может как увеличить, так и уменьшить показатель времени исполнения инструкций.

Рассмотрим программу:

```
rdtsc             ; Считывание счётчика
mov    time,eax   ; Сохранение значения
fsqrt             ; Извлечение корня
rdtsc             ; Второе считывание счётчика
sub    eax,time   ; Нахождение разницы
```

Например, второй замер может быть произведён до вычисления квадратного корня. Соответственно, полученное значение не будет зависеть от времени исполнения инструкции fsqrt. Для избежания этого эффекта нужно использовать упорядочивающие инструкции (serializing instructions), которые гарантируют очистку процессорного конвеера перед их выполнением. Примеры таких инструкций: cpuid для x86, SYNC для MIPS.

Кэширование.

Использование счётчика процессорных тактов для измерений может приводить к существенно разным результатам при повторном исполнении программы (в зависимости от того, находятся ли необходимые данные в кэше или нет), поэтому при изучении поведения программы нужно это учитывать. Полностью избавиться от такого эффекта можно только для маленьких участков кода. Для этого достаточно воспользоваться техникой «cache warming», суть которой заключается в следующем: нужно исполнить инструкции без замера (необходимые данные загрузятся в L1-кэш), а после этого — исполнить их ещё раз, но уже замерив время. Для этого можно использовать цикл или вызов функции два раза подряд.

Переключение контекстов.

При профилировании больших участков кода нужно принять во внимание переключение контекстов, которое, конечно, повлияет на результат измерений. Для того, чтобы избежать этого, можно повышать приоритет изучаемого процесса.

Многоядерные процессоры.

Дело в том, что счётчик тактов между ядрами не синхронизируется, а это значит, что для избежания ошибок в измерении нужно избегать миграции процесса между ядрами.

Создание нового инструмента профилирования

При реализации автоматического инструментирования можно сделать важное наблюдение: обычно разбиение на модули подразумевает, что используемые функции не будут очень большими по объёму. Из-за этого чаще всего необходимо замерять не время исполнения произвольного участка кода, а время работы определённых функций, либо время исполнения функций из определённого модуля. Исходя из этого предположения, можно существенно уточнить задачу.

GNU Compiler Collection (GCC) предоставляет ряд возможностей для сбора информации о вызываемых функциях. Одна из них — сборка всех объектных файлов изучаемой программы с ключом -finstrument-functions. При этом перед вызовом каждой функции будет вызываться __cyg_profile_func_enter, а после их завершения — __cyg_profile_func_exit, которые можно использовать для определения времени работы функции.

При таком инструментировании кода при помощи компилятора требуется, чтобы программистом эти функции были определены. В качестве аргументов передаётся указатель на вызываемую функцию и указатель на функцию вызывающую. На основе этой информации и генерируются trace_block во время исполнения программы.

Одним из решений может быть добавление нужных флагов для инструментирования сразу всех модулей системы. В этом случае следует учитывать, что есть ряд функций, которые всё же нежелательны для инструментирования: это все функции, связанные с обработкой профилировочной информации; также это ряд системных функций. Для исключения этих функций из перечня профилируемых можно использовать флаг компиляции -finstrument-functions-exclude-function-list=func1,func2... Однако, никаким разумным образом не получится перечислить все часто вызываемые системные функции, которые изучать не нужно, а значит возникнут бессмысленные, но ощутимые накладные расходы.

Отсюда вытекает вывод о необходимости «точечного» инструментирования.

ОСРВ Embox имеет высокую конфигурируемость, что наталкивает на идею добавить поддержку автоматического инструментирования с помощью изменения специальных конфигурационных файлов. Mybuild — это система сборки для модульных приложений, которая используется в Embox.

Mybuild предлагает переход на уровень абстрации модулей, где нет речи о возможности добавить произвольный флаг для конкретного файла. Мной

была реализована опция @InstrumentProfiling(), которая сообщает сборщику о необходимости добавить нужные для профилирования параметры.

Простейший конфигурационный файл, описывающий программу, состоящую из одного исходного файла, может выглядеть так:

```
package embox.cmd
@Cmd(name = “hello_world”,help = “”,man = ‘’’’’’)
module hello_world {
    @InstrumentProfiling(“true”)
    source «hello_world.c»
}
```

После запуска программы hello_world можно увидеть результат профилирования с помощью команды trace_blocks -n. Эта программа просто перебирает все trace_block, созданные за время работы системы и выводит информацию о них.

Часто информации о работе приложения самого по себе бывает недостаточно, и хочется понимать также и поведение функций ядра ОС при работе этого приложения. В этом случае было бы глупо расставлять во всех конфигурационных файлах одну и ту же опцию.

Мной была реализована программа tbprof. Идея заключается в том, что tbprof запускает наблюдаемую программу, собирая информацию лишь о тех вызовах функций, которые происходят во время исполнения исследуемой программы (включая такие вещи, как, например, системные обработчики прерываний, переключение потоков, выделение памяти и так далее).

На первый взгляд, такое больше количество накладных расходов может слишком сильно исказить результаты измерений, но на самом деле, это искажение будет не случайной природы — оно будет детерминированным, поэтому, храня информацию о количестве вызовов функций, можно будет восстановить точную оценку времени их работы.

Тем не менее, при неудачной реализации время работы при профилировании может возрасти на порядки (например, если программа обращается преимущество к быстрым функциям, таким как, например, abs(), но обращается к ним очень часто, в то время как сохранение профилировочной информации о вызове этих функций не столь быстрая операция). В таком случае, детерминированность сохраняется, но время выполнения становится неприемлемым.

Тем не менее, удалось добиться достаточной скорости обработки и сохранения профилировочной информации налету.

В качестве демонстрации результатов работы профилировщика в таблице 1 приведена зависимость времени переключения потока от стратегии выбора очередного потока при переключении.

Таблица 1

Время переключения потока при различных стратегиях

Стратегия	Количество переключений	Суммарное количество тактов	Среднее время выполнения (микросекунды)
`trivial`	324	66384378	39.371
`priority_based`	360	70343894	41.436
`priority_based_smp`	334	71182952	46.052

Заключение

В работе были рассмотрены методы профилирования программ в цеорм и операционных систем реального времени в частности, а так же описана реализация профиилровщика на основе ОСРВ Embox.

Литература

1. Intel, «Using the RDTSC Instruction for Performance Monitoring», URL: http://www.ccsl.carleton.ca/~jamuir/rdtscpm1.pdf
2. GNU Make Manual, URL: https://www.gnu.org/software/make/manual/
3. GCC 4.8.2 Manual, URL: http://gcc.gnu.org/onlinedocs/gcc-4.8.2/gcc/
4. *Крамар А. С.* Трассировки ОСРВ Embox, URL: http://se.math.spbu.ru/SE/YearlyProjects/2012/YearlyProjects/2012/345/345_Kramar_report.pdf
5. *Одеров Р. С.* «Исследование и тестирование семплирующего метода профайлинга», URL: http://se.math.spbu.ru/SE/YearlyProjects/2012/YearlyProjects/2012/345/345_Oderov_report.pdf
6. QEMU Emulator user documentation, URL: http://wiki.qemu.org/download/qemu-doc.html
   http://www.youtube.com/watch?v=DwLbcFBwhi4

ОБЗОР МЕТОДОВ АНАЛИЗА ПРЕДМЕТНОЙ ОБЛАСТИ

А. Гудошникова

студентка 4 курса кафедры системного программирования СПбГУ

E-mail: gudoshnikova.anna@gmail.com

Аннотация. При создании программного обеспечения всегда возникают проблемы с пониманием некоторых терминов и связей в рассматриваемой предметной области. Поэтому этап анализа предметной области является неотъемлемой частью всего процесса разработки. В настоящее время этот этап, как правило, происходит в неформальном виде, в результате чего все равно может остаться недопонимание между аналитиком и программистом. В данной работе представлены некоторые формальные методы анализа предметной области и их применение.

1. Введение

В основе качества и надежности поставляемого ПО лежит, в первую очередь, понимание той сферы, где предстоит разработать тот или иной продукт. Такая сфера деятельности называется предметной областью. На глубокое погружение в предметную область всегда требуются большие ресурсы и время. Как правило, анализ предметной области выполняется неформально, но при этом остаются трудности с пониманием тех или иных терминов, связей и сущностей. Поэтому были разработаны некоторые формальные методы анализа предметной области. Для того, чтобы использовать один из них, надо прежде всего понять, какой результат нужен от такого анализа (определить цели анализа предметной области). Также необходимо осознать, что будет сделано в процессе анализа предметной области. И в завершении всего, определить процесс построения модели предметной области. В данной статье будет рассмотрено, что понимается под анализом предметной области, что должна включать в себя модель предметной области, а также представлены некоторые формальные методы анализа предметной области и их применение.

2. Анализ предметной области

В настоящее время нет четкого, ясного и единственного определения термина «анализ предметной области». Фере [1] выделил такие:

1) Процесс идентификации, организации и представления релевантной информации о предметной области;
2) Процесс, при котором знания клиента/пользователя идентифицируются, конкретизируются и систематизируются;
3) Деятельность, которая предваряет системный анализ.

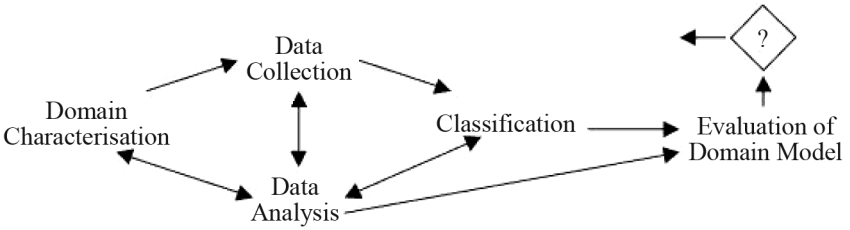


Рис. 1. Схема общего процесса получения модели предметной области

Аранго [2] показал, что все методы анализа предметной области придерживаются так называемого общего процесса получения модели предметной области. Схема этого процесса приведена на Рис. 1.

Этап характеристики предметной области (Domain Characterisation) подразумевает анализ реализуемости проекта и фазу планирования. На этапе сбора данных (Data collection) выбираются источники релевантной информации в данной области, которые могут варьироваться от экспертов предметной области до различных документов и т. д. Этап анализа данных (Data analysis) называется вспомогательным моделироваем. Целью этого этапа является выделение сходств и различий между используемыми элементами. На следующем этапе (Classification), или на этапе основного моделирования, смоделированная на предыдущем шаге информация уточняется, выделяются целые блоки с похожими описаниями, добавляются новые описания к существующим или новым блокам, и затем эти блоки выстраиваются в обобщенную иерархию. На заключительном этапе (Evaluation of Domain Model) происходит оценка созданной модели предметной области, любые недочеты исправляются.

Анализ предметной области также используется в задаче переиспользования. В этом случае выбор метода для анализа предметной области будет зависеть и от типа объекта, который будет впоследствии переиспользован, и от задачи переиспользования. В качестве задачи переиспользования может рассматриваться, например, построение библиотеки переиспользованных компонент.

Сейчас большое внимание уделяется визуальному программированию, которое позволяет упростить создание программного обеспечения при помощи представления отдельных ее аспектов виде диаграмм. Существуют языки, которые создаются конкретно под предметную область. Такие языки называются предметно-ориентированными. Эти языки призваны наиболее точно описать определенную предметную область. Работая с определенной областью знаний, такие языки легче использовать, чем языки общего назначения, а также уменьшается время разработки и стоимость поддержки приложений, созданных на таких языках. При представлении аспектов программного обес-

печения в виде диграмм чаще всего используются предметно-ориентированные языки. Язык, с помощью которого описывается предметно-ориентированный, или просто визуальный язык, называется метаязыком. На метаязыке строятся различные модели для описания визуального языка. Такие модели называются метамоделями. Метамодель предметно-ориентированного языка хранит в себе знания о предметной области, т. е. способствует переиспользованию знаний. Поэтому методы анализа предметной области при проектировании предметно-ориентированного языка имеют особое значение.

Модель предметной области и методы анализа

1. Модель предметной области

По Мернику [3] модель предметной области должна включать в себя не только словарь терминов предметной области, но также должна описывать сходства и изменчивость этих понятий. Такая модель должна точно задавать границы области, т. е. четко и ясно описывать тот круг вопросов, который будет рассматриваться в рамках этой области.

2. Формальные методы анализа предметной области

При разработке приложения так или иначе происходит анализ предметной области, как правило в неформальном виде. Так как при использовании неформальных подходов могут остаться трудности с пониманием ключевых терминов и связей, то были разработаны формальные методы анализа предметной области:

1) DARE (Domain Analysis and Reuse Environment) [4]. В рамках этого метода вся собранная информация об области собирается в так называемую книгу предметной области (domain book), которая содержит также и универсальную архитектуру для приложения в этой области и библиотеку переиспользуемых компонент;
2) DSSA (Domain-Specific Software Architectures) [5]. Этот метод основан на сценариях. Пользователи описывают некоторые сценарии, которые будут в дальнейшем использоваться для разработки словаря данной области;
3) FODA (Feature-Oriented Domain Analysis) [6] Метод основан на выделении той функциональности, которую нужно будет реализовать в рамках рассматриваемой области;
4) FAST (Family-Oriented Abstractions, Specification and Translation) [7]. Этот метод анализирует сходства и различия между продуктами в рамках одной линейки продуктов;
5) FORM (Feature-Oriented Reuse Method) — расширение метода FODA[8]. В рамках метода рассматривается 4х-уровневая модель характеристик (feature model);

6) ODE (Ontology-based Domain Engineering) [9]. Метод соединяет онтологии и объектно-ориентированную технологию;
7) ODM (Organization-Domain Modeling) [10]. Данный подход рассматривает множество моделей, которые связаны какими-то семантическими отношениями. Эти отношения описываются некоторым математическим формализмом.

Метод FODA используется нами для генерации метамодели предметно-ориентированного языка по модели функциональностей (feature model) в конкретной предметной области в терминах этой области.

Эксперт в предметной области строит модель функциональностей, или характеристик, описывая тем самым эту предметную область. Затем выбирая определенные характеристики из модели функциональностей, генерируется метамодель в терминах предметной области. Эта метамодель используется для описания разных предметно-ориентированных языков. На основе этих визуальных языков уже создаются разные приложения в рамках этой предметной области. Таким образом, используя лишь одну модель предметной области, мы можем создавать разные программные продукты в рамках одной предметной области. А так как модель предметной области определена однозначно, никаких трудностей в понимании тех или иных терминов и связей здесь не возникнет. Поэтому перед экспертом становится непростая задача при построении модели описать предметную область как можно более точно и детально.

В дополнение к модели функциональностей предметной области, также можно рассмотреть модель сущность-связь (entity-relationship) этой области. В результате, можно получить наиболее полное описание предметной области.

3. Заключение

В документе был представлен обзор методов анализа предметной области, а также была рассмотрена методика получения метамодели предметно-ориентированного языка по модели предметной области.

Литература

1. *Ferre X., Vegas S*. An Evaluation of Domain Analysis Methods.
2. *Arango G*. Software Reusability. Ch. 2: Domain analysis methods. Pp. 17–49. Workshops M. E. Horwood. London, 1994.
3. *Mernik Marjan*. When and How to Develop Domain-Specific Languages // ACM Computing Surveys. Vol. 37. No. 4. December 2005. Pp. 316–344.
4. *Frakes W., Prieto-Diaz R.,* and *Fox C*. 1998. DARE: Domain analysis and reuse environment. Annals of Software Engineering 5, 125–141.
5. *Taylor R. N., Tracz W.* and *Coglianese L*. 1995.Software development using domain-specific software architectures. ACM SIGSOFT Software Engineering Notes 20, 5. Pp. 27–37.

6. *Kang K. C., Cohen S. G., Hess J. A., Novak W. E.* and *Peterson A. S*. 1990. Feature-oriented domain analysis (FODA) feasibility study. Tech. rep. CMU/SEI-90-TR-21. Software Engineering Institute, Carnegie Mellon University.
7. *Weiss D*. and *Lay C. T. R*. 1999. Software Product Line Engineering. Addison-Wesley.
8. *Kyo C. Kang, Sajoong Kim, Jaejoon Lee1, Kijoo Kim, Gerard Jounghyun Kim, Euiseob Shin*. FORM: A Feature-Oriented Reuse Method with Domain-Specific Reference Architectures.
9. *Falbo R. A., Guizzardi G.* and *Duarte K. C*. 2002. An ontological approach to domain engineering. In Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering (SEKE»02). ACM. Pp. 351–358.
10. *Simos M*. and *Anthony J*. 1998. Weaving the model web: A multi-modeling approach to concepts and features in domain engineering. In Proceedings of the 5th International Conference on Software Reuse. IEEE Computer Society. Pp. 94–102.

АВТОМАТИЧЕСКАЯ МИГРАЦИЯ МОДЕЛЕЙ

Т. Ю. Агапова

математико-механический факультет СПбГУ, 344 группа

E-mail: tatjana.agapova@gmail.com

Аннотация. При изменении языка моделирования модели, созданные с его помощью, могут перестать соответствовать ему. В таком случае некорректную модель необходимо перенести на новую версию языка, причём по возможности с сохранением семантики. Этот процесс называется миграцией модели. Данная статья описывает различные подходы к этой задаче и рассматривает основные сложности, препятствующие её автоматическому решению. Также приведён пример гибридной реализации, сочетающей преимущества рассмотренных подходов.

Введение

В настоящее время набирает популярность модельно-ориентированный подход к разработке программного обеспечения [1]. В его основе лежит идея представления программы в виде набора моделей, описанных с помощью некоторых, чаще всего визуальных, языков моделирования. При этом использование языков, предназначенных для узкой предметной области, упрощает процесс моделирования и восприятие моделей человеком [3]. Таким образом, желательно наличие инструмента, который позволяет быстрое создание визуальных предметно-ориентированных языков и моделей на этих языках. Такие средства называются предметно-ориентированными платформами (DSM-платформами). Создаваемый в них язык, как правило, также описывается в виде модели на специализированном языке метамоделирования — так называемой метамодели этого языка.

Со временем, язык моделирования может изменяться, например, с целью исправления ошибок в первоначальном проектировании либо ввиду уточнения используемой модели предметной области. В результате изменения метамодели зависящие от неё модели могут стать некорректными в новой версии языка (к примеру, если в моделях есть экземпляры типов, исключённых из языка). Таким образом, возникает задача миграции моделей — такой их модификации, которая восстанавливает их соответствие метамодели.

Перенос моделей на новую версию языка вручную — трудоёмкий процесс, при котором велика вероятность ошибки. Это ведёт к необходимости какого-либо способа спецификации миграционной стратегии в виде трансформаций [6], которые можно автоматически выполнить над моделями.

Похожая задача возникает в базах данных, когда данные необходимо перенести на изменившуюся схему базы данных [4]. Кроме того, обеспечение возможности работы с устаревшими моделями подобно обратной совмести-

мости приложений, при которой со старыми данными можно работать в новой версии приложения.

В настоящее время существуют различные подходы к миграции моделей — от полностью ручных до автоматизированных. Данная статья рассматривает существующие подходы к этой проблеме и основные препятствия на пути к автоматической миграции моделей. Также предлагается некоторый гибридный подход, позволяющий разработчику метамодели выбирать степень автоматизированности (а следовательно, и точности) миграции.

Классификация существующих подходов к миграции

Существующие подходы к миграции моделей можно разделить на три категории: ручная спецификация миграции, операторный подход и сопоставление моделей [5].

В случае ручной спецификации разработчик метамодели задаёт трансформации моделей вручную. Трансформации могут быть описаны на языке программирования общего назначения либо с помощью текстовых или визуальных языков описания трансформаций. Преимущество данного подхода — в том, что разработчик языка имеет более полный контроль над процессом миграции, а значит, семантика модели максимально сохраняется. Кроме того, процесс миграции абсолютно прозрачен для пользователя языка. Недостаток ручной спецификации — в трудоёмкости и необходимости учить язык спецификации миграционной стратегии разработчику метамодели.

При операторном подходе эволюция метамодели выражается в терминах композиции применённых к ней операторов. Каждому из них соответствует трансформация, применяемая к модели на данном языке. Последовательность трансформаций, соответствующих операторам, с помощью которых была произведена эволюция метамодели, представляет собой миграционную стратегию. При использовании данного разработчик DSM-платформы должен предоставить библиотеку операторов. В зависимости от реализации операторы могут выбираться разработчиком языка при редактировании метамодели или выводиться из истории её изменений автоматически. Точность миграции зависит от полноты предоставляемой библиотеки операторов. Она может быть слишком бедной, в этом случае в ней просто отсутствуют необходимые операторы. Излишняя же избыточность библиотеки приводит к неоднозначности выбора того или иного оператора в каждом конкретном случае, что затрудняет, в зависимости от реализации, спецификацию миграции разработчиком метамодели или автоматический вывод миграционной стратегии.

Сопоставление моделей — автоматический подход, в идеале не требующий никакого вмешательства разработчика либо пользователя метамодели в процесс миграции. Подходы на основе сопоставления моделей используют один из двух механизмов: историю изменений метамодели либо модель разницы старой и новой метамоделей. В случае использования истории из-

менений все действия, совершённые над метамоделью в процессе её редактирования, записываются и сохраняются вместе с метамоделью. Впоследствии, при миграции эти записи анализируются для вывода миграционной стратегии. Модель разницы является представлением различий старой и новой метамоделей и отвечает на следующие вопросы: какие типы добавились, исчезли или заменились на другие, как изменились иерархии наследования, отношения вложенности элементов и т. д. Эта информация используется для вывода трансформаций мигрируемой модели. Лог изменений и модель разницы — взаимодополняющие механизмы, при совместном использовании позволяющие значительно повысить эффективность миграции. Модель разницы даёт компактное представление важных для миграции изменений метамодели, с которым удобнее работать, нежели с последовательностями изменений, многие из которых промежуточные и не отражаются на результирующей разнице версий метамодели. С другой стороны, при замене типа посредством удаления одного типа с последующим созданием другого в модели разницы эти типы будут никаким образом не связаны, но анализ лога может выявить типичную последовательность и вывести трансформацию корректно. Таким образом, для достижения наилучшей эффективности миграции эти два механизма должны использоваться вместе.

Сложности автоматической миграции

Для того, чтобы определить основные сложности, препятствующие эффективной реализации автоматической миграции моделей, рассмотрим изменения, которые могут происходить с метамоделью языка, и их влияние на зависящие от неё модели. В [2] предлагается следующая классификация изменений.

1. Не нарушающие целостность моделей. К этой категории относятся аддитивные изменения, такие как добавление к языку новых типов или их свойств. Такие изменения не затрагивают существующие модели, а следовательно, никак не влияют на процесс миграции.
2. Нарушающие целостность моделей, но разрешимые автоматически. К таким изменениям относятся, например переименования типов языка. Если их отслеживать в процессе редактирования метамодели, то автоматическая миграция осуществляется с помощью замены экземпляров старого типа на экземпляры нового.
3. Нарушающие целостность моделей и неразрешимые автоматически. Некоторые примеры изменений из этой категории рассмотрены далее в данной статье.

Таким образом, для решения задачи автоматической миграции моделей необходимо исследовать третью категорию на предмет изменений, которые на самом деле могут быть произведены автоматически.

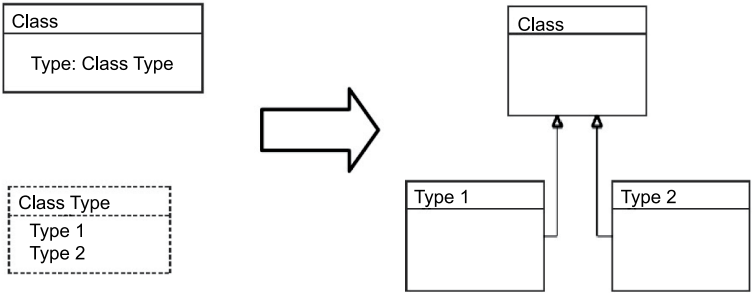


Рис. 1. Изменение метамодели при разделении типа на подтипы

В качестве примера рассмотрим разделение какого-либо типа элементов на подтипы. Бывший ранее конкретным тип становится абстрактным, и его экземпляры должны быть заменены на экземпляры некоторого конкретного подтипа. В общем случае правила такой замены нельзя вывести из метамодели языка, и миграция должна быть специфицирована человеком. Однако в частном случае, если разделение на подтипы представляет собой рефакторинг замены поля типа наследованием, выбор подтипа можно осуществить автоматически. Сложность может представлять лишь установление соответствия между значениями старого поля типа и новыми подтипами, для чего может потребоваться анализ их имён (рис. 1).

Рассмотрим ещё один пример. Допустим, имеется язык, который позволяет создавать контейнеры элементов (рис. 2).

В случае, если разработчик этого языка примет решение добавить ещё один уровень вложенности, требуя группировать элементы внутри контейнеров, необходимо каким-то образом осуществить такую группировку в уже существующих моделях (рис. 3).

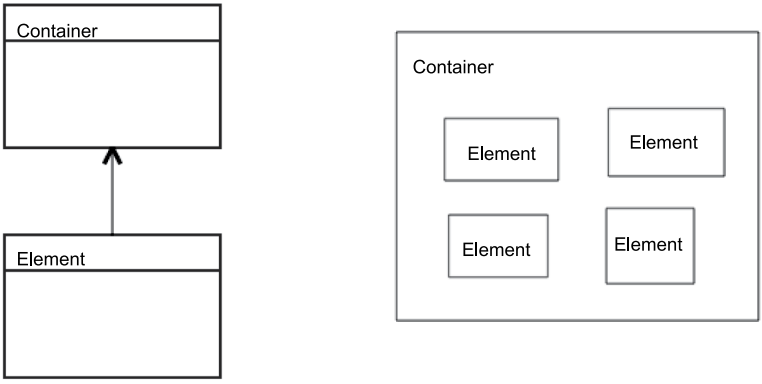


Рис. 2. Метамодель языка контейнеров и пример модели на этом языке

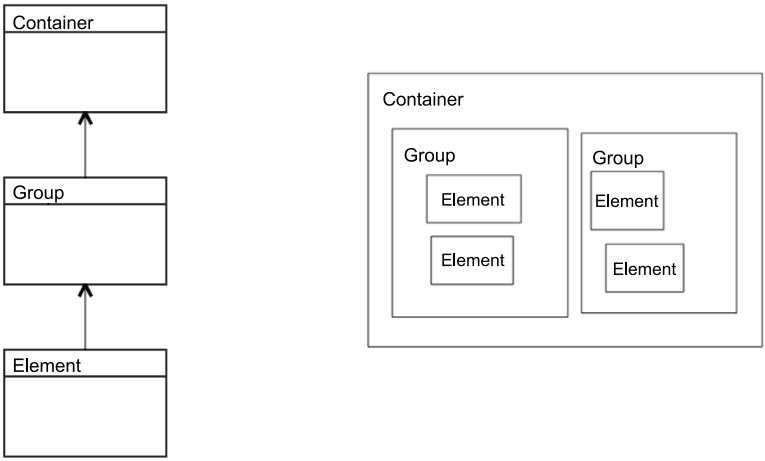


Рис. 3. Преобразованный язык контейнеров и возможное преобразование модели

Если допускать, что группировка может осуществляться по произвольному признаку, то определить автоматически наиболее подходящий в каждом случае способ группировки не представляется возможным. Более того, это не под силам даже разработчику языка, так как в общем случае для этого требуется знание смысла, который вкладывает пользователь в такую группировку.

Таким образом, в данном случае невозможно произвести автоматическую миграцию с сохранением семантики модели. Всё, что можно сделать, — это выбрать некоторый вырожденный вариант группировки (допустим, считать, что все элементы принадлежат одной группе) и предупредить пользователя о возможной потере смысла.

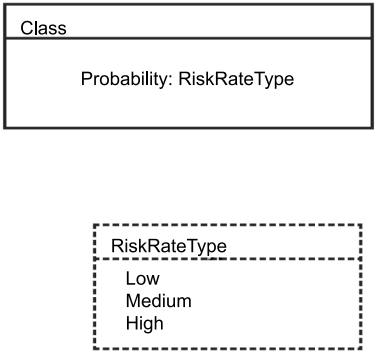


Рис. 4. Тип со свойством перечисляемого типа

Последний пример также иллюстрирует необходимость вмешательства пользователя в процесс миграции. Пусть в некотором типе было поле, определяющее вероятность некоторого события как «низкую», «среднюю» или «высокую» (рис. 4).

Такая точность определения вероятности не удовлетворяла некоторых пользователей, в связи с чем решено было сделать это поле целочисленным со значениями от единицы до ста, чтобы оно означало вероятность в процентах (рис. 5).

В данном случае также едва ли можно определить какую-либо разумную стратегию миграции, так как понимание того, что такое «низкая» или «высокая» вероятность, может зависеть от контекста использования.

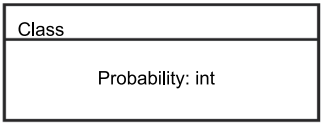


Рис. 5. Свойство перечисляемого типа заменено на целочисленное

Рассмотренные примеры выявляют основную сложность задачи автоматической миграции моделей, а именно, зависимость процесса миграции от восприятия языка моделирования и моделей на нём человеком: разработчиком или пользователем языка. Также установлено, что в некоторых случаях даже разработчик языка не в силах специфицировать миграционную стратегию, позволяющую перенести модели на новую версию с сохранением семантики, что требует вмешательства пользователя не только при автоматической миграции, но и в случае спецификации стратегии разработчиком языка.

Помимо данной проблемы, автоматическую миграцию затрудняет сложность большинства эволюционных изменений метамодели. Они могут состоять из нескольких элементарных изменений (в случае разделения на подтипы — удаление свойства типа, создание подтипов и связей с базовым типом), а также затрагивать несколько элементов (в примере группировки в контейнерах всего одно эволюционное изменение затрагивает всю метамодель). Следовательно, нужно иметь возможность отличить последовательность не связанных друг с другом изменений от одного сложного изменения.

Описанные сложности делают задачу автоматической миграции моделей в общем случае неразрешимой, в связи с чем желательной представляется возможность оптимизировать процесс миграции с помощью ручной спецификации сложных изменений разработчиком языка и оповещения его пользователя о спорных ситуациях. Пример подхода к миграции моделей, осуществляющего такую оптимизацию, рассмотрен ниже.

Реализация в QReal

QReal — это DSM-платформа, разрабатываемая на кафедре системного программирования Санкт-Петербургского государственного университета [9]. Данная система накладывает некоторые специфические требования к процессу миграции ввиду необходимости поддержки режимов интерпретации и метамоделирования «на лету» [8]. Цель этих режимов — ускорение цикла разработки-использования языка, в связи с чем миграция должна проходить максимально прозрачно для пользователей.

В основе подхода к миграции моделей, выбранного для реализации в QReal, лежит автоматический подход с использованием лога изменений и модели разницы. Улучшить точность миграции при этом возможно с помощью спецификации миграционных изменений, основанной на суще-

ствующем в QReal механизме рефакторингов [7]. Поскольку рефакторинг представляет собой сложное, и при этом цельное, изменение метамодели, определение разработчиком языка миграции, соответствующей ему, исключит необходимость её автоматического вывода.

Таким образом, при использовании данного подхода есть возможность выбирать степень автоматизации миграции, увеличивая точность или снижая трудозатраты на спецификацию миграционной стратегии, что способствует использованию режимов интерпретации и метамоделирования «на лету».

Заключение

В данной статье была рассмотрена задача миграции моделей на новую версию языка моделирования и основные подходы к её решению. Трудоёмкость подходов, требующих участия разработчика языка в выводе миграционной стратегии, ведёт к необходимости разработки методов автоматической миграции.

К сожалению, как было показано выше, не для всех эволюционных изменений языка возможно автоматически вывести соответствующие изменения моделей. В некоторых случаях единственный способ произвести миграцию с сохранением семантики модели заключается в вовлечении разработчика или пользователя языка в процесс миграциии. При этом возможны гибридные решения задачи миграции, дополняющие автоматические подходы возможностью вручную определить миграционную стратегию. Пример такого решения также приведён в статье.

Для достижения наибольшей эффективности автоматических или автоматизированных подходов к миграции требуется как можно более интеллектуальный анализ истории изменения метамодели и модели разницы, который может включать в себя, к примеру, анализ имён сущностей языка, как было показано в примере с разделением на подтипы. Также необходимо распознавать сложные изменения, что может быть частично реализовано с помощью определения наиболее типичных последовательностей изменений в логе и шаблонов в модели разницы. Разработку методов анализа артефактов, используемых при выводе трансформаций, можно считать основным направлением исследований в области автоматической миграции моделей.

Литература

1. *Brambilla M., Cabot J., Wimmer M.* Model-Driven Software Engineering in Practice // Morgan & Claypool. 2012. 182 p.
2. *Gruschko B., Kolovos D., Paige R.* Towards synchronizing models with evolving metamodels // International Workshop on Model-Driven Software Evolution, MoDSE. 2007.

3. *Kelly S., Tolvanen J*. Domain-Specific Modeling: Enabling Full Code Generation // Wiley-IEEE Computer Society Press. 2008. 448 p.
4. *Rahm E., Bernstein P. A*. A Survey of Approaches to Automatic Schema Matching // The VLDB Journal. Springer-Verlag, 2001.
5. *Rose L. M., Paige R. F., Kolovos D. S., Polack F. A. C*. An Analysis of Approaches to Model Migration // Joint MoDSE-MCCM 2009 Workshop — Models and Evolution, 2009. Pp. 6–15.
6. *Rozenberg G*. Handbook of Graph Grammars and Computing by Graph Transformation. Volume 1: Foundations. World Scientific, 1997.
7. *Кузенкова А. С., Литвинов Ю. В*. Поддержка механизма рефакторингов в DSM-платформе QReal // Материалы межвузовского конкурса-конференции студентов, аспирантов и молодых ученых Северо-Запада «Технологии Microsoft в теории и практике программирования». СПб.: Изд-во СПбГПУ, 2013. С. 71–72.
8. *Птахина А. И*. Разработка метамоделирования «на лету» в системе QReal // Список-2013: Материалы всероссийской научной конференции по проблемам информатики. 2013 г., Санкт-Петербург. СПб.: ВВМ, 2012. С. 28–36.
9. *Терехов А. Н., Брыксин Т. А., Литвинов Ю. В*. QReal: платформа визуального предметно-ориентированного моделирования // Программная инженерия. 2013. №6. С. 11–19.

ФУНКЦИОНАЛЬНОЕ РЕАКТИВНОЕ ПРОГРАММИРВОАНИЕ РОБОТОВ НА БАЗЕ ПЛАТФОРМЫ ТРИК

А. Ю. Кирсанов

студент 3-го курса Математико-механического факультета СПбГУ

E-mail: kashmervil@gmail.com

Аннотация. Роботы и встроенные системы всё плотнее входят в нашу жизнь, малогабаритные микроконтроллеры можно найти почти в каждом электронном приборе. Стремительное развитие роботов, увеличение мощностей и уменьшение размеров плат позволяет расширять круг допустимых задач и использовать всё более смелые идеи для их решения. В данной статье на примере реализации библиотеки на F# описан опыт применения технологий, не предназначенных для робототехники, использование которых стало возможным благодаря развитию микроконтроллеров.

Введение

Современные смартфоны и специализированные платы отстают по скоростям и объёмам памяти от больших настольных компьютеров всего на пару лет. Это позволяет переходить от классического для микроконтроллеров «С» к языкам с более богатыми средствами выразительности и современным технологиям. Их применение накладывает определённые издержки на скорость, и использование которых было недопустимо в микроконтроллерах с маломощными процессорами.

Кроме использования новых платформ и решений для встроенных систем, которые позволяют расширить круг решаемых задач и упростить разработку, становится возможным применять решения и технологии, разработанные для персональных компьютеров. Это позволяет ускорить процесс разработки за счёт увеличения уровня абстракции, с которой приходится работать программисту. Кроме того, использование уже существующих технологий имеет ещё одно важное преимущество: программистам не придётся учиться работать на роботе, если они уже знакомы с теми или иными средствами.

Современная робототехника

Робототехника в наши дни является точкой соединения целой группы самостоятельных научных областей таких как:

1. Искусственный интеллект.
2. Теория управления.
3. Компьютерное зрение.

Алгоритмы, которые робот использует во время своей работы, неразделимо связаны с большим количеством вычислений. А сложность и объёмы программ для современных роботов требуют правильной организации проекта для создания расширяемых, повторно используемых программ. Таким образом переход к современным технологиям в области разработки ПО давно востребованы в мире микроконтроллеров.

Реактивное программирование

Реактивное программирование — одна из парадигм программирования, основным понятием которого является событие — некоторое изменение произошедшее с объектом или объектами. Вся логика программы заключается в правильной манипуляции потоками этих событий.

Роботы по своей сути являются очень реактивными системами, в том плане, что поведение робота очень просто описывается в реактивной схеме . В любой момент времени можно представить робота, как объект, принимающий показания датчиков и которому необходимо быстро реагировать на эти события в виде определённых действий. Например стабилизация или движение в обход препятствию, которое обнаружили сенсоры. В данной абстракции датчики являются генераторами потоков событий.

Используемые программные средства

- .NET
- F#
- Reactive Extension for .NET
- Mono

Фреймворк .NET, изначально разрабатываемый для операционной системы Windows, пережил уникальный в своём роде опыт. С появлением Mono, использование .NET стало возможным на Unix-like системах (Linux, MacOS, Androind). Открытие исходных кодов многих продуктов Microsoft, приближающийся выход C# 5.0 и его стандартизация в в двух вариантах для разных ОС, говорит о только большем внедрении .NET в современную разработку программного обеспечения. Поэтому использование .NET на встроенных системах осмысленно и обещает быть логичным шагом в развитии программных средств, используемых на микроконтроллерах.

Роботы проекта Трик используют специальный дистрибутив линукса. Кроме всего прочего на роботах установлена полноценная версия последнего Mono, что позволяет запускать исполняемые файлы для виртуальной машины .NET.

Библиотека полностью разрабатывается на языке F#.

F# — мультипарадигменный язык программирования, берущий свои корни в ML-семействе. По своему устройству язык очень похож на OCaml реализованный поверх .NET.

События в .NET — first class value, мы можем передавать события аргументами в функции. Существует модуль Events, в котором реализованы базовые функции для работы с событиями единообразно спискам и другим контейнерам.

Тем не менее использование событий и Observable последовательностей кардинально отличается моделью предоставления данных. Списки, массивы, всевозможные очереди и стеки реализуют интерфейс INumerable<T>, который использует модель вытягивания данных (Pulling data). Для обращения к определённому элементу необходимо явно указать, что мы хотим его получить.

Observable последовательности реализуют IObservable<T>. При использовании Observable, мы следуем модели выталкивания данных (Pushing Data). Вы один раз указываете, что вы хотите получать данные от этого источника, производя при этом некие формальные действия, все изменения происходящие с источником будут поступать к вам и должным образом обрабатываться. Сам интерфейс содержит ровно один метод Subscribe, после вызова которого вы начнёте получать изменения.

Кроме того, использование Observable решает ещё одну проблему, связанную уже с Event»ами, а не с роботами. Обычные F# события не безопасны в плане освобождения памяти. Если есть последовательность событий, которая модифицируется (например с помощью Event.Map), то память, которую занимала исходная последовательность, может не быть освобождена.

Не блокирующие чтения

Датчики представлены в операционной системе, как специальные файлы. Все сенсоры проекта Трик представлены в виде двух разных типов:

- Polling sensors.
- FILO sensors.

Polling сенсоры посылают данные несколько сотен раз в секунду. Обработка такого количества данных поглощала бы большую часть процессорного времени. Стремление получать и обрабатывать самые свежие данных в их полном объёме привело бы к увеличению времени отклика и зависанию. Более того, многие аналоговые датчики обладают характерными шумами, что ещё сильнее усложняет правильное управление.

Ситуацию спасает то, что обычным роботам просто не нужно так быстро менять своё поведение, и реакция на изменение в датчике с задержкой в 50 ms вполне допустима.

Как следует фильтровать данные? Активное ожидание в цикле не будет является лучшим решением для встроенной системы, ресурсы которой, хоть и позволяют использовать нам все преимущества высокоуровневой разработки, не настолько велики, чтобы пренебрегать очевидными оптимизациями. Кроме того, синхронизация получения результатов от разных датчиков ста-

вится проблемой, и с увеличением программы следить за синхронизацией становится невозможно. Представление датчиков, как генераторов Observable последовательностей, позволяет использовать готовые механизмы для работы с этими последовательностями.

Другой тип датчиков действует как некой буфер, в который периодически что-то поступает. Время, между которым гарантируется появление данных, не зависит от программиста. Чтение данных из сенсора происходит совершенно прозрачно для программиста и файловой системы. Тем не менее, данные могут так и не поступить в этот буфер в приемлемый срок. Всё это может затянуть чтение файла на неопределённое время. В месте с чтением ждать будет и программа, что никак не допустимо при разработке time critical программного обеспечения. На время зависания робот будет не в состоянии контролировать своё поведение. Поэтому чтение таких сенсоров нужно переносить в другие ветки. Observable последовательности решают эту проблему, так как используют асинхронные операции из .NET, распределяющиеся по пулу потоков среды исполнения.

Результаты

Реализована библиотека для реактивного программирования роботов, поддерживается работа со всеми доступными устройствами(силовые и сервомоторы, кнопки, инфракрасные датчики расстояний, гироскопы, акселерометры).

Основой упор сделан на использовании библиотеки с Rx, но реализован и базовый интерфейс для работы с датчиками и моторами, который можно использовать как при выводе промежуточных результатов для отладки и тестирования, так и при разработке приложений без использования реактивной парадигмы.

Заключение

Одной из задач проекта Трик является обучение программированию с помощью робототехники. Программирование роботов на языках высокого уровня с использованием современных технологий:

- Замыканиями
- Событиями
- Асинхронным и многопоточным
- Реактивным и функциональным программированием

Позволяет получать сразу несколько плюсов:

1. Возможность программирования роботов ясным декларативным способом без необходимости низкоуровневой разработки.
2. Отличная возможность познакомиться с новыми технологиями, парадигмами с помощью такого наглядного способа как роботы.

Литература

1. Programming Reactive Extensions and LINQ ISBN-10: 1430237473
2. Charles University in Prague, Faculty of Mathematics and Physics, MASTER THESIS, Tomáš Petříček, Reactive Programming with Events http://tomasp.net/academic/theses/events/events.pdf
3. Expert F# 3.0 Don Syme, Adam Granicz, Antonio Cisternino ISBN-10: 1430246502.

АППАРАТНАЯ ВОЗМОЖНОСТЬ ОРГАНИЗАЦИИ ДЕЦЕНТРАЛИЗОВАННОЙ СЕТИ МОБИЛЬНЫХ ДАТЧИКОВ[1]

А. Ю. Коровянский

студент 3-го курса кафедры системного программирования СПбГУ

E-mail: alexey_kor@hotmail.com

К. С. Амелин

к. ф.-м. н., стажер-исследователь кафедры системного программирования СПБГУ

E-mail: konstantinamelin@gmail.com

Аннотация. При организации коллективной работы различных мобильных датчиков возникает необходимость коммуникаций их друг с другом. Как правило, такая связь устанавливается не напрямую датчик — датчик, а через центр сбора информации. Такая централизованная схема имеет ряд минусов при организации автономного и адаптивного группового взаимодействия. Для организации такого взаимодействия все чаще стали применять мультиагентные системы, одной из функциональности которых входит работа в децентрализованных сетях, сетях без единого центра.

Работа посвящена реализации аппаратной возможности организации децентрализованной сети мобильных датчиков на примере Android Mini PC с использованием Xbee модемов.

Введение

При организации группового взаимодействия в крупных системах для преодоления сложностей коммутации элементов системы, планирования их работы и организации автономного взаимодействия все чаще применяют идеологию мультиагентных систем (МАС). В таких системах нет жестко заданных связей между элементами и каждый элемент — агент — обладает определенной самостоятельностью и способен образовывать связи с другими агентами в процессе решения задач по мере необходимости. Характерными особенностями агентов являются [1]:

- коллегиальность, т. е. способность к коллективному целенаправленному поведению в интересах решения общей задачи;
- автономность, т. е. способность самостоятельно решать локальные задачи;
- активность, т. е. способность к активным действиям ради достижения общих и локальных целей;

[1] Работа частично поддержана грантом РФФИ №13–07–00250.

- информационная и двигательная мобильность, т. е. способность активно перемещаться и целенаправленно искать и находить информацию, энергию и объекты, необходимые для кооперативного решения общей задачи;
- адаптивность, т. е. способность автоматически приспосабливаться к неопределённым условиям в динамической среде;

Эти возможности кардинально отличают мультиагентные системы (МАС) от существующих «жестко» организованных систем.

Важной задачей, возникающей при использовании МАС на практике, является распределение ресурсов между агентами. Если связи между агентами постоянны, передача информации из одной точки сети в другую не является сложной задачей. В ситуации постоянного изменения связей между агентами передача информации из одной точки сети в другую может быть сведена к задаче достижения консенсуса или согласования характеристик [2].

Одним из направлений применения МАС являются системы управления мобильными роботами, а в частности беспилотными летательными аппаратами. В качестве примера, можно рассмотреть систему управления группой БПЛА, которая описана в статье [3]. Каждому самолету ставится задача анализа определенного квадранта территории. По умолчанию каждый самолет пролетает свой квадрант, возвращается к базовому узлу и передает всю информацию ему. При данной организации выявляются несколько недостатков:

- медленная передача необходимой информации на базовый узел;
- статическая организация маршрутов полета, то есть маршруты устанавливаются при запуске самолетов и остаются неизменными на всем протяжении полета;
- неполадки в работе БПЛА приведут к потере данных и, возможно, самих устройств, при отсутствии специализированных средств для решения данных проблем/

При реализации возможности взаимодействия БПЛА данные недостатки исчезают:

- возможность передачи информации через других агентов по цепочке, таким образом, не дожидаясь прилета каждого самолета к базовому узлу;
- при необходимости можно изменить маршрут полета, например, поменять местами целевые квадранты между БПЛА;
- при поломке устройства, можно подать сигнал другим БПЛА для дополнительного анализа потерянного квадранта и обнаружения вышедшего из строя самолета.

Аппаратная возможность организации децентрализованной сети

В статье [3] описан принцип построения трехуровневой системы управления легким БПЛА, основной идеей такой архитектуры является добавление

дополнительного микрокомпьютера, за счет которого и появляется возможность применения мультиагентных технологий. Таким образом, опишем аппаратную возможность организации сети между этими микрокомпьютерами.

Описание устройств

В качестве датчиков можно использовать Android Mini PC, в частности MK808 или Imito MX 1/2. Оба устройства используют Rockchip RK3066. В RK3066 интегрировано два ядра ARM Cortex-A9. Также на обоих устройствах по умолчанию установлена ОС Android 4.1 [4].

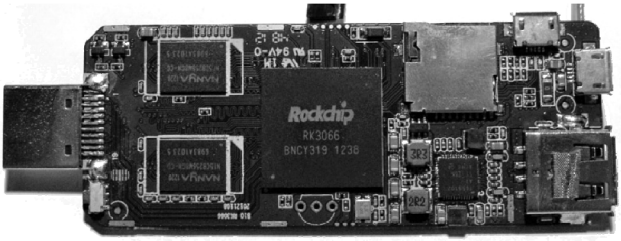


Рис. 1. Imito MX2

Для обеспечения связи можно использовать встроенные в устройства Wi-Fi модули, но в таком случае радиус передачи будет достаточно невелик, поэтому лучше использовать специализированные модули связи, например, XBee модемы [5].

XBee и XBee-PRO модули были разработаны для работы со стандартом IEEE 802.15.4 и поддержки дешевых, отличающихся низким энергопотреблением, беспроводных сетей. Радиус передачи на открытой местности достигает 1600 метров. Модули работают на частоте 2.4 ГГц. Скорость передачи составляет 250 бит в секунду. Данные устройства используют UART интерфейс на базе RS-232.

Рис. 2. Xbee PRO модуль

Подключение модулей связи

Для подключения модулей связи можно использовать FTDI переходники на базе чипа FT232RL. К сожалению, в ОС Android нет драйверов для данных устройств.

Данную проблему можно решить двумя способами:

- Использование USB Host API, официально поддерживаемое операционной системой Android с версии Android 3.1;
- Установка дистрибутива Linux для ARM устройств на Android Mini PC;

Первый способ позволяет работать с любыми внешними устройствами, подключенными к Android Mini PC, но, несмотря на официальную поддержку данного функционала, путем тестирования было обнаружено, что работа API корректна на очень малом количестве устройств, например, HTC One или различных моделях Nexus. На специфических же устройствах данный функционал не работает.

Также, поскольку ОС Android использует Linux ядро, то можно собрать драйвера системы Linux для используемых устройств, но для этого необходим исходный код ядра, доступа к которому нет, опять же из-за специфичности устройств.

Также можно использовать уже собранные драйвера для похожих ARM устройств, но в данном случае добиться их корректной работы не удалось.

Таким образом, предлагается использовать второй способ, то есть установку дистрибутива Linux на используемое устройство.

Установка Linux дистрибутива на ARM Android Mini PC

Большая часть данных ARM устройств базируются на различных моделях Rockhip (RK3066, RK3188). На всех устройствах такого типа установка проводится в два этапа:

- Установка Linux ядра, обычно в область Recovery устройства, для того, чтобы иметь возможность загрузиться в Linux;
- Копирование Linux Root File System (RFS) на карту памяти устройства для того, чтобы ОС могла загрузиться оттуда.

Для установки Linux ядра существуют специализированные инструменты, как официальные от компании Rockchip (RKFlashTool/RKFlashKit), так и модификации, сделанные на основе официальных (Finless FlashTool). Они предназначены для обновления данных устройств, но могут использоваться и для установки стороннего ядра в Recovery область.

К сожалению, информации, касающейся второго шага очень мало, поэтому стало необходимо написание инструкции по его выполнению. Задача осложняется тем, что большинство дистрибутивов доступны в достаточной мере лишь для x86 ЦП, а не ARM процессоров. И даже при нахождении

скомпилированных под ARM дистрибутивов необходимо провести дополнительную настройку, например, скопировать на них необходимые модули ядра.

Таким образом, выбор ограничен лишь несколькими статическими RFS, созданными специально для данной задачи (Picuntu). Эти RFS обычно уже содержат популярные дополнительные модули ядра, но становится ясно, что используя их появляется зависимость от разработчика, а также ограниченность в дополнительных возможностях, например, установке своих модулей ядра.

Существует проект Linaro занимающийся программным обеспечением для платформ ARM. Найти дистрибутив уже скомпилированный под ARM устройства можно в этом проекте.

Далее необходимо записать RFS на карту памяти устройства. После подключения карты к ПК с Linux, необходимо понять какое имя было ей дано. Это можно сделать при помощи команды:

```
parted –l
```

```
Model: Sony Card R/W -SD/MS (scsi)
Disk /dev/sdg: 7948MB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Number  Start   End     Size    Type     File system  Flags
1       32.3kB  7946MB  7946MB  primary  ext4
```

Рис. 3. Результат команды «parted –l»

Например, на рисунке 3 видно, что карте было дано имя /dev/sdg.

Для того чтобы задать разбиение карты, необходимо выполнить следующее:

```
umount {Имя устройства}
mkfs ext4 –F –L linuxroot {Имя устройства}
```

Далее необходимо снова монтировать карту и перенести на нее образ RFS. Перенести образ RFS можно, например, так:

```
tar xvfz {адрес образа}
```

Выполнять команду необходимо из каталога, в который монтировалась карта.

Поскольку в Linaro архив содержит RFS внутри каталога «binary», то необходимо перенести все его содержимое в корень карты и удалить пустой каталог «binary». Это можно сделать, например, так:

```
mv binary/*
rmdir binary
```

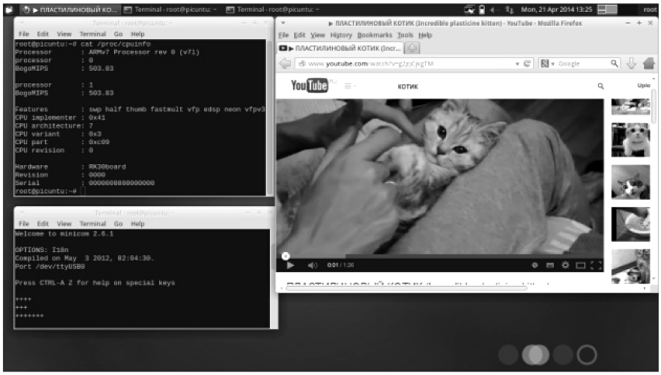


Рис. 4. Скриншот работы Linux на Imito MX2

Далее можно распаковать все необходимые модули на карту, в итоге добившись необходимой функциональности.

Заключение

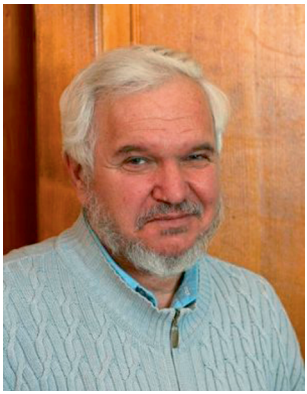
В статье были описаны подходы к реализации аппаратной возможности организации децентрализованной сети мобильных датчиков. Детально был описан процесс установки дистрибутива Linux на ARM Android Mini PC.

В результате было получено устройство, предоставляющее всю необходимую функциональность для подключения модулей связи и, следовательно, позволяющее организовать децентрализованную сеть.

Литература

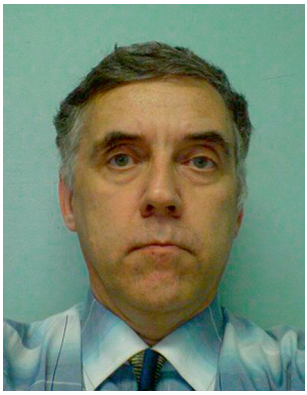
1. *Городецкий В. И., Грушинский М. С., Хабалов А. В*. Многоагентные системы (обзор) // Новости искусственного интеллекта. 1998. №2. С. 64–116.
2. *Amelina N., Fradkov A.* and *Amelin K*. Approximate Consensus in Multi-agent Stochastic Systems with Switched Topology and Noise // Proc. of MSC IEEE 2012, October 3–5, 2012. Dubrovnik, Croatia. Pp. 445–450.
3. *Амелин К. С., Граничин О. Н*. Мультиагентное сетевое управление группой легких БПЛА // Нейрокомпьютеры: разработка, применение. №6. 2011. С. 64–72.
4. Ресурс сайта производителя Mini PC: http://www.imito.net/
5. Ресурс сайта производителя Xbee: http://www.digi.com/

Фундаментальная информатика



Косовский
Николай Кириллович

д.ф.-м.н., профессор
заведующий кафедрой информатики СПбГУ



Герасимов
Михаил Александрович

к.ф.-м.н., доцент кафедры информатики СПбГУ

ПОНЯТИЕ НЕПОЛНОЙ ВЫВОДИМОСТИ ПРЕДИКАТНОЙ ФОРМУЛЫ И ЕГО ПРИМЕНЕИЯ К РЕШЕНИЮ ЗАДАЧ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА[1]

Т. М. Косовская

профессор кафедры информатики СПбГУ

E-mail: kosovtm@gmail.com

Аннотация. Работа содержит обзор различных применений ранее введённого автором понятия неполной выводимости предикатной формулы к решению различных задач искусственного интеллекта в рамках логико-предметного подхода.

Среди таких задач рассматриваются: задачи с неполной информацией об объекте; введение метрики в множестве элементарных конъюнкций атомарных формул, позволяющей установить степень схожести двух объектов, заданных описанием свойств составляющих его элементов и отношений между ними; построение иерархического описания классов объектов, учитывающего общие характеристики его объектов и существенно понижающее число шагов решения задачи.

Введение

Многие задачи искусственного интеллекта (ИИ) допускают формализацию средствами исчисления предикатов при условии, что объект рассматривается как совокупность своих элементов [4]. Такой подход назван логико-предметным в отличие от логико-алгебраического, при котором весь объект целиком описывается набором значений выбранных признаков.

Ниже описываются применения понятия неполной выводимости предикатной формулы, введённое в [2] для решения задач распознавания образов с неполной информацией об объекте при логико-предметном подходе.

Это понятие оказалось полезным для введения метрики в множестве всех элементарных конъюнкций атомарных формул [5]. Такая метрика позволяет вычислять степень похожести объектов, описанных в рамках логико-предметного подхода.

Задачи, возникающие при логико-предметном подходе, являются NP-трудными [3]. Для уменьшения числа шагов их решения в [1] предложено создание уровневого описания классов, заключающееся в выделении «часто» встречающихся среди исходных описаний классов подформул «небольшой сложности». Термины «часто» и «небольшой сложности» уточняются в зависимости от алгоритма, с помощью которого решается поставленная задача.

Понятие неполной выводимости предикатной формулы предлагается использовать для выделения подформул с требуемыми свойствами.

[1] Работа поддержана грантом РФФИ 14-08-01276.

Постановка задач ИИ при логико-предметном подходе

Пусть исследуемый объект представлен как множество своих элементов $\omega = \{\omega_1, ..., \omega_t\}$. На ω задан набор предикатов $p_1, ..., p_n$, характеризующих свойства элементов ω и отношения между ними. Логическим описанием $S(\omega)$ объекта ω называется множество всех атомарных формул или их отрицаний, истинных на ω. Множество всех объектов разбито на классы $\Omega = \cup_{k=1}^{K} \Omega_k$. Логическим описанием класса называется формула $A_k(\boldsymbol{x})$, заданная в виде дизъюнкции элементарных конъюнкций, такая что если $A_k(\boldsymbol{\omega})$ истинна, то $\omega \in \Omega_k$. (Здесь и далее обозначение $\boldsymbol{x}$ используется для списка, состоящего из элементов множества x.)

С помощью построенных описаний предлагается решать следующие задачи. **Задача идентификации:** *проверить, принадлежит ли объект* ω *или его часть классу* Ωk. **Задача классификации.** *Найти все такие номера классов k, что* $\omega \in \Omega k$. **Задача анализа сложного объекта.** *Найти и классифицировать все части* τ объекта ω, для которых $\tau \in \Omega k$.

Решение задач идентификации, классификации и анализа сложного объекта сведено к доказательству соответственно формул $S(\omega) \Rightarrow \exists \boldsymbol{x}_{\neq} A_k(x)$, $S(\omega) \Rightarrow \vee_{k=1}^{K} A_k(\boldsymbol{x})$, $S(\omega) \Rightarrow \vee_{k=1}^{K} \exists \boldsymbol{x}_{\neq} A_k(\boldsymbol{x})$. Решение всех этих задач основано на решении задачи $S(\omega) \Rightarrow \exists \boldsymbol{x}_{\neq} A(\boldsymbol{x})$, где $A(\boldsymbol{x})$ — элементарная конъюнкция.

В [3] доказаны оценки числа шагов алгоритмов, решающих сформулированные задачи. Доказана NP-трудность рассматриваемых задач.

Понятие неполной выводимости

Рассматривается задача проверки того, что из истинности всех формул множества $S(\omega)$ следует истинность $A(\boldsymbol{x})$ или некоторой её максимальной подформулы $A'(\boldsymbol{x}')$ на наборе различных констант из ω.

Пусть a и a' — количества атомарных формул в формулах $A(\boldsymbol{x})$ и $A'(\boldsymbol{x}')$, m и m' — количества переменных формул в формулах $A(\boldsymbol{x})$ и $A'(\boldsymbol{x}')$ соответственно. Параметры q и r определяются соответственно по формулам $q = a'/a$, $r = m'/m$. В этом случае формула $A'(\boldsymbol{x}')$ называется (q, r) — фрагментом формулы $A(\boldsymbol{x})$.

В [2] приведён один из возможных алгоритмов нахождения (q, r) — фрагмента с максимально возможным значением параметра q.

Решение задач с неполной информацией об объекте

Задача распознавания объекта в условиях неполной информации заключается в том, что задано неполное описание объекта $S(\omega)$, содержащее не все истинные на ω атомарные формулы или их отрицания, а лишь некоторое его подмножество $S'(\omega) \subset S(\omega)$.

При решении сформулированных задач при наличии неполного описания объекта вместо проверки справедливости $S(\omega) \Rightarrow \exists \boldsymbol{x}_{\neq} \mathrm{A}(\boldsymbol{x})$ имеется возможность проверки лишь того, что $\mathrm{S}'(\omega) \Rightarrow \exists \boldsymbol{x}'_{\neq} A'(\boldsymbol{x}')$ для некоторого максимального (q, r) — фрагмента $A'(\boldsymbol{x}')$ формулы $A(x)$.

Однако только нахождение такого максимального (q, r) — фрагмента не достаточно для того, чтобы с некоторой степенью уверенности утверждать, что элементы объекта ω удовлетворяют формуле $A(\boldsymbol{x})$. Введено понятие дополнения $D(A(\boldsymbol{x}))$ фрагмента $A'(\boldsymbol{x}')$ до формулы $A(\boldsymbol{x})$, определяемого как результат замены в конъюнктивных членах, входящих в $A(\boldsymbol{x})$, но не вошедших в $A'(\boldsymbol{x}')$, всех переменных из $\boldsymbol{x}'$ на их значения, определённые при доказательстве следствия $S'(\omega) \Rightarrow \exists \boldsymbol{x}'_{\neq} \mathrm{A}'(\boldsymbol{x}')$. Необходимо также потребовать, чтобы $S'(\omega)$ не противоречило формуле $A(\boldsymbol{x})$, т. е. $S'(\omega) \Rightarrow \neg \exists \boldsymbol{x}'_{\neq} D(A(\boldsymbol{x}))$. При этом со степенью уверенности q можно утверждать, что объект ω содержит r-ую часть объекта, удовлетворяющего формуле $A(\boldsymbol{x})$.

Метрика для определения степени похожести объектов

Понятие неполной выводимости может служить основой для определения метрики в множестве объектов, допускающих описания средствами предикатных формул [5]. Пусть ω_1 и ω_2 — два объекта с описаниями $S(\omega_1)$ и $S(\omega_2)$ соответственно. Заменив в каждом из описаний различные константы на различные переменные и вставив знак & между их формулами получим соответственно формулы $A_1(\boldsymbol{x}_1)$ и $A_2(\boldsymbol{x}_2)$. Выделим максимальный (q, r)-фрагмент формул $A_1(\boldsymbol{x}_1)$ и $A_2(\boldsymbol{x}_2)$. Если a_1 и a_2 — количества атомарных формул в этих формулах, то $\rho(\omega_1, \omega_2) = (a_1 - q) + (a_2 - q)$ задаёт расстояние между объектами.

Так вычисленное расстояние не вполне адекватно представлениям о похожести объектов, так как объекты, описания которых отличаются всего двумя формулами, находятся на расстоянии 4, независимо от того, сколько всего формул в их описаниях: 2 или 1000. Более адекватной является функция $d(\omega_1, \omega_2) = \rho(\omega_1, \omega_2)/(a_1 + a_2)$. Однако для последней не выполняется неравенство треугольника, поэтому её можно назвать степенью похожести объектов.

Вычисление расстояния между описаниями объектов позволяет, в свою очередь, использовать широко известный принцип «ближайший сосед» не только при наличии конечномерного пространства признаков (как при логико-алгебраическом подходе), но и в пространстве предикатных формул.

Построение многоуровневого описания классов

В [1] описано построение многоуровневого описания классов, позволяющее существенно уменьшить число шагов алгоритмов, решающих каждую из трёх сформулированных задач. Такое построение основано на выделении «часто» встречающихся в описаниях классов подформул $P_i^1(\boldsymbol{y}_i^1)$ «небольшой

сложности» и заменой их на новые предикаты $p_i^1(x_i^1)$, где x_i^1 — новые переменные первого уровня. При повторении этой процедуры с выделенными подформулами можно получить 2-уровневое, 3-уровневое, …, L-уровневое описание вида

$$A_k^L(\boldsymbol{x}_k^L)$$

$$\ldots$$

$$p_i^l(y_i^l) \Leftrightarrow P_i^l(\boldsymbol{y}_i^l)$$

$$\ldots$$

$$p_{nL}^L(y_{nL}^L) \Leftrightarrow P_{nL}^L(\boldsymbol{y}_{nL}^L).$$

В [1] доказаны условия на понятия «часто» и «небольшой сложности» в зависимости от того, с помощью какого алгоритма решается основная задача. Однако выделение подформул $P_i^1(\boldsymbol{y}_i^1)$ оставлялось на усмотрение исследователя или эксперта.

Понятие неполной выводимости формулы позволяет разработать алгоритм выделения подформул с требуемыми свойствами.

1. Для каждой пары элементарных конъюнкций, входящих в описания классов, посредством применения метода неполной выводимости для $A_i(\boldsymbol{x}_i) \Rightarrow \exists \boldsymbol{x}_{j \neq} A_j(\boldsymbol{x}_j)$ выделяем их максимальную подформулу $Q_{i,j}^1(\boldsymbol{x}_{i,j})$.
2. Повторяем процесс выделения общих подформул для $Q_{i\mathbf{1}\ldots il}^{l\ 1}(\boldsymbol{x}_{i1\ldots il})$ и $Q_{j1\ldots jl}^{l-1}(\boldsymbol{x}_{j1\ldots jl})$, получив их общие подформулы $Q_{i1\ldots il,\ j1\ldots jl}^l(\boldsymbol{x}_{i\mathbf{1}\ldots il,\ j\mathbf{1}\ldots jl})$ (l=2, …, L). Процесс завершится, так как на каждой итерации длины подформул уменьшаются.
3. Выберем среди подформул $Q_{i1\ldots il,\ j1\ldots jl}^l(\boldsymbol{x}_{i\mathbf{1}\ldots il,\ j\mathbf{1}\ldots jl})$ такие, которые удовлетворяют требуемым условиям и обозначим их посредством $P_i^1(\boldsymbol{y}_i^1)$ (i=1, …, n_1).
4. Формулы $P_i^{l+1}(\boldsymbol{y}_i^{l+1})$ (i=1, …, n_{l+1}, l=2, …, L) строятся из выделенных ранее подформул $Q_{i1\ldots il,\ j1\ldots jl}^l(\boldsymbol{x}_{i\mathbf{1}\ldots il,\ j\mathbf{1}\ldots jl})$ с учётом требуемых условий.

Заключение

В докладе описаны различные применения понятия неполной выводимости предикатной формулы при логико-предметном подходе к решению задач искусственного интеллекта. В частности, описаны опубликованные ранее метод решения задач с неполной информацией об объекте и введение метрики в множестве предикатных формул. Приводится основанный на понятии неполной выводимости алгоритм построения многоуровневого описа-

ния классов. Такое описание позволяет существенно уменьшить число шагов алгоритма, решающего основные NP-трудные сформулированные задачи.

Литература

1. *Косовская Т. М.* Многоуровневые описания классов для уменьшения числа шагов решения задач распознавания образов, описываемых формулами исчисления предикатов // Вестн. С.-Петербург. ун-та. Сер. 10. 2008. Вып. 1. С. 64–72.
2. *Косовская Т. М.* Частичная выводимость предикатных формул как средство распознавания объектов с неполной информацией // Вестн. С.-Петербург. ун-та. Сер. 10. 2009. Вып. 1. С. 74–84.
3. *Косовская Т. М.* Некоторые задачи искусственного интеллекта, допускающие формализацию на языке исчисления предикатов, и оценки числа шагов их решения // Труды СПИИРАН, 2010. Вып. 14. С. 58–75.
4. *Kosovskaya T.* Discrete Artificial Intelligence Problems and Number of Steps of their Solution // International Journal on Information Theories and Applications. Vol. 18. No. 1. 2011. P. 93–99.
5. *Kosovskaya T.* Distance between objects described by predicate formulas // Mathematics of Distances and Applications (Michel Deza, Michel Petitjean, Krasimir Markov (eds.)), ITHEA — Publisher, Sofia, Bulgaria, 2012. P. 153–159.

ОСОБЕННОСТИ ЯЗЫКА ПРОГРАММИРОВАНИЯ РЕФАЛ-5Е

В. А. Гошев
аспирант Кафедры информатики СПбГУ
Н. К. Косовский
д. ф-м. н., проф

Язык рефал-5е [3] — современный эффективный диалект языка программирования рефал, разработанный Гошевым В. А. и Косовским Н. К. Диалект рефал-5е основан на наиболее популярном диалекте — рефал-5 [4]. Язык программирования рефал-5е расширяет существующий диалект языка программирования рефал — рефал-5 [1] и позволяет решать современные важные задачи программирования эффективнее, чем другие диалекты языка рефал при использовании современной вычислительной техники. Среди особенностей языка программирования рефал-5е можно выделить такие, как: наличие встроенного интерпретатора, возможность подключения внешних динамически загружаемых библиотек кода, добавление возможности возврата функциями таких значений, как «Успех» и «Неуспех», возможность реализации функций высшего порядка и анонимных функций. Так же транслятор языка программирования рефал-5е оптимизирован для современных многопроцессорных систем, что позволяет увеличить скорость выполнения программ в несколько раз.

Первая особенность, которую видит программист на языке рефал-5е — это подключение модулей при помощи ключевого слова «$import», а не при помощи указания их в командной строке при запуске рефал-5е машины. Так, в языке рефал-5е все системные функции распределены на модули, например модуль SystemIO содержит различные функции ввода/вывода, модуль Math — математические функции, Cmp — функции сравнения значений, а модуль Eval содержит функции для доступа к встроенному интерпретатору. Встроенный интерпретатор — одно из основных нововведений и особенностей языка рефал-5е. Так как язык рефал является обобщением нормальных алгоритмов Маркова, то интерпретатор можно реализовать и средствами самого языка, в работе [2] указан проект реализации универсальной функции для языка рефал-5 средствами самого языка. Но стоит заметить, что реализация универсальной функции средствами самого языка будет работать значительно медленнее, чем реализация средствами рефал-5е машины, так как рефал-5е машина будет компилировать переданный ей код и потом выполнять его как любую другую функцию. Реализация универсальной функции средствами языка заставляет рефал-5е машину выполнять код, который будет вычислять значение рефал-функции, являющейся аргументом. В этом случае

происходит двойная интерпретация кода. Как было описано выше, в языке программирования рефал-5е для доступа к встроенному интерпретатору используется модуль Eval. Модуль Eval экспортирует 2 функции:

- Функция Compile компилирует переданные ей аргументы в код функции рефал-5е машины и возвращает сгенерированное для нее имя. В результате такую функцию можно вызывать в программе неограниченное количество раз без необходимости заново производить трансляцию ее кода в код рефал-машины.
- Функция Run компилирует функцию из кода, переданного первым аргументом (обособленным структурными скобками) в код рефал-5е машины и интерпретирует ее, передав скомпилированной функции аргументы для нее.

Пример подключения модулей и использования встроенного интерпретатора:

```
/*Подключение необходимых модулей*/
$import SystemIO;
$import Cmp;
$import Eval;
/*Функция — точка входа в программу, читаем строку
с стандартного ввода и передаем ее функции Check, результат
выполнения этой функции выводим на экран*/
Main {
 = <SystemIO::PrintLn <Check (<SystemIO::ReadLn>) (1 2) (3) > >;
};
/*Функция Check принимает Код функции, аргументы для нее и
ожидаемый результат и проверяет, вернет-ли эта функция (если
она синтаксически верна) ожидаемый результат*/
Check {
(e.Code) (e.Args) (e.Result) = {
  <Cmp::Eq (<Eval::Run (e.Code) e.Args >) (e.Result)> 'Ok';
  'Fail';
  };
};
```

Второй основной особенностью языка рефал-5е является возможность легко подключить динамически-загружаемую библиотеку и использовать ее функции. Так же не маловажным является то, что транслятор языка рефал-5е оптимизирован для современных многопроцессорных компьютеров. Когда рефал-5е машина встречает команду вызова функции и если эта функция не создает никаких побочных действий, то вызов функции добавляется в очередь выполнения с указанием, куда должны быть помещены результаты

выполнения функции, далее свободный поток выполнения начинает выполнение этой функции, таким образом достигается возможность параллельного выполнения функций, что в некоторых случаях позволяет значительно увеличить быстродействие программы. Так, на четырех-процессорном компьютере удалось достичь ускорения выполнения некоторых программ в 3 раза.

В данный момент язык рефал-5е разработан и реализован для него транслятор, который проходит тестовую эксплуатацию.

Литература

1. *Бабаев И. О., Герасимов М. А., Косовский Н. К., Соловьев И. П.* Интеллектуальное программирование. Турбо Пролог и Рефал-5 на персональных компьютерах. Л.: Издательство ЛГУ, 1992.
2. *Гошев В. А*. Схема универсальной функции для языка рефал-5 // Материалы всероссийской научной конференции по проблемам информатики. 23–26 апр. 2013 г., Санкт-Петербург. СПб.: Издательство ВВМ, 2013. 792с.
3. *Гошев В. А., Косовский Н. К*. Рефал-5е: разработка языка и реализация транслятора // Сборник тезисов конференции «Технологии Microsoft в теории и практике программирования». СПб.: Изд-во Политехн. ун-та, 2014. 138 с.
4. *Turchin V. F*. Refal-5. Programming Guide and Reference Manual. New England Publishing Co., Holyoke, 1989.

КВАДРАТИЧНЫЙ ПО ВРЕМЕНИ АЛГОРИТМ ПРИБЛИЖЕННОГО РАЗБИЕНИЯ МНОЖЕСТВА НАТУРАЛЬНЫХ ЧИСЕЛ С ГАРАНТИРОВАННОЙ ОЦЕНКОЙ ТОЧНОСТИ

М. А. Герасимов
Кафедра информатики СПбГУ
E-mail: ge@star.math.spbu.ru

Аннотация. Рассматривается алгоритм приближенного разбиения произвольного множества натуральных чисел (предполагается, что все числа больше нуля) на два дизъюнктных множества равного «веса». Как показано в данной работе, временная сложность алгоритма не превосходит O(m2), т. е. полиномиальна от размера входного набора данных. Алгоритм не дает точного разбиения, точность алгоритма не превышает максимального числа из входного набора данных (более точно, разности максимального и минимального из входных чисел). Алгоритм может быть преобразован в алгоритм разбиения входного потока данных на два потока натуральных чисел приблизительно равного веса в реальном масштабе времени.

Во многих случаях данный алгоритм с небольшой модификацией гарантированно дает точное разбиение входного набора данных [1], что делает его удобным даже в статическом виде в ряде специальных случаев входных наборов данных.

1. Основные определения

Пусть задано множество $X = \{x_1, x_2, \ldots, x_m\}$ на котором задана весовая функция $W: X \to N$, где $W(x)$ — вес элемента x из X, соответственно. Задача разбиения множества X сводится к нахождению подмножеств Y и Z таких, что $Y \cup Z = X$, $Y \cap Z = \varnothing$, $\sum_{x \in Y_1} W(x) = \sum_{x \in Z} W(x)$.

В общем случае для произвольного набора данных полиномиального алгоритма для решения этой задачи не найдено. Более того, доказано, что нахождение такого точного решения — это NP — полная задача [2, 3]. Поэтому, для практических нужд полезен так называемый приближенный алгоритм, который будет разбивать множество X на два подмножества не совсем точно, но будет работать полиномиальное время от размера входного набора данных.

2. Детерминированная машина Тьюринга

В качестве модели вычисления рассмотрим детерминированные машины Тьюринга [4] с конечным множеством состояний $Q = \{q_1, \ldots, q_N\}$. Предпо-

ложим, что входные данные записаны на входной ленте, обрабатываются на рабочей ленте, а результат записывается на выходной ленте.

При работе машины Тьюринга используем алфавит, состоящий из четырех символов $\{\#, b, 0, 1\}$. Этот алфавит применяется для всех трех лент машины Тьюринга: входной, выходной и рабочей. Результатом работы алгоритма считаются битовая последовательность, представляющая два выходных множества натуральных чисел. Множества выходной цепочки представляются номером и содержанием. Номер указывает, к какому из множеств Y или Z принадлежит данное число, содержание — двоичное представление самого числа. На выходной ленте можно только записывать результат вычисления в виде последовательности символов рабочего алфавита.

Входные данные записаны в виде битовой последовательности на входной ленте между двумя маркерами '#'. Считывание второго маркера означает конец цепочки входных данных. Входная лента позволяет считывать входные данные произвольное количество раз.

3. Алгоритм разбиения

Для реализации данного алгоритма отсортируем множество X по возрастанию относительных весов его элементов. Составляем список L_x, содержащий этот отсортированный массив элементов X.

Построим начальные множества Y и Z, содержащие по одному элементу. Определим среднее из множества X, как среднее относительных весов элементов множества X. Пусть $\mu = \frac{1}{|X|}\sum_{x\in X} W(x)$.

Первый шаг алгоритма. Найдем такие x_i и x_{i+1}, что $W(x_i) = W(x_{i+1}) = \mu$. Если это удалось, то поместим x_i в Y, x_{i+1} в Z. Удалим из списка L_x оба элемента x_i и x_{i+1}. На этом первых шаг алгоритма будет закончен. Если же не существует x_i и x_{i+1}, то найдем x_j из $\{x \mid W(x) > \mu\}$ с наибольшим весом, и x_{i_1}, x_{i_2}, …, x_{i_k} из $\{x \mid W(x) < \mu\}$ такие, что $x_j \geq \sum_{x\in\{x_{i_1}, x_{i_2} \ldots x_{i_k}\}} W(x)$. При этом, добавление любого $x \in \{x \mid W(x) < \mu\}$, отличного от $\{x_{i_1}, x_{i_2} \ldots x_{i_k}\}$, приводит к нарушению данного неравенства. Поместим элементы $\{x_{i_1}, x_{i_2} \ldots x_{i_k}\}$ в Y, элемент x_j в Z. Удалим $x_{i_1}, x_{i_2} \ldots x_{i_k}$ и x_j из X. Закончим первый шаг алгоритма.

Последующие шаги алгоритма. Для всех $k \geq 1$, пока X_k содержит элементы: $\mu_k = \frac{1}{|X_k|}\sum_{x\in X_k} W(x)$. Найдем такие x_i и x_{i+1}, что $W(x_i) = W(x_{i+1}) = \mu_k$. Если это удалось, то поместим x_i в Y, x_{i+1} в Z. Удалим из X оба элемента x_i и x_{i+1}. Получим множество X_{k+1}. Закончим k-тый шаг алгоритма.

Если же не существует таких x_i и x_{i+1}, то найдем x_j из $\{x \mid W(x) > \mu,\ x \in X_k\}$ с наибольшим весом, и x_{i_1}, x_{i_2}, …, x_{i_k} из $\{x \mid W(x) < \mu,\ x \in X_k\}$ такие, что $x_j \geq \sum_{x \in \{x_{i_1} x_{i_2} \ldots x_{i_k}\}} W(x)$. При этом, добавление любого $x \in \{x \mid W(x) < \mu,\ x \in X_k\}$, отличного от $\{x_{i_1}, x_{i_2}, ..., x_{i_k}\}$ приводит к нарушению данного неравенства. Поместим элементы $\{x_{i_1}, x_{i_2}, ..., x_{i_k}\}$ в наибольшее по весу из множеств Y и Z, элемент x_j в наименьшее по весу из множеств Y и Z. Удалим $x_{i_1}, x_{i_2}, ..., x_{i_k}$ и x_j из X_k. Получим новое множество X_{k+1}. Закончим k-тый шаг алгоритма.

Если множество X_k содержит только один элемент, поместим его в наименьшее по весу из множеств Y и Z.

4. Время выполнения алгоритма

Предварительная сортировка может иметь сложность порядка $O(m \log m)$, где m — мощность множества X.

На каждом шаге число элементов в X_k уменьшается не менее чем на два, таким образом, общее число шагов $k \leq \frac{m}{2} + 1$.

Сложность каждого шага. Вычисление μ_k требует не более m сложений и одно деление. Для поиска элемента x_j и элементов $x_{i_1}, x_{i_2}, ..., x_{i_k}$ не более m сравнений. Включение/удаление элементов $x_{i_1}, x_{i_2}, ..., x_{i_k}$ и x_j требует не более m операций добавления/удаления. Итого на каждом шаге требуется не более $m + 1 + m + m = 3m + 1 = O(m)$ шагов.

Общая сложность алгоритма. Общая сложность алгоритма может быть определена как произведение числа шагов на оценку сложности каждого шага:

$$\left(\frac{m}{2} + 1\right) O(m) = O(m^2).$$

Таким образом, для приближенного разбиения множества $X = \{x_1, x_2, \ldots, x_m\}$ натуральных чисел на два равновесных подмножества Y и Z требуется квадратичное от m число шагов.

5. Погрешность алгоритма

После каждого шага алгоритма разница весов множеств Y и Z не превосходит разницы значений $W(x_j)$ и $\max\{x \mid W(x) \leq \mu_k\}$. Таким образом, после нормального завершения работы алгоритма, погрешность разбиения не превосходит разницы между максимальным значением и минимальным. Если алгоритм заканчивает свою работу с одним элементом, то погрешность ал-

горитма не превосходит минимального из весов элементов, превосходящих по весу среднее, т. е. $\min_{x\in\{x|W(x)>\mu(X)\}} W(x)$.

6. Пример работы алгоритма

Рассмотрим пример работы алгоритма на множестве $X_1=\{4, 5, 6, 7, 8, 9\}$. Всего элементов $m=6$, сумма равна 39, среднее $M_1=6{,}5$. Вес среднего элемента находится между x_3 и x_4, поэтому $x_j=9$, $x_i=6$, $Y=\{6\}$, $Z=\{9\}$, $X_2=\{4, 5, 7, 8\}$.

Для $X_2=\{4, 5, 7, 8\}$ $m=4$, сумма равна 24, среднее $M_2=6$. Вес среднего элемента находится между x_2 и x_3, поэтому $x_j=8$, $x_i=5$, $Y=\{6, 8\}$, $Z=\{5, 9\}$, $X_3=\{4, 7\}$.

Для $X_3=\{4, 7\}$ $m=2$, сумма равна 11, среднее $M_3=5{,}5$. Вес среднего элемента находится между x_1 и x_2, поэтому $x_j=7$, $x_i=4$, $Y=\{4, 6, 8\}$, $Z=\{5, 7, 9\}$, $X_4=\varnothing$. Множество пусто, алгоритм останавливается.

Вес $W(Y)=18$, вес $W(Z)=21$. Разность весов множеств Y и Z равна 3 и не превосходит разности $9\_4=5$.

Литература

1. *Герасимов М. А*. NP-полная задача о разбиении множества на К подмножеств и субэкспоненциальные функции // Материалы всероссийской научной конференции по проблемам информатики. Санкт-Петербург, СПбГУ, 2013. С. 74–77.
2. *Гэри М., Джонсон Д*. Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982.
3. *Минский М*. Вычисления и автоматы. М.: Мир, 1971.
4. *Fischetti M., Martello S*. Wost-case analysis of the differencing method for the partition problem // Math. Programming. 1987. Vol. 37. No. 1. Pp. 117–120.
5. *Horowits E., Sahni S*. Fundamentals of Computer Algorithms. Computer Science Press, 1978.
6. *Mertens S*. The Easiest Hard Problem: Number Partitioning // Computational complexity and statistical physics. Oxford University Press US, 2006. P. 125.

ИССЛЕДОВАНИЕ НОВОСТНОГО ПОТОКА МЕТОДАМИ ЧАСТОТНОГО АНАЛИЗА

Д. М. Августинов
аспирант кафедры информатики СПбГУ
E-mail: avgustinov@bk.ru

Аннотация. Проблема обработки большого количества информации в сжатые сроки сегодня актуальна как никогда. Составление прогноза на основе информации, полученной из информационных источников, является задачей, от которой может зависеть состояние не только отдельно взятой компании, но и экономики целой страны. Целью данной статьи является описание программного комплекса на языке Java, использующего алгоритм обработки текстов основанного на комбинации методов латентно-семантического анализа и методов экстраполяции.

Введение

Исследуемый подход ставит перед собой задачу нахождения в последовательности статей одной тематики закона распределения частот наиболее информативных терминов или словосочетаний. Рассмотри задачу на примере.

Будем исследовать периодические статьи экономических аналитиков о нефтяном секторе России. Допустим обзор выходит раз в неделю. Тогда соберем 8 обзоров за 2 месяца и попробуем найти неявные закономерности. Входными данными нашего алгоритма будут служить отобранные 8 текстовых обзоров, на выходе мы получим таблицу из наиболее информативных терминов и словосочетаний по теме нефтяной сектор с указанием динамики их упоминаний в течение времени. На основании полученной информации можно сделать выводы, недоступные при последовательном прочтении статей экспертом. Если на выходе алгоритм покажет, что в течение 2 месяцев частота упоминаний словосочетаний *позитивная динамика, повышение цены, рекомендация к покупке и нефть дорожает* стабильно растет, то можно сделать вывод о том, что аналитик все сильнее уверен в том, что цены на нефть будут расти. На основании полученных данных возможно также построить прогноз о частоте упоминаний различных слов и словосочетаний в будущем.

Данный метод также применим для формирования консенсус прогнозов. Если в качестве входных данных взять множество статей различных аналитиков по заданной тематике, то на выходе можно получить частотную таблицу семантического ядра исследуемых статей.

Основные задачи

1. Разработать программный комплекс на основе методов частотного анализа, позволяющего прогнозировать экспертные оценки о состояние фондового рынка.
2. Разработать алгоритм прогнозирования экспертных оценок.
3. Разработать алгоритм интерпретации полученных численных результатов.

Основные результаты

1. Разработан и реализован программный комплекс, позволяющий прогнозировать новостной фон о состоянии фондового рынка на основании существующих экспертных прогнозов.
2. Разработан алгоритм прогнозирования экспертных оценок, основанный на методах частотного анализа и математической статистики.
3. Предложен интерпретатор численных результатов, полученных с использование алгоритма прогнозирования.

Предлагаемый метод прогнозирования состоит из следующих этапов:

1. Определение области для прогнозирования;
2. сбор статей на заданную тематику при помощи экспертного суждения или методов автоматической классификации текстов;
3. процедура нормализации текстов;
4. получение частотной характеристики (вектора) каждой статьи (координаты точки в *N*-мерном пространстве) в различные промежутки времени (от 1 до T);
5. формирование базиса частотных характеристик;
6. экстраполяция полученных значений координат векторов для нахождения координат системы в момент времени T+1;
7. интерпретация полученной экстраполяции в виде прогноза.

Был предложении следующий алгоритм нормализации, который позволяет убрать из текста информацию, не несущую смысловую составляющую. Это означает, что из текста удаляются служебные части речи: предлоги, местоимения, союзы и т. д. Также удаляются стоп-слова. Под стоп-словами также понимаются часто употребительные вспомогательные слова, которые по отдельности не несут смысловой нагрузки. Примером стоп слов могут служить:

1) Отдельно стоящие знаки препинания: . , / ? ! ; : ( )
2) Цифры: 0, 1, 2, 3, 4, и т. д. 3) Отдельно стоящие буквы алфавита: а, б, в, г, д, е, ..., ь, ы, ю, я.

4) Слова выбранные пользователем, которые, по его мнению, являются мусорными.

Оставшиеся слова приводятся к своей нормальной словарной форме. Для этого могут используется алгоритм лемматизации.

Лемматизация — процесс приведения словоформы к лемме — её нормальной (словарной) форме. Данный процесс возможен с использование специального словаря, в котором различным словоформам сопоставляются соответствующие леммы.

Автором предложен подход к формализации входного потока текста. Эффективность работы с данными зависит от способа представления текста.

Наиболее популярной моделью представления текста является Vector Space Model (VSM) . Согласно этому представлению текст представляется в виде вектора, размерность которого измеряется количеством параметров текста. Значения этих параметров — это функции частот, с которыми эти параметры появляются в текстовом корпусе. В данном случае игнорируются порядок между словами и взаимосвязи.

Большинство представлений происходят от модели VSM. Возможны представления, основывающиеся на фразах, а не на отдельных словах; другие учитывают семантику слов или отношения между ними, в третьих используется иерархическая структура текста .

До формирования вектора, текст проходит процедуру нормализации, а так же формируется базис частотных характеристик .

Сравнительная характеристика представления текста по словам и фразам представлена в таблице.

Таблица 1

Преимущества и недостатки слов и фраз в представлении документа

	Преимущества	**Недостатки**
Слова	— хорошее качество при статистической обработке; — определение синонимов; — наличие детально разработанных алгоритмов	— отсутствие контекстной информации; с выявлением; — проблемы устойчивых словосочетаний
Фразы	— наличие контекстной информации; — возможность обнаружить устойчивые словосочетания	— среднее качество при статистической обработке

Предложено использование методов экстраполяции для прогнозирования значений частот словоформ.

На практике были исследованы следующие методы:

- линейная аппроксимация,
- многочлен Лагранжа,
- кубический сплайн.

Для этого были проведено исследование на различных комбинациях статей. Размерность множества входящих статей не превышала десяти. Данная размерность обусловлена тем, что аналитические прогнозы, как правило, строятся не более чем на квартал, а за это время по отдельному эмитенту выходит не более десяти промежуточных аналитических статей от одного источника.

Линейная аппроксимация

Построим линейный аппроксимирующий полином для полученных данных. Для этого используем Метод наименьших квадратов (МНК). В качестве параметра x возьмем номер периода изучаемой статьи. В качестве параметра y — значение лексемы из частотной характеристики соответствующей статьи. Таким образом, опишем зависимость y от x уравнением вида $P_1(x) = a_0 + a_1 \cdot x$.

Найдем неизвестные коэффициенты a_0 и a_1 по МНК

$$F = \sum_{i=1}^{n} \left(y_i - a_0 - a_1 \cdot x_i \right)^2 \to \min;$$

$$\begin{cases} \dfrac{\partial F}{\partial a_0} = -2 \cdot \sum\limits_{i=1}^{n} \left(y_i - a_0 - a_1 \cdot x_i \right) \cdot 1 = 0, \\ \dfrac{\partial F}{\partial a_1} = -2 \cdot \sum\limits_{i=1}^{n} \left(y_i - a_0 - a_1 \cdot x_i \right) \cdot x_i = 0; \end{cases}$$

$$\begin{cases} \sum\limits_{i=1}^{n} y_i - a_0 \cdot n - a_1 \cdot \sum\limits_{i=1}^{n} x_i = 0, \\ \sum\limits_{i=1}^{n} \left(y_i \cdot x_i \right) - a_0 \cdot \sum\limits_{i=1}^{n} x_i - a_1 \cdot \sum\limits_{i=1}^{n} x_i^2 = 0; \end{cases}$$

$$\begin{cases} a_0 \cdot n + a_1 \cdot \sum\limits_{i=1}^{n} x_i = \sum\limits_{i=1}^{n} y_i, \\ a_0 \cdot \sum\limits_{i=1}^{n} x_i + a_1 \cdot \sum\limits_{i=1}^{n} x_i^2 = \sum\limits_{i=1}^{n} \left(y_i \cdot x_i \right); \end{cases}$$

Решим систему уравнений и выразим коэффициенты:

$$= \frac{\begin{vmatrix} \sum_{i=1}^{n} y_i & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} (y_i \cdot x_i) & \sum_{i=1}^{n} x_i^2 \end{vmatrix}}{\begin{vmatrix} n & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 \end{vmatrix}} = \frac{\sum_{i=1}^{n} y_i \cdot \sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} x_i \cdot \sum_{i=1}^{n} (y_i \cdot x_i)}{n \cdot \sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2};$$

$$: \frac{\begin{vmatrix} n & \sum_{i=1}^{n} y_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} (y_i \cdot x_i) \end{vmatrix}}{\begin{vmatrix} n & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & \sum_{i=1}^{n} x_i^2 \end{vmatrix}} = \frac{n \cdot \sum_{i=1}^{n} (y_i \cdot x_i) - \sum_{i=1}^{n} x_i \cdot \sum_{i=1}^{n} y_i}{n \cdot \sum_{i=1}^{n} x_i^2 - \left(\sum_{i=1}^{n} x_i\right)^2}.$$

Получив коэффициенты a_0 и a_1 можно найти уравнение искомой прямой. Подставив в него значение x в период $T+1$, получим результат экстраполяции частоты леммы из частотной характеристики

В результате при помощи методов экстраполяции можно получить координаты точки в момент времени $T+1$. Выполнив экстраполяцию по всем словам из базиса, мы получим частотную характеристику прогнозируемого новостного фона.

Заключение и выводы

В данной статье рассмотрен программный комплекс для выявления неявных закономерностей в аналитических статьях при помощи методов частотного анализа и экстраполяции. Предложены два варианта представления текстов для используемой аналитической модели, а так же даны обоснования их использования. Описанный метод и его реализация наиболее подходят для изучения аналитических прогнозов с экономическим уклоном, так как описывают состояния конкретного субъекта(состояние акций конкретной компании, экономика страны и т. д.) через равный периоды времени. Стоит отметить, что пользователь может выбрать для изучения статьи любой тематики, написанные в текстовом формате , что говорит об универсальности метода. В дальнейшем планируется доработать предложенный подход и привести данные информационные модели к автоматической системе построения и прогноза, а так же к модели построения консенсус-прогнозов.

Литература

1. *Мисько О. Н*. Рынок ценных бумаг: организация и функционирование. СПб.: Изд-во С.-Петерб. Ун-та, 2001. С. 91–92.
2. *Бердникова Т. Б*. Рынок ценных бумаг. М.: ИНФРА-М, 2002.
3. *Писарева О. М*. Методы социально-экономического прогнозирования. М.: ГУУ — НФПК, 2003. С. 10.
4. *Константиновская Л. В*. Методы и приемы прогнозирования. 2006.
5. *Тузов В. А*. Математическая модель языка. Л.: Изд-во Ленингр. Ун-та, 1984.
6. *Элдер А*. Как играть и выигрывать на бирже: Психология. Технический анализ. Контроль над капиталом. М.: Альпина Бизнес Букс, 2007. 472 с.
7. *Турунцева М. Ю*. Прогнозирование в России: обзор основных моделей // Экономическая политика. 2011. № 1. С. 193–202.
8. *Thomas Landauer*. Introduction to Latent Semantic Analysis / Thomas Landauer Peter W. Foltz, Darrell Laham. 1998. P. 259–284
9. *Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, Richard Harshman*. Indexing by Latent Semantic Analysis // Journal of the American Society for Information Science. 1990. 41 (6). P. 391–407.
10. *Дюк В. А., Флегонтов А. В., Фомина И. К*. Применение технологий интеллектуального анализа данных в естественнонаучных, технических и гуманитарных областях // Известия Российского Государственного Педагогического Университета им. А. И. Герцена. Естественные и точные науки. Вып. 138. 2011.
11. *Коваленко О. С*. Обзор проблем и перспектив анализа данных.
12. *Hearst M. A*. Untangling text data mining // Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics. 1999. P. 3–10.
13. *Fan W., Wallace L., Rich S., Zhang Z*. Tapping the power of text mining // Communications of the ACM. 2006. 49(9). P. 76–82.
14. *Оробинская Е. А., Кочуева З. А*. Технологии text mining: обзор методов и задач обработки смысловой информации // Вестник ХНТУ. № 2 (38). 2010.
15. *Yang Y*. An Evaluation of Statistical Approaches to Text Categorization // Journal of Information Retrieval. 1999. 1. Pp. 69–90.
16. *Hobbs J., Appelt D., Bear J., Israel D., Kameyama M., Stickel M., Tyson M*. Hobbs FASTUS: a Cascaded Finite-State Trasducer for Extracting Information from Natural-Language Text // AIC, SRI International. Menlo Park California, 1996.

КОМПЬЮТЕРНАЯ РЕАЛИЗАЦИЯ ТЕМАТИЧЕСКОЙ КЛАССИФИКАЦИИ ТЕКСТОВ МЕТОДАМИ ЧАСТОТНОГО АНАЛИЗА

Е. Д. Заболотский
Санкт-Петербург

Введение

В данной статье представлена реализация задачи классификации текстов методом частотного анализа. На текущий момент существует множество реализованных решений задачи классификации, многие из которых представлены на сайте www.kdnuggets.com. В данной работе за основу для сравнения используются нормативно-правовые акты Российской Федерации, которые составляются по правилам юридической техники и отличаются от привычного всем стиля написания. Задача реализована, в результате чего получен инструмент для классификации текстов и отнесения их к компетенциям управлений Федеральной антимонопольной службы посредством сравнения текстов с частотными словарями, полученными в результате анализа нормативно-правовых актов по отраслевым управлениям Федеральной антимонопольной службы. Результатом данного исследования является алгоритм классификации текстов. Кроме того, получены частотные списки по отраслевым управлениям Федеральной антимонопольной службы, что может быть использовано для прикладных задач, например, таких как, обработка обращений, анализ текста на принадлежность к тому или иному управлению.

Задача классификации

Задача классификации текстов (далее — Задача) является подзадачей мультидисциплинарной области Data Mining. которая возникла и развивается на базе таких наук как прикладная статистика, распознавание образов, искусственный интеллект, теория баз данных [1].

Результатом решения задачи классификации является выделение признаков, которые характеризуют категории объектов исследуемого набора текстов. По этим признакам новый объект можно отнести к той или иной категории.

На текущий момент основными методами классификации текстов являются [2]: классификация с помощью деревьев решений, байесовская (наивная) классификация, классификация при помощи искусственных нейронных сетей, классификация методом векторного представления документа, статистические методы, в частности, линейная регрессия, классификация при помощи метода ближайшего соседа, классификация CBR-методом, классификация при помощи генетических алгоритмов, классификация методом частотного анализа.

Постановка задачи

Существует множество категорий. Существует обучающая выборка для каждой категории. Рассматривается входящий текстовый файл. Необходимо определить принадлежность входящего файла к той или иной категории.

В качестве категорий выступают 13 основных отраслевых направлений деятельности Федеральной антимонопольной службы [3]:

- ЖКХ, строительство и природные ресурсы;
- Государственный заказ;
- Промышленность и оборонный комплекс;
- Информационные технологии;
- Топливно-энергетический комплекс;
- Транспорт и связь;
- Электроэнергетика;
- Финансовые рынки;
- Реклама и недобросовестная конкуренция;
- Социальная сфера и торговля;
- Химическая промышленность и агропромышленный комплекс;
- Международное экономическое сотрудничество;
- Иностранные инвестиции.

В качестве текстов, определяющих категории, используются нормативно-правовые акты (далее — НПА) Российской Федерации, регулирующие отношении в соответствующей отрасли.

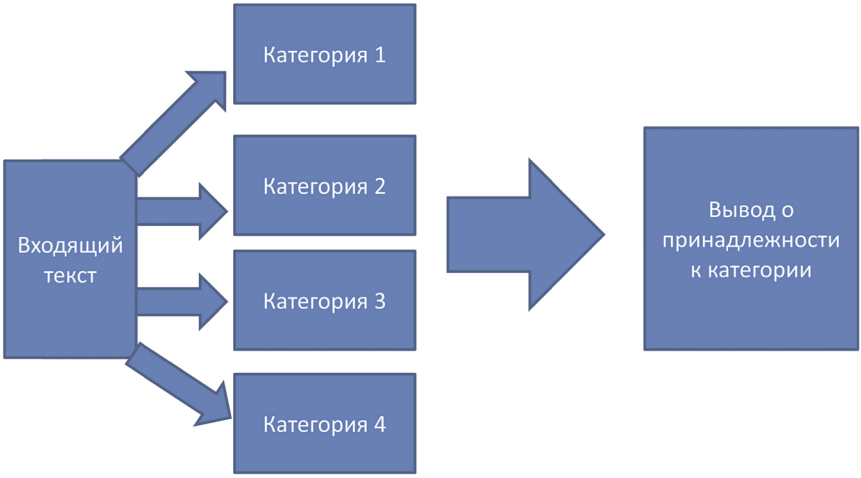


Рис. 1

Данная работа преследует следующие цели:

- снижение документооборота в Федеральной антимонопольной службе;
- уменьшение времени рассмотрения заявлений, обращений и жалоб;
- формирование базы данных частотных списков по каждому управлению (департаменту) Федеральной антимонопольной службы;
- разработка и реализация алгоритма классификации документов в правовой среде.

Входной информацией для программы должен являться текстовый файл формата doc, txt, rtf на русском языке.

Графическое представление задачи приведено на рисунке 1.

НПА

НПА — официальный документ установленной формы, принятый в пределах компетенции уполномоченного государственного органа с соблюдением установленной законодательством процедуры, содержащий общеобязательные правила поведения, рассчитанные на неопределённый круг лиц и неоднократное применение [4].

НПА имеют определенный вид, облекаются в документальную форму и составляются по правилам юридической техники. НПА, действующие на территории Российской Федерации, образуют единую систему [5].

Существуют следующие виды НПА:

- Законы;
- Указы Президента;
- Постановления Правительства;
- Приказы федеральных органов исполнительной власти;
- Локальные НПА.

НПА имеют иерархическую структуру, соответствующую вышеуказанному списку. Основными НПА являются федеральные законы, на основании законов выходят постановления, на основании постановлений выпускаются приказы федеральных органов исполнительной власти.

Метод решения задачи

Алгоритм классификации

Изначально было необходимо определить категории. Для этого были выбраны НПА соответствующие вышеуказанным направлениям. Для обучения каждой категории было выбрано по 30 НПА, которые наиболее полно отражают то или иное направление, в результате чего были построены частотные словари по каждой категории, упорядоченные по убыванию.

В качестве обучающей выборки использовались законы (в среднем 15% от обучающей выборки для категории), постановления Правительства

(в среднем 30% от обучающей выборки для категории) и приказы федеральных органов исполнительной власти (в среднем 55% от обучающей выборки для категории). Законы формируют общее представление категории, а постановления и приказы углубляются в соответствующую отрасль в полном объеме и задают наиболее точный словарь категории.

Сравнение текста с категорией происходит по следующим параметрам:

1) Среднеквадратичное отклонение частот текста и частотного списка;
2) Число совпавших слов в категории и тексте;
3) Число совпавших слов, имеющих наибольшую частоту в частотных списках категорий;
4) Число совпадений словосочетаний.

Для решения задачи проводится анализ входящего текста, в результате которого исключаются частицы, предлоги, местоимения, слова, длина которых не превышает трех символов, и определяются частоты каждого слова анализируемого текста.

После анализа входящего текста происходит сравнение частотного списка текста с частотными списками категорий. Сравнение происходит, если во входящем тексте и в категории имеется более трех общих слов. Вывод о принадлежности к категории делается исходя из максимального значения величины F, которая определяется следующим образом:

$$F = A(1-\alpha) + B \cdot w + C \cdot (C_1 \cdot q_1 + C_2 \cdot q_2 + C_3 \cdot q_3) + D \cdot \beta,$$

где A, B, C, C_1, C_2, C_3, D — коэффициенты, определяемые на основании тестирования; α — среднеквадратичное отклонение текста от категории; w — количество совпавших слов в категории и тексте; q_1 — количество слов в тексте, попадающих в список из 25 слов с наибольшей частотой по категории; q_2 — количество слов в тексте, попадающих в список из 60 слов с наибольшей частотой по категории — q_1 по соответствующей категории; q_3 — количество слов в тексте, попадающих в список из 80 слов с наибольшей частотой по категории — $(q_1 + q_2)$ по соответствующей категории; β — количество совпавших связок слов в категории и тексте.

Связки слов определяются как пара существительное + прилагательное или прилагательное + существительное, записываются в базу данных и упорядочиваются по существительному в алфавитном порядке.

Таким образом, принадлежность текста к категории определяется исходя из максимального значения F. Предусмотрена возможность корректировки коэффициентов A, B, C, C_1, C_2, C_3, D.

Для реализации вышеуказанной задачи был выбран метод, основанный на частотном анализе. Частотный анализ основывается на предположении о существовании нетривиального статистического распределения отдельных слов и их последовательностей в тексте [6]. Исходя из анализа методов решения задачи классификации, частотный анализ имеет преимущество над дру-

гими методами классификации в разрезе скорости и точности для небольших текстов.

Используемые средства

В качестве программных средств для решения Задачи были выбраны язык Java и средства разработки Eclipse. Для хранения и обработки текстовой информации необходимы база данных и СУБД. В качестве базы данных была выбрана СУБД MS SQL Server.

Описание приложения

На рисунке 2 представлено реализованное приложение.

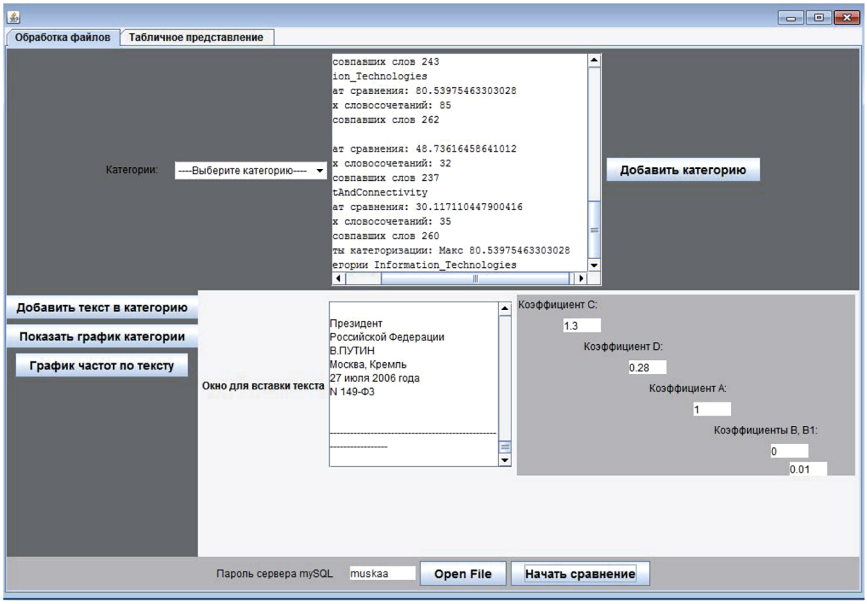


Рис. 2

Реализовано два способа обработки входящего файла: скопировать файл в диалоговое окно или открыть файл через проводник. Также в приложении предусмотрена возможность добавления новой категории и добавление текста в определенную категорию в ручном режиме. При новом сравнении приложение обрабатывает категорию с учетом этого текста и обновляет данные в базе данных.

Через вкладку табличное представление осуществляется просмотр частотных списков входящего текста и любой из категорий. Аналогичные спис-

ки по категориям хранятся в базе данных. Также реализована возможность просмотра графиков частот как по тексту, так и по категории.

Описание экспериментов

Было выбрано и обработано приложением 1500 писем, поступивших в Федеральную антимонопольную службу. Приложение правильно распределило 1215 писем. 147 писем были отнесены к смежным управлением. Например, часть писем, которые относятся к компетенции Управления контроля электроэнергетики были отнесены к Управлению топливно-энергетического комплекса и наоборот. Оставшиеся 138 писем были отнесены к другим управлениям, что является целью дальнейшей работы в сторону увеличения точности работы приложения. Стоит отметить, что объем текстов, которые бели неправильно определены, был менее страницы.

Также приложением были обработаны проекты НПА, которые с 1 апреля 2013 года размещаются на сайте www.regulation.gov.ru соответствующими министерствами или ведомствами. Было выбрано и обработано 1000 проектов НПА. Стоит отметить, что для классификации проектов НПА в качестве категорий использовались 16 основных направлений деятельности Правительства Российской Федерации. Приложение правильно определило принадлежность 851 проектов НПА. 92 проектов НПА были отнесены к смежным категориям, оставшиеся 57 проектов НПА — к другим категориям.

Заключение

Поставленная задача реализована, приложение готово к работе, частотные списки категорий составлены. Целью дальнейшей работы является устранение неточностей при обработке текстов небольшого объема посредством реализации фильтра общих часто встречающихся слов, таких как «постановление», «федерация» и.т.п. Выбранная метрика показала достаточную точность и быстродействие. В результате тестирования в 81% принадлежность входящего текста была определена верно, в 9,8% текст был отнесен к смежной категории и в 9,2% категория текста была определена некорректно. Результаты экспериментов приведены на рисунке 3.

Таким образом, данное приложение может использоваться Федеральной антимонопольной службой в целях автоматической классификации входящей корреспонденции. На данный момент такая классификация осуществляется в ручном режиме после прочтения документа.

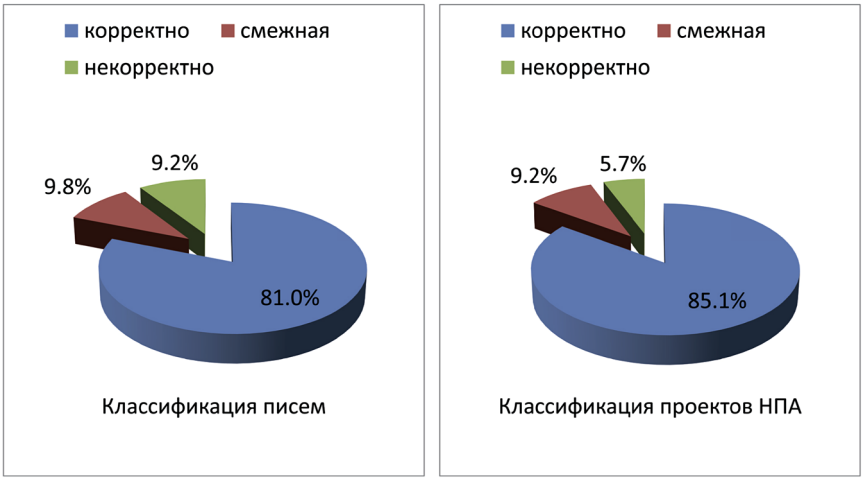


Рис. 3

Литература

1. *Дюк В., Самойленко А.* Data Mining: учеб. курс. СПб/: Питер, 2001. 368 с.
2. *Sebastiani F.* Machine Learning in Automated Text Categorization // ACM Computing Surveys. 2002. Vol. 34. P. 1–47.
3. Официальный сайт Федеральной антимонопольной службы Российской Федерации (URL: www.fas.gov.ru).
4. *Поляков А. В., Тимошина Е. В.* Общая теория права. СПб.: Изд. Дом С.-Петерб. гос. ун-та, 2005. 472 с.
5. *Червонюк В. И., Гойман-Калинский И. В., Иванец Г. И.* Элементарные начала общей теории права. М.: Право и закон; КолосС, 2003. 542 с.
6. *Рандал Р. Б.* Частотный анализ. 1989. 389 с.

ЭФФЕКТИВНОЕ ПО ВРЕМЕНИ И ПАМЯТИ ВЫЧИСЛЕНИЕ W-ФУНКЦИИ ЛАМБЕРТА

М. А. Старицын, С. В. Яхонтов

Каф. информатики, математико-механический факультет, Санкт-Петербургский государственный университет

E-mails: m.staritzyn2012@yandex.ru, SergeyV.Yakhontov@gmail.com

В данной работе предлагается алгоритм расчета вещественной **W**-функции Ламберта W_0 [1] на отрезке $[-(r \cdot e)^{-1}, (r \cdot e)^{-1}]$, где r — рациональное, $r > 1$, (говоря более точно, основной ветви W_0 вещественной **W**-функции Ламберта W) с полиномиальной временной и линейной емкостной сложностью на машине Тьюринга. Данный алгоритм строится на основе разложения в ряд Тейлора данной функции с использованием алгоритма вычисления линейно сходящихся степенных рядов в пределах **FP//LINSPACE** из [2] в качестве базового алгоритма.

При построении алгоритмического аналога функции Ламберта W_0 берется модель алгоритмических чисел и функций, изложенная в [3]. Посредством **FP//LINSPACE** будем обозначать класс алгоритмов, полиномиальных по времени и линейных по памяти при вычислении на машине Тьюринга [4].

Последовательность $\varphi : \mathbb{N} \to \mathbf{D}$ (здесь **D** — множество двоично-рациональных чисел) двоично-рационально сходится к вещественному числу x, если для любого $n \in N$ выполняется $|\varphi(n) - x| \le 2^{-n}$.

Множество всех функций, двоично-рационально сходящихся к вещественному числу x, обозначается CF_x.

Вещественное число x называется *CF* алгоритмическим [3], если CF_x содержит вычислимую функцию φ.

Вещественная функция f, заданная на отрезке $[a, b]$, называется *CF* алгоритмической функцией [3] на этом отрезке, если существует машина Тьюринга M с оракульной функцией такая, что для любого $x \in [a, b]$ и любой вычислимой функции $\varphi \in CF_x$ функция ψ, вычисляемая M с оракульной функцией φ, принадлежит $CF_{f(x)}$.

Определение 1. [2] *Число* $x \in \mathbb{R}$ *назовем* **FP//LINSPACE** *алгоритмическим вещественным числом, если существует функция* $\varphi \in CF_x$*, вычислимая в пределах* **FP//LINSPACE**.

Определение 2. [2] *Вещественную функцию f, заданную на отрезке* $[a, b]$*, назовем* **FP//LINSPACE** *алгоритмической вещественной функцией на отрезке* $[a, b]$*, если для любого* $x \in [a, b]$ *функция* ψ *(указанная в определении алгоритмической функции) из* $CF_{f(x)}$ *является* **FP//LINSPACE** *вычислимой.*

Вещественная **W**-функция Ламберта W определяется как решение функционального уравнения

$$x = W(x)\, e^{W(x)};$$

данное решение является функцией, обратной к функции $f(x) = x \cdot e^x$. Вещественная **W**-функция Ламберта W определена на интервале $[-e^{-1}, \infty)$ и имеет две ветви, верхнюю W_0 и нижнюю W_{-1}.

Так как ряд Тейлора функции W_0 в окрестности точки $x = 0$

$$W_0(x) = \sum_{k=1}^{\infty} a_k x^k = \sum_{k=1}^{\infty} \frac{(-k)^{(k-1)}}{k!} x^k$$

линейно сходится [2] на отрезке $[-(r \cdot e)^{-1}, (r \cdot e)^{-1}]$, где r – рациональное, $r > 1$, то воспользуемся алгоритмом $SeriesSum_1$ из [2] для вычисления данного ряда в пределах **FP//LINSPACE**. Алгоритм $SeriesSum_1$ позволяет вычислять с полиномиальным временем и линейной памятью на машине Тьюринга линейно сходящиеся степенные ряды вида

$$S(x) = \sum_{k=0}^{\infty} a_k x^k$$

$$\varrho \le \frac{3}{4} + 2^{-5}$$

на отрезке $[-\varrho, \varrho]$, при условии, что $|a_i| \le 1$, и коэффициенты a_k являются **FP//LINSPACE** вычислимыми. Так как все условия, при которых алгоритм $SeriesSum_1$ применим, выполняются для ряда Тейлора функции W_0, то верна следующая теорема.

Теорема 1. *Основная ветвь W_0 вещественной* **W***-функции Ламберта является* **FP//LINSPACE** *алгоритмической функцией на любом отрезке* $[-(r \cdot e)^{-1}, (r \cdot e)^{-1}]$ **FP//LINSPACE** *алгоритмических вещественных чисел, где r — рациональное, $r > 1$.*

Литература

1. *Дубинов А. Е., Дубинова И. Д., Сайков С. К.* **W**-функция Ламберта и ее применение в математических задачах физики. Саров: Изд-во ФГУП «РФЯЦ-ВНИИЭФ», 2006. 160 с.
2. *Яхонтов С. В., Косовский Н. К., Косовская Т. М.* Эффективные по времени и памяти алгоритмические приближения чисел и функций. СПб.: Изд. СПбГУ, 2012. 256 с.
3. *Ko K.* Complexity theory of real functions. Boston: Birkhauser, 1991. 310 p.
4. *Du D., Ko K.* Theory of Computational Complexity. New York: John Wiley & Sons, 2000. 491 p.

АЛГОРИТМИЧЕСКАЯ ВЕЩЕСТВЕННАЯ ФУНКЦИЯ, ЗАДАННАЯ НА ОТРЕЗКЕ [0, 1], КОТОРАЯ НЕ ЯВЛЯЕТСЯ ПОЛИНОМИАЛЬНО ВЫЧИСЛИМОЙ ПО ВРЕМЕНИ

С. В. Яхонтов

Каф. информатики, математико-механический факультет, Санкт-Петербургский государственный университет

E-mail: SergeyV.Yakhontov@gmail.com

В данной работе проводится построение алгоритмической вещественной функции $\mathcal{F}$, заданной на отрезке [0, 1] вещественной прямой, для которой существует экспоненциальный по времени алгоритм вычисления двоично-рациональных приближений на данном отрезке, но невозможен полиномиальный по времени алгоритм вычисления таких приближений на данном отрезке.

Для построения функции $\mathcal{F}$ используется модель алгоритмических вещественных чисел и функций, определенная в [1]. Основные сведения о классах вычислительной сложности можно взять из [2]; с помощью FPTIME и FEXPTIME обозначаются классы алгоритмов, полиномиальных по времени и экспоненциальных по времени соответственно.

Алгоритмические вещественные числа и функции

Последовательность $\varphi : \mathbb{N} \to \mathbf{D}$ двоично-рационально сходится к вещественному числу x, если для любого $n \in N$ выполняется $|\varphi(n) - x| \le 2^{-n}$ (здесь **D** — множество двоично-рациональных чисел). Множество всех функций, двоично-рационально сходящихся к вещественному числу x, обозначается CF_x.

Вещественное число x называется *CF* алгоритмическим, если CF_x содержит вычислимую функцию φ.

Вещественная функция f, заданная на отрезке $[a, b]$, называется *CF* алгоритмической вещественной функцией на этом отрезке, если существует машина Тьюринга M с оракульной функцией такая, что для любого $x \in [a, b]$ и любой вычислимой функции $\varphi \in CF_x$ функция ψ, вычисляемая M с оракульной функцией φ, принадлежит $CF_{f(x)}$.

Определение 1. [1, 3] Функция $f : [a, b] \to \mathbb{R}$ называется **FPTIME (FEXPTIME)** алгоритмической вещественной функцией на отрезке $[a, b]$, если для любого алгоритмического вещественного числа $x \in [a, b]$ функция

ψ (указанная в определении алгоритмической функции) из $CF_{f(x)}$ является **FPTIME (FEXPTIME)** вычислимой.

Модуль равномерной непрерывности

Понятие модуля равномерной непрерывности выражает вычислимую зависимость δ от ε в классическом определении равномерной непрерывности функции.

Определение 2. [1] *Пусть* $f:[a, b]\to\mathbb{R}$ *— равномерно непрерывная функция на* $[a, b]$. *Функция* $\omega:\mathbb{N}\to\mathbb{N}$ *называется модулем равномерной непрерывности функции f на* $[a, b]$, *если для всех* $n\in\mathbb{N}$ *и для всех* $x, y\in[a, b]$ *выполняется следующее:*

$$|x-y|\le 2^{-\omega(n)}\Rightarrow|f(x)-f(y)|\le 2^{-n}.$$

Следующая теорема утверждает, что для алгоритмических функций ω является вычислимой функцией от *n*.

Теорема 1. [1] *Если функция* $f:[a, b]\to\mathbb{R}$ *является CF алгоритмической на* $[a, b]$, *то f имеет вычислимый модуль равномерной непрерывности на* $[a, b]$.

Говорят, что функция *f* имеет полиномиальный модуль равномерной непрерывности на отрезке $[a, b]$ [1], если существует полином $\omega:\mathbb{N}\to\mathbb{N}$ такой, что для всех $n\in\mathbb{N}$ и для всех $x, y\in[a, b]$ из $|x-y|\le 2^{-\omega(n)}$ следует $|f(x)-f(y)|\le 2^{-n}$.

Теорема 2. [1] *Если f* — **FPTIME** *алгоритмическая вещественная функция, то f имеет полиномиальный модуль равномерной непрерывности, то есть существует полином* $\omega:\mathbb{N}\to\mathbb{N}$ *такой, что для всех* $n\in\mathbb{N}$ *и для всех* $x, y\in[a, b]$ *из* $|x-y|\le 2^{-\omega(n)}$ *следует* $|f(x)-f(y)|\le 2^{-n}$.

Построение алгоритмической вещественной функции $\mathcal{F}$

Построим алгоритмическую вещественную функцию $\mathcal{F}$, определенную на отрезке [0, 1], такую, что невозможен полиномиальный модуль равномерной непрерывности данной функции и, следовательно, данная функция не является полиномиально вычислимой по времени, и приведем идею экспоненциального по времени алгоритма вычисления данной функции.

Определим функцию $\mathcal{F}$ на отрезке [0, 1] следующим образом:

1) построим вещественную функцию $\beta_{p,q}(x)$ на отрезке [0, 1]:

$$\beta_{p,q}(x) = \begin{cases} 0 & \text{if} \quad 0 \le x \le \dfrac{p}{q}, \\ 2^q \left(x - \dfrac{p}{q} \right) & \text{if} \quad \dfrac{p}{q} \le x \le \dfrac{p + \dfrac{1}{2^q}}{q} \\ \dfrac{1}{q} & \text{if} \quad \dfrac{p + \dfrac{1}{2^q}}{q} \le x \le 1, \end{cases}$$

для натуральных p и q таких, что $q \ge 1$ и $p \in [0..(q-1)]$;

2) построим вещественную функцию $\alpha_q(x)$ на отрезке [0, 1]:

$$\alpha_q(x) = \sum_{p=0}^{q-1} \beta_{p,q}(x);$$

3)

$$\mathcal{F}(x) = \sum_{q=1}^{\infty} \frac{1}{q^2} \alpha_q(x). \qquad (1)$$

Функция $\mathcal{F}$ обладает следующими свойствами:

1) функция $\mathcal{F}$ монотонно возрастает на отрезке [0, 1];
2) функция $\mathcal{F}$ является равномерно непрерывной на отрезке [0, 1];
3) для вычисления функции с точностью 2^{-n} необходимо, чтобы выполнялось $\omega(n) \ge 2^{C \cdot n}$, где ω — функция из определения 2.

Из пункта 3) следует, что функция $\mathcal{F}$ не может иметь полиномиальный модуль равномерной непрерывности, и поэтому верны следующие теоремы.

Теорема 3. *Вещественная функция $\mathcal{F}$ не является полиномиально вычислимой по времени на отрезке* [*0, 1*].

Теорема 4. *Вещественная функция $\mathcal{F}$ не является полиномиально вычислимой по времени на отрезке в любой рациональной точке интервала* (*0, 1*).

Для вычисления функций $\beta_{p,q}$ можно воспользоваться «склейкой» функций [4].

Теорема 5. *Вещественная функция $\mathcal{F}$ является экспоненциально вычислимой по времени на отрезке* [*0, 1*].

Литература

1. *Ko K*. Complexity Theory of Real Functions. Boston: Birkhauser, 1991. 309 p.
2. *Du D., Ko K*. Theory of Computational Complexity. New York: John Wiley & Sons, 2000. 491 p.
3. *Яхонтов С. В., Косовский Н. К., Косовская Т. М*. Эффективные по времени и памяти алгоритмические приближения чисел и функций. СПб., 2012. 256 с.
4. *Кушнер Б. А*. Лекции по конструктивному математическому анализу. М.: Наука, 1973. 448 с.

Технологии и инструменты разработки программ и облачные вычисления

Сафонов
Владимир Олегович

д.т.н., профессор кафедры информатики СПбГУ
академик Американского биографического института (ABI)
член-корреспондент РАЕ
заслуженный деятель науки и образования РАЕ

АСПЕКТНО-ОРИЕНТИРОВАННЫЙ РЕФАКТОРИНГ CMS ORCHARD С ПОМОЩЬЮ СИСТЕМЫ ASPECT.NET

А. И. Михайлова

студ. кафедры параллельных алгоритмов СПбГУ

E-mail: anita.mikhailova@gmail.com

Аннотация. Система управления контентом Orchard предоставляет широкие возможности для создания и поддержки интернет-проектов, при этом, благодаря открытому исходному коду и продуманной архитектуре, позволяет дополнять систему новыми модулями, обеспечивая при этом легкое внедрение новой функциональности.

Введение

Архитектура Orchard имеет четко выраженную структуру. В ее основе лежат приложения Orchard.Core и Orchard.Framework, которые содержат фундаментальные классы, составляющие основу всего приложения и которые невозможно выделить в отдельные модули, а так же классы, необходимые для запуска приложения. Остальная функциональность разбита на независимые модули. Под модулем в данном случае понимается приложение, не зависящее ни от одного другого.

Но даже при такой архитектуре возникает сквозная функциональность (cross-cutting concern), которая появляется при обработки исключений, логгировании, авторизации и т. п.

В рамках данной работы был предложен подход к устранению сквозной функциональности в этом решении с помощью аспектно-ориентированного программирования и инструмента Aspect.NET. Также представлены возможные методы для рефакторинга, позволяющего извлечь сквозную функциональность.

Другой целью работы стояло повышение удобства сопровождения кода там, где это возможно. Поэтому рассматривались классы и методы, излишне насыщенные функциональностью. Модули, относящиеся к сторонним фреймворкам и библиотекам во внимание не принимались. К таким модулям, например, относится Lucene, используемый для поиска и индексации информации и TinyMSE — платформонезависимый HTML редактор.

Рефакторинги для извлечения сквозных функциональностей

Следующие виды рефакторингов могут быть применены для извлечения сквозных функциональностей:

1. Изменение абстрактного класса на интерфейс. Используется в случае когда необходимо наследование не только от абстрактного класса.

2. Вынесение функциональности в аспект. Применяется когда функциональность разбросана среди нескольких методов и классов переплетаясь с несвязным с данной функциональностью кодом.
3. Вынесение фрагмента кода в действие. Применяется, когда часть метода, связанная с функциональностью, перенесена в аспект.
4. Вынесение вложенного класса. Применяется в случае когда вложенный класс связан с функциональностью, извлеченной в аспект.
5. Внесение класса в аспект. Применяется когда небольшой автономный класс используется только кодом внутри аспекта.
6. Внесение интерфейса в аспект. Применяется когда один или несколько интерфейсов используется только кодом внутри аспекта.
7. Разделение абстрактного класса на аспект и интерфейс. Применяется в том случае если невозможно наследование другого класса, поскольку данный класс уже наследуют абстрактный класс, определяющий некоторые члены.

Рефакторинги, позволяющие обобщать код

1. Извлечение супераспекта. Применяется тогда, когда два и более аспекта содержат аналогичный друг другу код и функциональность
2. Внесение действия в супераспект. Используется когда все аспекты одними и теми же действиями действуют на разрез кода, объявленный в супераспекте.
3. Внесение разреза кода в супераспект. Применяется в случаях, когда в подаспектах объявляются идентичные разрезы кода.
4. Вынесение из супераспекта действия. Применяется если часть действия используется только некоторыми подаспектами или каждый подаспект требует различного действия.
5. Вынесение из супераспекта разреза кода. Применяется если разрез кода в супераспекте не используются некоторыми аспектами, которые его наследуют.

Каталог аспектов

В рамках данной работы предложены решения в виде аспектов Aspect. NET для решения следующих задач:

1. Извлечение методов, выполняющих качественно схожую работу. В данном случае выделяются методы, отвечающие за форматирование конкретных типов данных.
2. Обработка исключений. Приведен аспект, который обрабатывает возможные исключения в классе ObjectDumper.
3. Запись в журнал событий. Предложен аспект, в котором представлены методы, которые выполняют запись событий в журнал.

4. Логгирование. После вызова определенного метода, аспект вставляет код, отвечающий за логгирование информации, в данном случае, подробности вызова этого метода.

Для оценки полученных результатов использовалась интегральная метрика Maintainability Index, которая встроена в MS Visual Studio 2012 и определяет удобство сопровождения исходного кода отдельных классов и проекта в целом. Так, например, извлечение схожих по функциональности методов и обработки исключений позволило увеличить для них этот индекс на 12%. Для целевых классов другого приложения аспектное управление конфигурацией дало прирост этого индекса на 5%.

Заключение

Таким образом, применение АОП и Aspect.NET в разработке облачного приложения позволяет повысить легкость его сопровождения, увеличить скорость разработки и снизить затраты за счет повторного использования универсальных аспектов.

Литература

1. *Сафонов В. О*. Аспектно-ориентированное программирование. Издательский дом СПбГУ, 2011.
2. Сайт проекта CMS Orchard: http://www.orchardproject.net/
3. *Григорьев Д. А*. Разработка и практическое применение аспектно-ориентированной среды программирования для платформы .NET.
4. *Miguel P. Monteiro, Joao M. Fernandes*. Towards a Catalog of Aspect-Oriented Refactorings.

РЕАЛИЗАЦИЯ МЕХАНИЗМА ДОСТУПА К ДИНАМИЧЕСКОМУ КОНТЕКСТУ В ТОЧКАХ ПРИМЕНЕНИЯ АСПЕКТОВ ДЛЯ СИСТЕМЫ ASPECT.NET

Д. А. Григорьев
доцент, к. ф.-м. н., кафедра информатики, ММФ, СПбГУ
E-mail: gridmer@mail.ru

А. В. Григорьева
инженер, лаб. Java-технологии, ММФ, СПбГУ
E-mail: nastya001@mail.ru

В. О. Сафонов
д.т.н., проф., каф. информатики, ММФ, СПБГУ
E-mail: v_o_safonov@mail.ru

Аннотация. Аспектно-ориентированное программирование предоставляет аспектам возможность влиять на целевую программу, исследовать ее состояние и изменять при необходимости. В данной работе обсуждается реализация механизма доступа действий аспекта к динамическому контексту, который окружает точку внедрения. Приведены свойства доступа к контексту в системе Aspect.NET и рассмотрены вопросы их реализации.

Технология Aspect.NET

Aspect.NET — это инструментарий АОП для платформы .NET, разработанный в лаборатории Java-технологии мат.-мех. факультета СПбГУ под научным руководством профессора В. О. Сафонова [1]. С помощью Aspect.NET можно определять аспекты в отдельных библиотеках классов, а затем вплетать вызовы их методов в заданные места целевой сборки (joinpoints). Определения аспектов не зависят от конкретного языка, а разрабатывать их можно в любой среде разработки, поддерживающей платформу .NET.

Aspect.NET вплетает действия на уровне MSIL-инструкций после этапа компиляции целевой сборки, что влечет повышение производительности целевого приложения по сравнению с IOC-контейнерами. Более того, такая «пост-обработка» дает возможность выбирать конкретные места применения действий аспектов [2].

Аспектом может быть любой класс, производный от класса Aspect (предопределенного в библиотеке Aspect.NET). Реализация аспекта осуществляется статическими методами («действиями»), которые затем будут вставлены компоновщиком в заданные точки внедрения (joinpoints) в сборке целевого приложения. Требуемое множество точек внедрения задается в пользова-

тельском атрибуте AspectAction() своего действия аспекта. Любое действие можно вставлять перед (ключевое слово %before), после (%after) или вместо (%instead) вызова заданного целевого метода. Название целевого метода задается с помощью регулярных выражений относительно его сигнатуры.

Компоновщик аспектов — это отдельное консольное приложение. Его параметрами являются пути к сборкам аспектов и целевого приложения. При работе в MS Visual Studio требуется в свойствах проекта аспекта включить его вызов в события пост-компиляции (post-build events) [3].

В данной работе описывается подход к реализации вышеупомянутой АОП-функциональности в компоновщике аспектов.

Контекст точки внедрения

Внутри действий можно использовать свойства базового класса Aspect, предоставляющие доступ к контексту точки внедрения [2]:

- *SourceFileLine* — это строка, представляющая номер строчки в исходном коде файла, где расположен вызов целевого метода.
- *SourceFilePath* — это строка, представляющая путь к целевому файлу исходного кода, в котором расположен вызов целевого метода.
- *TargetObject* — это ссылка типа *System.Object* на объект в целевой программе, к которому применяется целевой метод в точке присоединения. Например, если вызов целевого метода в точке присоединения имеет вид p.M(), то *TargetObject* обозначает ссылку на объект p. Если целевой метод статический, *TargetObject* равно *null*.
- *TargetMemberInfo* — это ссылка типа *System.Reflection.MemberInfo* на объект, представляющий метаданные целевого метода.
- *WithinMethod* — это ссылка типа *System.Reflection.MethodBase* на метаданные, представляющие метод, в котором расположен вызов целевого метода.
- *WithinType* — это ссылка типа *System.Type* на определение класса, в котором расположен вызов целевого метода.
- *RetValue* — это ссылка типа *System.Object* на результат, возвращаемый целевым методом. Для всех действий кроме %after имеет значение *null*.
- *This* — это ссылка типа *System.Object* на объект в целевой программе, внутри метода которого оказалось вставлено данное действие аспекта. Например, если в верхнем примере p.M() содержится внутри метода объекта X, то X является ссылкой *This* для действия аспекта, применяемого к p.M().

В дополнение к свойствам контекста, компоновщик обеспечивает «захват» аргументов, передавая их от целевого метода в аргументы действия аспекта (см. Листинг 1).

Листинг 1

```
int Sum(int a, int b, int c){…} //Целевой метод

…
/* Действие аспекта с первым и третьим захваченными
у целевого метода аргументами*/

[AspectAction("%after %call *Sum(int,int,int) &&
                                        %args(arg[0],arg[2]))]
public static AspectAfterSum(int a, int c) {…}
```

Подробно синтаксис «захвата» аргументов рассмотрен в [4]. Нам же осталось упомянуть, что данный подход «точечной» передачи нужных аргументов выигрывает по накладным расходам, по сравнению со способом, когда весь набор аргументов целевого метода упаковывается в коллекцию объектов и действие аспекта вынуждено перебором выбирать нужные (см. напр. PostSharp [5]). С другой стороны, явное указание нужных аргументов усиливает связь между аспектом и конкретным целевым методом. Это может быть оправдано в задачах АОП-рефакторинга [6], но не при написании универсальных аспектов. Поэтому мы не исключаем в будущем реализацию подобной упаковки аргументов в массив объектов.

Реализация доступа к контексту точки внедрения

Основной критерий, которому должно удовлетворять проектное решение — это высокая производительность, иначе было бы трудно убедить рядовых разработчиков потратить свое время на освоение новой технологии.

Рассмотрим целевой код и действие аспекта на Листинге 2.

Листинг 2

```
Program p = new Program();
p.Method(); // Целевой код

[AspectDotNet.AspectAction("%before %call *Method")]
static public void BeforeAction() { // Действие аспекта
    Console.WriteLine(AspectDotNet.Aspect.SourceFilePath);
    Console.WriteLine(AspectDotNet.Aspect.This);
}
```

Компоновщик аспектов имеет дело с бинарными сборками и кодом MSIL. Сканируя целевую сборку, компоновщик находит точки внедрения и вставляет туда требуемый код загрузки контекста и вызов действия аспекта (см. листинг 3).

Листинг 3

```
…
IL_0001:  newobj     instance void Program::.ctor()
IL_0006:  stloc.0
//Аналог Program p = new Program()
IL_0011:  ldstr      «.../Program.cs»
IL_0016:  call       void
     AspectDotNet.Aspect::InternalSetSourceFilePath(string)
//Аналог Aspect.InternalSetSourceFilePath(«.../Program.cs»)
IL_001b:  ldloc.0
IL_001c:  call       void
    AspectDotNet.Aspect::InternalSetTargetObject(object)
//Аналог Aspect.InternalSetTargetObject(p)
IL_0021:  call       void MyAspectClass::BeforeAction()
//Аналог MyAspectClass.BeforeAction()
IL_0026:  ldloc.0
IL_0035:  callvirt   instance void
                          Program::Method()
//Аналог p.Method()
```

Как можно видеть, каждая необходимая переменная для контекста передается в специальные методы класса Aspect, начинающиеся с InternalSet…, которые сохраняют ее в соотв-х полях Aspect. Затем get-свойства их просто возвращают действию аспекта. Также видно, что отладочные данные *SourceFileLine* и *SourceFilePath* напрямую записываются в строковые ресурсы. Далее на строке 26 видно, что перед вызовом каждого нестатического метода, компилятор загружает в стек его целевой объект (инструкция ldloc.0). Поэтому, чтобы передать объект *TargetObject* в метод InternalSetTargetObject() мы можем просто продублировать инструкцию ldloc.0. Аналогично компилятор предваряет любой вызов метода текущего объекта инструкцией ldarg.0, которую мы дублируем и передаем в InternalSetThisObject(). Наконец, заметим, что компоновщик загружает только тот контекст, который реально используется в действии аспекта.

Рассмотрим загрузку свойств *TargetMemberInfo* и *WithinMethod*. Можно было бы воспользоваться стандартным способом и получить их через Type.GetMethod(). В этом случае, в процессе прогона .NET будет просматривать все методы заданного типа, сопоставлять их с условиями и определять нужный. Авторами был предложен другой вариант, когда нужный метод определяется в виде System.RuntimeMethodHandle на этапе внедрения аспектов, загружается в стек инструкцией ldtoken и передается в InternalSetTargetMethod() или InternalSetWithinMethod() (см. листинг 4). Внутри get-свойства *TargetMemberInfo* мы преобразуем RuntimeMethodHandle

в MethodInfo с помощью вызова MethodInfo.GetMethodFromHandle(aRuntime MethodHandle). Аналогичного механизма на C# нет, поэтому попытка декомпилировать результирующую сборку может провалиться.

Листинг 4

```
…
IL_0001:  ldtoken method instance void
DemoAspect.Program::Method1()
IL_0006:  call void
    Aspect::InternalSetWithinMethodHandle(valuetype
                    [mscorlib]System.RuntimeMethodHandle)
IL_000b:  ldtoken    method instance void
                           DemoAspect.Program::Method2()
IL_0010:  call void
   Aspect::InternalSetTargetMethodHandle(valuetype
                    [mscorlib]System.RuntimeMethodHandle)
```

Реализация свойства *WithinType* проблем не вызывает — это просто WithinMethod.DeclaringType.

Последнее свойство — RetValue. Результатом метода в .NET может быть либо ссылочный тип (object), либо простой тип, напр. int. Для простоты реализации мы приняли, что свойство RetValue имеет тип object. Тогда перед присваиванием мы должны упаковать (boxing) результат простого типа в object. После вызова целевого метода переменная с этим результатом будет находится на вершине стека, откуда мы ее можем взять, провести при необходимости упаковку (boxing), передать в метод InternalSetRetValue(object), распаковать обратно, положить на вершину стека и продолжить выполнение (см. листинг 5). Для результата ссылочного типа все аналогично, но без упаковки-распаковки.

Листинг 5

```
…
IL_0008:  callvirt   instance int32 Program::Method()
//Аналог вызова int i = p.Method();
IL_000d:  box        [mscorlib]System.Int32
IL_0012:  call        object Aspect::InternalSetRetValue(obj
ect)
IL_0017:  unbox.any  [mscorlib]System.Int32
IL_001c:  call       void MyAspectClass::AfterAction()
```

В конце рассмотрим реализацию захвата аргументов на примере из листинга 1. Допустим, что мы должны вызвать в целевом коде метод Sum(100,200,300) и применить после него действие AspectAfterSum, захва-

тив первый и третий аргументы. Компоновщик сгенерирует код, показанный на листинге 6.

Листинг 6

```
…
IL_0008:  ldc.i4.s   100
IL_000a:  ldc.i4     0xc8
IL_000f:  ldc.i4     0x12c
IL_0014:  callvirt   instance void
                        Program::Sum(int32,int32,int32)
//Аналог вызова Program.Sum(100, 200, 300)
IL_0019:  ldc.i4.s   100
IL_001b:  ldc.i4     0x12c
IL_0020:  call       void
                 MyAspectClass::AfterSumAction(int32, int32)
// Аналог вызова. MyAspectClass. AfterSumAction (100, 300)
```

Для передачи списка аргументов в целевой метод, компилятор C# загружает их поочередно в стек (8–14 строки). Теперь становится очевидно, что нам будет достаточно продублировать соответствующие ldc инструкции, что автоматически приведет к подстановке нужных аргументов целевого метода в параметры действия аспекта.

Литература

1. Сайт проекта Aspect.NET: http://aspectdotnet.org [дата просмотра: 24.04.2014].
2. *Григорьев Д. А.* Реализация и практическое применение аспектно-ориентированной среды программирования для Microsoft .NET // Научно-технические ведомости. СПб.: Изд-во СПбГПУ, 2009. №3. 225 с.
3. *Григорьев Д. А., Григорьева А. В., Сафонов В. О.* Бесшовная интеграция аспектов в облачные приложения на примере библиотеки Enterprise Library Integration Pack for Windows Azure и Aspect.NET // Компьютерные инструменты в образовании. СПб.: Изд-во АНО «КИО», 2012. №4. 5 стр.
4. *Сафонов В. О.* Аспектно-ориентированное программирование: Учебное пособие. СПб.: Изд-во СПбГУ. 2011. 28 с.
5. *Shearer P.* Parameter checking with PostSharp // http://www.peteonsoftware.com/index.php/2014/02/08/parameter-checking-with-postsharp/ [дата просмотра: 24.04.2014].
6. *Григорьева А. В.* Аспектно-ориентированный рефакторинг облачных приложений MS Azure с помощью системы Aspect.NET // Компьютерные инструменты в образовании. СПб.: Изд-во АНО «КИО», 2012. №1. 21 с.

СРАВНЕНИЕ АЛГОРИТМОВ ОБОБЩЕННОГО ВОСХОДЯЩЕГО И НИСХОДЯЩЕГО СИНТАКСИЧЕСКОГО АНАЛИЗА

А. К. Рагозина

студентка кафедры системного программирования СПБГУ

E-mail: ragozina.anastasiya@gmail.com

Аннотация. Синтаксические анализаторы используются во многих задачах, возникающих в процессе реинжиниринга, это приводит к необходимости создания их автоматически. Парсеры можно разделить на две категории — восходящие и нисходящие. Нисходящие синтаксические анализаторы популярны из-за своей простоты, хотя они позволяют обрабатывать очень узкий круг грамматик. Восходящие синтаксические анализаторы позволяют обрабатывать более широкий класс грамматик, но они более сложны для написания и отладки. Оба класса анализаторов страдают от необходимости приведения грамматики к однозначной форме. Обобщенные алгоритмы синтаксического анализа позволяю бороться с этим. В данной статье описывается процесс создания обобщенного табличного нисходящего анализатора и его сравнение с обобщенным восходящим анализатором.

Введение

Одной из важных задач, возникающих в процессе автоматического реинжиниринга программного обеспечения, является создание синтаксических анализаторов[1] языков программирования. Синтаксический анализ может использоваться для перевода исходной системы на другой язык программирования, анализа кода и других задач.

Синтаксические анализаторы можно разделить на два класса — нисходящие и восходящие. Нисходящие синтаксические анализаторы привлекательны тем, что их структура полностью соответствует структуре грамматики. К сожалению, класс грамматик, которые допускают нисходящие анализаторы является весьма ограниченным. На языки, которые могут быть обработаны LL-анализаторами[2] накладываются жёсткие ограничения: любая LL(k) — грамматика должна быть однозначной. Леворекурсивные грамматики не принадлежат классу LL(k) ни для какого k. Иногда удается преобразовать не LL-грамматику в эквивалентную ей LL-грамматику с помощью устранения левой рекурсии и факторизации. Однако проблема существования эквивалентной LL(k) — грамматики для произвольной не LL(k) — грамматики неразрешима[3]. Можно использовать backtracking[4] методы для расширения класса обрабатываемых языков, но даже это не поможет справиться с проблемой левой рекурсии. Восходящие LR-анализаторы[2] позволяют обрабатывать более широкий класс грамматик, но не имеют такой тесной связи

с грамматикой. Такие анализаторы позволяют работать с леворекурсивными грамматиками, но не могут обрабатывать скрытую левую рекурсию. Так же производительность таких анализаторов часто ниже, чем у парсеров, построенных с использованием нисходящих алгоритмов, а размер управляющих таблиц может экспоненциально зависеть от размера грамматики[5].

Для того, чтобы расширить класс языков, обрабатываемых нисходящими анализаторами, можно использовать обобщенный алгоритм разбора — Generalised LL (GLL)[6], который позволяет обрабатывать все контекстно-свободные грамматики и работает в худшем случае за кубическое время, а для LL-грамматик[2] за линейное. Синтаксические анализаторы, построенные с помощью такого алгоритма, позволяют бороться с проблемой левой рекурсии, как скрытой, так и обычной, значительно расширяя класс обрабатываемых нисходящими синтаксическими анализаторами языков. Это важно, потому что в процессе реинжиниринга грамматика часто подвергается изменениям, которые могут делать её неоднозначной и приводить к конфликтам, обработать которые возможно, используя восходящий алгоритм синтаксического анализа. Для обеспечения возможности работы с неоднозначными грамматиками в рамках проекта реализован GLR-генератор, порождающий восходящие парсеры с использованием алгоритма RNGLR[6], работающий со всеми контекстно-свободными грамматиками. Так же реализован алгоритм восстановления после ошибок, предоставляющий информацию необходимую для диагностики ошибок, и механизм, предоставляющий информацию о конфликтах.

Основная часть

Для автоматического создания синтаксических анализаторов существует несколько подходов: можно полностью генерировать весь код парсера по грамматике, а потом использовать его.

При другом подходе генерируется только дополнительная необходимая для работы синтаксического анализатора информация, которая используется интерпретатором, содержащим в себе основную логику алгоритма.

В статьях, описывающих алгоритм обобщенного нисходящего анализа[6] и пример практического создания парсера с использованием данного алгоритма[10], используется первый подход. По грамматике генерируются функции, с помощью которых происходит разбор и в результате строится дерево разбора. В рамках проекта было решено использовать второй подход из-за большей гибкости — возможности независимо реализовывать несколько интерпретаторов, что необходимо для того, чтобы получить в перспективе абстрактный синтаксический анализ, для которого нужны обычные таблицы и специальный интерпретатор.

В связи с решением использовать подход, отличный от описанного в статьях[10], в алгоритм были внесены некоторые изменения. Для осуществления выбора правила, по которому необходимо провести свертку, используется модифицированная LL-таблица. Отличие такой таблицы от обычной

LL-таблицы в том, что в каждой ячейке может содержаться несколько правил, по которым можно продолжать разбор на данном этапе работы синтаксического анализатора. Такая ситуация возникает из-за возможности наличия в грамматике неоднозначных правил.

Ещё одним значительным изменением, обусловленным отказом от генерации всего кода парсера, является сам процесс работы синтаксического анализатора. Вместо нескольких функций, соответствующих нетерминалам, используется пара взаимно рекурсивных функций: управляющая и обрабатывающая. Управляющая функция координирует работу интерпретатора, а для работы обрабатывающей функции выделено несколько основных ситуаций, которые возможны в процессе разбора.

Результаты

На данном этапе работы в качестве результатов получен генератор дополнительной информации, используемой для анализа. Эта информация содержит представление грамматики, функции для работы с ней и модифицированную LL-таблицу, о которой говорилось ранее. В процессе разработки происходит активная интеграция с уже существующим GLR-модулем, о котором упоминалось выше, и многие структуры переиспользуются. Например, структуры, позволяющие хранить грамматику в компактном виде, создавать деревья разбора и другое. Так же реализован распознаватель на основе алгоритма GLL.

Литература

1. *Alfred V. Aho* and *Ullman*. The Theory of Parsing, Translation and Compiling. Vol. 1: Parsing of Series in Automatic Computation. Prentice-Hall, 1972. Pp. 33–45.
2. *Alfred V. Aho, Monica S. Lam, Ravi Sethi,* and *Jeffrey D*. Ullman. Compilers: Principles, Techniques, and Tools. Pearson Education, Inc, 2006.
3. *Rosenkrantz D. J*. and *Stearns R. E*. Proceeding STOC»69 Proceedings of the first annual ACM symposium on Theory of computing. ACM, 1969. Pp. 165–180.
4. *Dick Grune* and *Ceriel J. H. Jacobs*. Parsing Techniques: A Practical Guide (Second Edition). Springer, 2008.
5. *Dick Grune, Kees van Reeuwijk, Henri E. Bal, Ceriel J. H. Jacobs,* and *Koen G. Langendoen*. Modern Compiler Design (Second Edition). John Wiley & Sons, 2010.
6. *Elizabeth Scott* and *Adrian Johnstone*. GLL Parsing. Electronic Notes in Theoretical Computer Science 253 (2010). Pp. 177–189.
7. Кириленко Я. А., Григорьев С. В., Авдюхин Д. А. Разработка синтаксических анализаторов в проетах по автоматизированному реинжинирингу информационных систем. Научно-технические ведомости СПбГПУ. Вып. 3 (174). 2013.
8. YaccConstructor home page https://code.google.com/p/recursive-ascent/wiki/YaccConstructor.
9. *Elizabeth Scott* and *Adrian Johnstone*. Right Nulled GLR Parsers.
10. *Elizabeth Scott* and *Adrian Johnstone*. Modelling GLL Parser Implementations. Engineering Lecture Notes in Computer Science. Vol. 6563. 2011. Pp. 42–61.

ПРИМЕРЫ БЕСШОВНОЙ ИНТЕГРАЦИИ ФУНКЦИОНАЛЬНЫХ БЛОКОВ MS ENTERPRISE LIBRARY С ИСПОЛЬЗОВАНИЕМ ASPECT.NET

М. Е. Стрельцова

студентка 3 курса кафедры информатики СПбГУ

E-mail: marinaarrow@yandex.ru

Аннотация. MS Enterprise Library — это продукт компании Microsoft для создания надежных и удобных приложений. Использование этой библиотеки так или иначе подразумевает модификацию исходного кода целевого приложения. Зачастую это может затруднить дальнейшее сопровождение кода. Данная статья описывает применение подходов аспектно-ориентированного программирования для бесшовной интеграции функциональных блоков MS Enterprise Library с использованием Aspect.NET.

Введение

Не секрет, что при разработке или сопровождении различных приложений, необходимо реализовывать ту или иную сквозную функциональность. Это не всегда простые задачи, которые необходимо выполнить в нескольких модулях. Так, например, решение задач ведения журнала системы или обработки исключений в различных точках различных модулей может осложнить восприятие целевого кода и затруднить дальнейшую разработку приложения.

Решением проблемы «запутанной функциональности» может стать выделение сквозной функциональности в один модуль, аспект [1] и дальнейшая бесшовная интеграция этих аспектов в целевой проект. Таким образом, при помощи бесшовной интеграции наш целевой код останется неизменным, а нужная функциональность будет реализована.

В апреле 2013 компания Microsoft выпустила продукт MS Enterprise Library 6 (EL) [2], предназначенный для реализации сквозной функциональности. Библиотека EL представляет собой набор функциональных блоков (подключаемые программные компоненты с возможностью многократного использования [3]).

Таким образом, было бы целесообразно внедрить вызовы методов классов, которые предоставляет нам библиотека EL, в наше целевое приложение с помощью Aspect.NET [4].

Хотелось бы отметить, что Aspect.NET далеко не единственное средство для внедрения сквозной функциональности. Для этих целей можно воспользоваться другим инструментом. Например, Autofac. Это некий IoC — контейнер, предназначенный для удобного внедрения зависимостей [5]. Он предоставляет возможность перехвата вызова компонент зарегистриро-

ванных в контейнере. Для этого необходимо пометить класс, вызовы методов которого нам надо перехватить, соответствующим атрибутом [Intercept («MyIntercept»)]. Затем необходимо переопределить метод Intercept, добавив нужную функциональность. Но у этого подхода есть существенный недостаток: перехватываются вызовы всех методов. В том случае, когда необходимо переопределить лишь некоторые методы, то необходимо выбрать один из двух вариантов. Первый способ — обязательно пометить соответствующим атрибутом те методы, которые обрабатывать не стоит. Но этот способ явно неудобный. Методов может быть очень много. Второй способ состоит в том, чтобы описать специальный класс, в котором будет определено несколько перехватчиков [6]. Но все равно придется фильтровать, для какого метода какой перехватчик использовать. Это накладывает некоторые сложности на разработку и уводит программиста в стороны от решения главных задач.

Более того, используя какие-либо IoC-контейнеры, необходимо в первую создавать экземпляры этих контейнеров и производить их настройку непосредственно в целевом коде приложения. В этом случае, говорить о бесшовной интеграции, к сожалению, не приходится.

К счастью, Aspect.NET может решить проблему бесшовной интеграции. Его преимущество, по сравнению с Autofac, состоит в том, что можно без труда перехватить какой-либо метод, какого-либо класса и добавить нужную функциональность. Также можно переопределять методы класса, создавая замещающий наследник. Для этого необходимо указать специальный атрибут для аспектного класса [ReplaceBaseClass] [7]. Более того, не требуется никакая модификация исходного кода, все аспекты находятся в отдельном проекте. Необходимо лишь добавить из него ссылку на целевой проект.

Постановка задачи

Итак, необходимо продемонстрировать примеры бесшовной интеграции функциональных блоков MS Enterprise Library с использованием Aspect. NET. Для рассмотрения были выбраны следующие функциональные блоки: Logging Block и Exception Handling Block.

Также для демонстрации использовались исходные коды приложений Hands-On Labs [8].

Примеры

1. Первым примером бесшовной интеграции является внедрение методов функционального блока, предназначенного для протоколирования.

До сих пор действие аспекта вставлялось перед или после вызова какого-то целевого метода [7]. Но что, если нам необходимо реализовать сквозную функциональность в середине целевого метода?

Рассмотрим исходный код приложения из Hands-On Labs для Logging Block (см. листинг 1).

Листинг 1

```
public string Calculate(int digits)
{
    …
    try
    {
        if (digits > 0)
        {
            using (Tracer trace =
                        traceMgr.StartTrace(Category.Trace))
            {
                 BuildPiString(pi, digits);
            }
        }
        result = pi.ToString();

        …
     } catch (Exception ex) {
        OnCalcException(new CalcExceptionEventArgs(ex));
    }
    return result;
}

protected StringBuilder BuildPiString(StringBuilder pi,
                                                    int digits)
```

В данном случае класс Calculator будет целевым. Мы видим, что здесь используется трассировщик Tracer, предназначенный для добавления дополнительного уникального идентификатора конкретным записям журнала [9]. Пусть действия по построению строки Pi вынесены в отдельный защищенный метод BuildPiString.

В таком случае, в замещающем наследнике можно переопределить метод BuildPiString, добавив блок using (см. листинг 2).

Листинг 2

```
[ReplaceBaseClass]
    public class TraceAspect : Calculator
    {
        private TraceManager traceMgr;
```

```
        public TraceAspect()
        {
            traceMgr = new TraceManager(Logger.Writer);
 }


protected StringBuilder BuildPiString(StringBuilder pi,
  int digits)
{
using ( Tracer trace=
traceMgr.StartTrace(Category.Trace))
    {
       base.BuildPiString(pi,digits);
    }
    return pi;
}
}
```

В итоге, в целевом проекте использовать Tracer нет необходимости.

2. Рассмотрим исходный код приложения из Hands-On Labs для Exception Handling (см. листинг 3).

Листинг 3

```
public class Puzzler : System.Windows.Forms.Form
{
     ...
     private void btnAddWord_Click(object sender,
     System.EventArgs e)
     {
         try
         {
             AddWord();

         } catch (Exception ex) {
            bool rethrow = ExceptionPolicy.
            HandleException(ex, «UI Policy»);

            if (rethrow)   throw;

            MessageBox.Show(string.Format(«Failed to add
            word {0}, please contact support.»,
```

```
            txtWordToCheck.Text));
        }

        private void AddWord() {}
            ...
    }
}
```

В этом примере мы видим, что ситуация осложнилась тем, что в блоке try есть вызов приватного метода AddWord (). В этой ситуации мы можем поступить следующим образом (см. листинг 4):

Листинг 4

```
[ReplaceBaseClass]
public partial class ExceptionAspect : Puzzler
{
    TextBox txtWordToCheck;

    public ExceptionAspect()
    {
        txtWordToCheck =
           (TextBox)this.GetType().BaseType.GetField
           («txtWordToCheck», BindingFlags.NonPublic |
               BindingFlags.Instance).GetValue(this);
    }

    private void btnAddWord_Click(object sender,
                                  System.EventArgs e)
    {
        try {
            base.AddWord();
        } catch (Exception ex) {
            bool rethrow =
                ExceptionPolicy.HandleException(ex,
                                        «UI Policy»);
            if(rethrow)   throw;

            MessageBox.Show(«Failed to add word
                             +txtWordToCheck.Text));
        }
    }
```

Создав замещающего наследника от целевого класса, мы переопределили метод и перенесли обработку исключения в аспектный класс. Таким образом, мы избавились от блока try и catch в исходном коде целевого приложения.

Заключение

Из представленных примеров видно, как вынести всю сквозную функциональность в отдельные аспекты, не изменяя при этом исходный код. Несомненно, это облегчает восприятие программного кода и его дальнейшее сопровождение. Ведь убрав из целевого кода всю второстепенную рутинную работу, мы сможем сконцентрироваться только на главных задачах приложения. А если нам понадобиться изменить способ протоколирования или стратегию обработки исключений, то нам не придется искать по всему проекту места, которые следует исправить. Каждая функциональность будет находится в своем аспекте и внести изменения надо будет только в одном месте.

Литература

1. *Сафонов В. О.* Аспектно-ориентированное программирование: Учебное пособие // СПБ: Издательство СПБГУ, 2011. 28 с.
2. Сайт проекта MS Enterprise Library // https://entlib.codeplex.com [дата просмотра 15.04.2014].
3. Руководство разработчика по Microsoft Enterprise Library 5.0 // http://msdn.microsoft.com/ru-ru/library/ff953181 (v=pandp.50).aspx [дата просмотра 15.04.2014] — Глава 1.
4. Сайт проекта Aspect.NET // http://aspectdotnet.org [дата просмотра: 15.04.2014].
5. Сайт проекта Autofac // http://autofac.org [дата просмотра: 15.04.2014].
6. Сайт проекта Castle // http://docs.castleproject.org/(X(1)S(ngjih2fkgagriwie2pssdtmz))/Tools.Use-proxy-generation-hooks-and-interceptor-selectors-for-fine-grained-control.ashx [дата просмотра: 15.04.2014].
7. *Григорьев Д. А., Григорьева А. В., Сафонов В. О.* Бесшовная интеграция аспектов в облачные приложения на примере библиотеки Enterprise Library Integration Pack for Windows Azure и Aspect.NET // Компьютерные инструменты в образовании // СПб.: Изд-во АНО «КИО», 2012. №4. 5 стр.
8. Сайт проекта Hands-On Labs for Enterprise Library 6.0 // http://www.microsoft.com/en-us/download/details.aspx?id=40286 [дата просмотра: 05.04.2014].
9. Руководство разработчика по Microsoft Enterprise Library 6.0 // http://msdn.microsoft.com/en-us/library/dn440724 (v=pandp.60).aspx [дата просмотра 15.04.2014] — Глава 6.

РЕАЛИЗАЦИЯ НАДСТРОЙКИ MS VS 2012 ДЛЯ ПОДДЕРЖКИ СИСТЕМЫ АСПЕКТНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ ASPECT.NET

М. А. Зотов
студент 3 курса ММФ СПбГУ
E-mail: zotov1994@mail.ru

Аннотация. В работе рассмотрены базовые понятия аспектно-ориентированного программирования (АОП) и системы Aspect.NET. Рассмотрен пример её использования на языке программирования C#. Дан обзор процесса создания надстроек для MS Visual Studio 2012, а также проанализированы способы для интеграции системы Aspect.NET с ней.

Введение

Аспектно-ориентированное программирование — парадигма программирования, в основе которой лежит идея выделения сквозной функциональности в отдельные модули, которые называются аспектами. В таких модулях объединяются все действия, выполняемые в определенных точках программы, и правила их внедрения в целевой код бизнес-логики. Под бизнес-логикой понимается исходный код, который описывает сущности предметной области и работу с ними в ее рамках [1].

Аспекты применяются к бизнес-логике, т. е. к основной программе, с помощь правил внедрения (weaving rules). На выходе получается система, которая решает поставленную задачу, причем вся сквозная функциональность вынесена в отдельный блок, код бизнес-логики — в другой. Сквозная функциональность представляет из себя функциональность, реализация которой рассредоточена по нескольким уровням системы [2].

Aspect.NET

Aspect.NET — это инструмент АОП для платформы.NET, разработанный под научным руководством профессора В. О. Сафонова [3].

При использовании Aspect.NET, решение (solution) в MS Visual Studio 2012 (VS) представляет из себя набор проектов, один из которых — код бизнес-логики, другой — код аспектов [4].

Для того чтобы воспользоваться возможностями Aspect.NET, достаточно написать интересующий код бизнес-логики, не учитывая сквозную функциональность. После этого останется лишь написать код аспектов и правила

их внедрения в бизнес-логику, а затем применить Aspect.NET. Результирующая сборка будет обладать нужной функциональностью.

Для платформы.NET разработано множество библиотек, каркасов и инструментов, таких как PostSharp [5], MS Code Contracts [6], MS EL [7] и пр., которые предоставляют полезные технические сервисы и службы. Однако их интеграция в целевое приложение приводит к изменениям в исходном коде, что может быть нежелательно при быстром прототипировании, когда необходимо удалить или заменить данную подсистему. В Aspect.NET код аспектов не изменяет исходный код [8].

Процесс применения аспектов к целевому коду можно классифицировать (по способу внедрения).

- Динамический. При таком способе внедрения запуск аспекта (внедрение) осуществляется по выполнению условий, т. е. прямо во время выполнения программы. Это означает то, что внедрение будет производиться тогда, когда оно понадобится, т. е. когда произойдет непосредственное обращение к функциям, к которым привязаны аспекты.
- В процессе загрузки приложения. Внедрение аспектов происходит при загрузке приложения.
- Статический. Аспекты и основной код сливаются на уровне сборки. При запуске приложения, аспекты будут уже применены к исходному коду и при обращении к целевым методам аспектов, они сразу будут запускаться [8].

Система Aspect.NET относится к последнему виду классификации. Благодаря этому все накладные расходы на использование аспектов минимизируются. Это дает разработчику большие возможности и избавляет его от обязанности заботиться о проблемах производительности при использовании Aspect.NET.

Пример использования Aspect.NET

Понять суть АОП легче всего на примере.

Действия аспекта могут быть использованы в конкретных точках внедрения целевого приложения, местонахождение которых задается с помощью текстовой маски или регулярного выражения. Например, правило «`%before %call *Car.MoveTo`» означет: вставить действие аспекта перед вызовом метода MoveTo класса Car. Место внедрения действия аспекта относительно целевого метода может быть задано следующим образом: «вызвать перед» (%before), «вставить вместо» (%instead), «вызвать после» (%after). Вызывая действия аспекта в определенных точках, целевое приложение реализует функциональность «сквозных» компонент [8].

Структура класса Car

```
public class Car
{
    private string curPos;
    public string CurrentPosition
    {
        get { return curPos; }
    }

    public Car(string curPos) { this.curPos = curPos; }
    public void MoveTo(string place)
    {
        Console.WriteLine(«Moving from '»+curPos+»' to
                                                '»+place+»'»);
        curPos = place;
    }
}
```

Структура класса-аспекта

```
public class SmartCar : Aspect
{
    [AspectAction(«%before %call *Car.MoveTo»)]
    static public void WarmEngineAction()
    {
      Console.WriteLine(«Warming the engine before starting»);
    }

    [AspectAction(«%after %call *Car.MoveTo»)]
    static public void TurnOffEngineAction()
    {
        Console.WriteLine(«Turn off the engine. Bye! «);
    }
}
```

Структура Main

```
static void Main(string[] args)
{
    new Car(«Home»).MoveTo(«Work»);
    Console.ReadKey();
}
```

Результат работы программы до построения проекта с аспектами.

После построения проекта с аспектами.

Как можно видеть, трассировка программы перед внедрением аспектов и после — различается.

Интересное поведение программы можно наблюдать в ходе ее пошагового выполнения (которое очень полезно при работе с АО-подходом). Запустим ее в режиме отладки и поставим breakpoint (точку останова) на строчке `new Car («Home»). MoveTo («Work»).`

Применим аспекты. При отладочном шаге с заходом, программа в проект аспекта на метод `WarmEngineAction`. Переход произойдет из проекта, который *никак* не связан с проектом аспектов, в проект, в котором содержится вся сквозная функциональность. Это очень полезное свойство Aspect.NET, которое помогает программисту быстрее и легче воспринимать взаимодействие и влияние проекта аспектов на проект бизнес-логики.

Результаты сравнения вызовов методов в результате выполнения программы, представлены в Таблице 1.

Таблица 1

Сравнение пошагового выполнения проекта без использования аспектов и проекта, в который внедрены аспекты

Без аспектов	С аспектами
`new Car(«Home»).MoveTo («Work»);`	`new Car(«Home»).MoveTo(«Work»);`
	`static public void WarmEngineAction()`
`public Car(string curPos)`	`public Car(string curPos)`
`public void MoveTo(string place)`	`public void MoveTo(string place)`
	`static public void TurnOffEngineAction()`

Улучшение системы

Для использования готового проекта (с внедренными аспектами), необходимо произвести 4 действия.

1. Прописать в post-build events проекта аспектов ссылки на целевые директории (и их проекты), директории библиотек Weaver»а — подсистемы применения аспектов в целевой код.
2. Установить зависимость проекта аспектов от проекта бизнес-логики.
3. Собрать проект аспектов.
4. Собрать проект бизнес-логики и запустить его.

Если целевое приложение действительно большое и требуется постоянная его отладка, внесение изменений в код аспектов и код самого приложения, то эти действия замедляют работу разработчика. Ведь если не собирать каждый раз проект аспектов, то они не применятся, ибо внесения вносятся в файлы сборки. А файл сборки меняется только при компиляции проекта. Свойства проекта и зависимость устанавливается единожды — но все равно устанавливается, причем вручную.

Есть желание автоматизировать этот процесс, чтобы пользователю оставалось нажимать только одну кнопку, по которой произойдет внедрение аспектов в целевой проект, установится зависимость проектов, запустится целевое приложение.

Вопрос с автоматическим выставлением свойств решён. Для этого достаточно рассмотреть конфигурационный файл проекта (*.csproj) и добавить нужные пользователю строки события после построения в секцию <PostBuildEvent>. Что же касается последовательной сборки *двух* проектов *одним* кликом — точнее, о последовательности сборки проектов и установки зависимостей — информация об этом вносится в другой сериализованный файл (*.suo). Работа с ним в значительной (и даже критической) степени усложняется, увы, пока что не удалось найти способа, который бы позволил решить проблемы зависимости и последовательной сборки.

Современные средства .NET позволяют расширять возможности VS, одним из таких средств является плагин, позволяющий встроить в VS свою кнопку и привязать к ней логику действий. Однако отладка такого приложения очень сложна. Плюс этого подхода в том, что имеется превосходная интеграция с VS и визуализация (Рисунок 1), минус — долгая компиляция, запутанная структура относительной адресации во время отладки.

Рис. 1. Расширение Visual Studio 2012

На данном этапе, с помощью bat-файлов, решена проблема установки свойств проекта. Такой файл запускает исполняемый файл, который добавляет необходимые строчки в файл проекта.

Интересен *результат* работы проекта-расширения, сам же проект представляет меньший интерес. Поставим breakpoint на строчке, в которой указана команда запуска bat-файла и запустим проект-расширение. После открытия тестового проекта — в котором необходимо выставить нужные свойства с помощью расширения — нажмем на кнопку ASPECT и в выпадающем меню выберем Insert Post-Build Events (Рисунок 1). Отладчик остановится на строчке с breakpoint»ом (Рисунок 2).

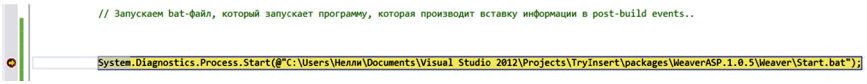


Рис. 2. Запуск bat-файла

Продолжим выполнение программы. Появится окно, в котором будет выведен путь запускаемого файла, который добавит нужные строчки в файл проекта (Рисунок 3).

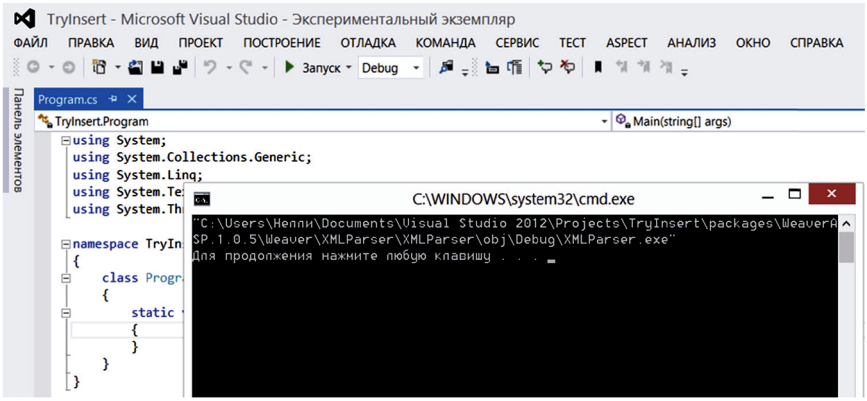


Рис. 3. Исполняемый файл XMLParser.exe

После этого будет выведена информация об успешном добавлении изменений (Рисунок 4). Затем будет предложено перезагрузить конфигурационный файл проекта (Рисунок 5).

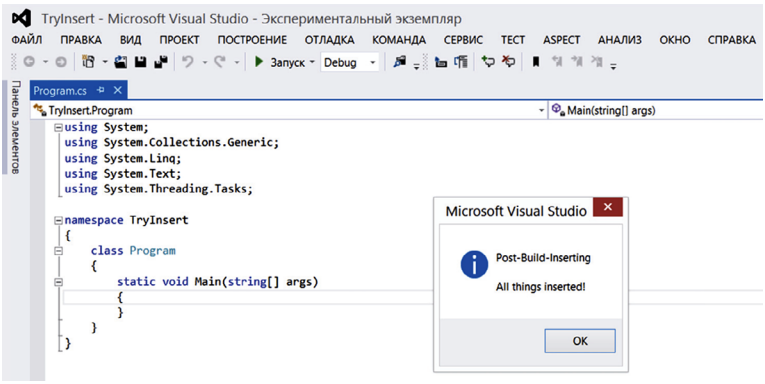


Рис. 4. Информационное окно

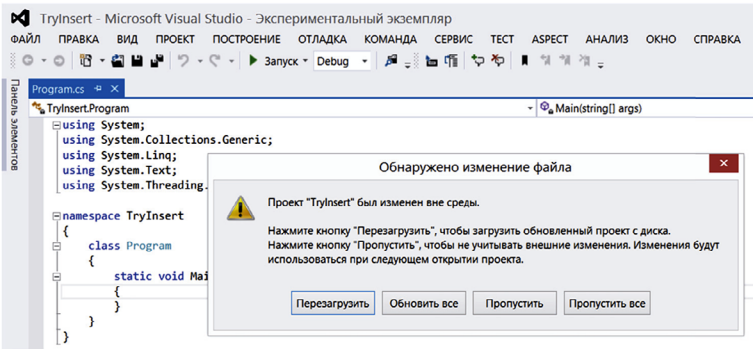


Рис. 5. Обновление конфигурации проекта

После нажатия клавиши «Перезапустить» или «Обновить все», в свойствах проекта можно будет наблюдать все необходимые изменения (Рисунок 6).

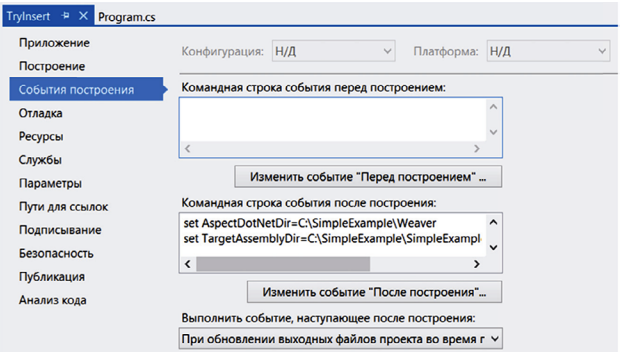


Рис. 6. Изменения применены

Заключение

Aspect.NET — мощная система для работы с АОП, которая предоставляет разработчику широкий спектр инструментов, с помощью которых он может вносить изменения в свой проект, не опасаясь снижения производительности и не внося в него изменения. Это позволяет разработчикам поддерживать свою конкурентоспособность и востребованность; в то же время продукт, в котором используется АОП и система Aspect.NET, всегда будет структурированнее, чем любые другие проекты без выделения сквозной функциональности, а значит, код продукта будет читаемый и легко воспринимаемый.

Литература

1. *Эспозито Д.* Аспектно-ориентированное программирование, перехват и Unity 2.0 // MSDN Magazine, 12.2010 // Режим доступа [проверено 22.04.2014]: http://msdn.microsoft.com/ru-ru/magazine/gg490353.aspx
2. Сайт проф. Эрика Боддена // Режим доступа [проверено 22.04.2014]: http://www.bodden.de/tools/aop-dot-net/
3. Сайт проекта Aspect.NET // http://aspectdotnet.org/
4. *Сафонов В. О.* Аспектно-ориентированное программирование: Учебное пособие. СПб.: Изд-во СПбГУ, 2011. 28 с. 5. Сайт проекта PostSharp // Режим доступа [проверено 22.04.2014]: http://www.sharpcrafters.com/
6. Сайт проекта Code Contracts // Режим доступа [проверено 22.04.2014]: http://msdn.microsoft.com/en-us/devlabs/dd491992
7. Сайт проекта MS Enterprise Library // Режим доступа [проверено 22.04.2014]: http://wag.codeplex.com/
8. *Григорьев Д. А.* Реализация и практическое применение аспектно-ориентированной среды программирования для Microsoft .NET // Научно-технические ведомости. СПб.: Изд-во СПбГПУ, 2009. № 3. 225 с. 9. Сайт лаборатории Java технологии // Режим доступа [проверено 22.04.2014]: http://polyhimnie.math.spbu.ru/jtl/AspectNET/Опроекте/tabid/98/Default.aspx

Адаптивное управление и распознавание образов в условиях неопределенностей

Граничин
Олег Николаевич

д.ф.-м.н.
профессор кафедры системного программирования СПбГУ
заведующий лабораторией
стохастических вычислительных систем НИИИТ СПбГУ

ПОИСКОВОЙ АЛГОРИТМ СТОХАСТИЧЕСКОЙ АППРОКСИМАЦИИ В ЗАДАЧЕ БАЛАНСИРОВКИ ЗАГРУЗКИ ПРИ НЕИЗВЕСТНЫХ, НО ОГРАНИЧЕННЫХ ВОЗМУЩЕНИЯХ НА ВХОДЕ

В. А. Ерофеева

студентка Кафедры системного программирования СПбГУ

E-mail: vicki.ultramarine@gmail.com

Аннотация. Использование стохастической оптимизации дает возможность по-новому решать проблемы, возникающие в процессе управления техническими системами. Рандомизированные алгоритмы стохастической аппроксимации позволяют решать классы сложных задач различной размерности, при этом они имеют простую форму, дают адекватные оценки искомых параметров при наблюдениях с помехами и адаптируются к изменениям внешней среды.

В данной статье мы рассмотрим возможности применения стохастической аппроксимации для решения задачи балансировки загрузки вычислительных узлов.

Введение

Стохастическая аппроксимация была предложена Роббинсом и Монро [1] и впоследствии была оптимизирована Кифером и Вулфовицем (KW) [2]. Спалл в [3] предложил поисковый алгоритм стохастической аппроксимации (Simultaneous Perturbation Stochastic Approximation, SPSA), использующий только два наблюдения на каждой итерации. Он использовал формулу, полученную для асимптотической дисперсии ошибки и сходную характеристику KW-процедуры, чтобы сравнить производительность данных алгоритмов [4]. В [5] так же рассмотрено сравнение KW-процедуры с алгоритмом SPSA. Позднее, Чен, Дункан и Пасик-Дункан [6] улучшили алгоритм SPSA таким образом, чтобы он больше подходил для систем реального времени.

В данной работе рассматривается применение алгоритма SPSA в задачах балансировки загрузки вычислительных узлов. В этом случае важную роль играют произвольные помехи. В [7–10] рассмотрена модификация алгоритма SPSA для работы в условиях неопределенности.

Применение рандомизированных алгоритмов в задачах балансировки загрузки рассмотрено в [11] для централизованного случая, а в [12–13] для децентрализованного.

Адаптивная стратегия управления техническими системами с постоянными параметрами обосновывается в [14–16].

1. Оптимальное значение функционала среднего риска

Пусть $f(\theta, w)$: $R_d \times W \to R$, $W \subset R_r$ — дифференцируемая по θ функция. Предположим, что $x_1, x_2, \ldots$ — последовательность экспериментальных параметров, в которых значения $y_1, y_2, \ldots$ функции $f(\cdot,\cdot)$ доступны для наблюдения в момент времени $t = 1, 2, \ldots$, с добавлением внешнего шума v_t:

$$y_t = f(x_t, w_t) + v_t, \tag{1}$$

где $w_t \in W$, $t = 1, 2, \ldots$ — неконтролируемые случайные последовательности (векторы). Используя наблюдения $y_1, y_2, \ldots$ построим последовательность оценок $\hat{\theta}$ неизвестного вектора θ^*, минимизируя при этом функционал среднего риска

$$F(\theta) = E_w f(\theta, w) \to \min_{\theta}. \tag{2}$$

Здесь и далее E_w — условное математическое ожидание относительно w.

Минимизация функции $F(\theta)$ обычно рассматривается относительно простейшей модели наблюдений

$$y_t = F(x_t) + v_t.$$

В формулировке (1) обобщение имеет несколько причин. Во-первых, учитывается случай возникновения мультипликативных возмущений в наблюдениях

$$y_t = w_t \bar{f}(x_t) + v_t.$$

Во-вторых, это позволяет разделить помехи в наблюдениях на «хорошие» $\{w_t\}$ и произвольные помехи $\{v_t\}$. Соответственно, в этом разделении нет необходимости, когда мы можем предположить, что $\{v_t\}$ — случайный одинаково распределенный вектор.

Рассмотрим более общую нестационарную задачу:

$$F_t(\theta) = E_{\xi_t, w} f_{\xi_t}(\theta, w) \to \min_{\theta}, \tag{3}$$

где $\{f_{\xi_t}(\theta, w)\}_{\xi \in \Xi}$ — семейство дифференцируемых по θ функций $f_{\xi_t}(\theta, w)$: $R^d \times W \to R$, $W \subset R^r$. Для выбранной последовательности $x_1, x_2, \ldots$ мы можем, с добавлением помех v_t, рассмотреть

$$y_t = f_{\xi_t}(x_t, w_t) + v_t, \tag{4}$$

где $t = 1, 2, \ldots$, $\{\xi_t\}$ — неконтролируемая последовательность: $\xi_t \in \Xi$, $w_t \in \mathrm{W}$ — неконтролируемые случайные переменные.

Проблема (2) является частным случаем (3), когда $\theta_t \equiv \theta^*$ и $\Xi = \{\xi 1\}$.

Сформулируем главные условия, касающиеся функций $F_t(x)$ и $f_{\xi_t}(x, w)$:

1. Функция $F_t(\cdot)$ имеет набор уникальных значений минимума θ_t и $x-\theta_t$, $E_{\xi_t, w}\nabla f_{\xi_t}(x, w) \geq \mu x-\theta_t^2$, $\forall x \in R^d$ с константой $\mu > 0$. Здесь и далее $\langle \cdot, \cdot \rangle$ — скалярное произведение двух векторов.
2. Вектор-градиент ∇f_{ξ_t} равномерно ограничен в среднеквадратичном смысле в минимальных точках: $\forall x', x'' \in R^d$ с константой $M > \mu$.

2. Пробное возмущение, оценка алгоритма, условия

Пусть Δ_n, $n = 1, 2, \ldots$ — последовательность наблюдений независимых случайных переменных в R^d, называемых пробным одновременным возмущением с функциями распределения $P_n(\cdot)$, и пусть $K_n(\cdot): R^d \to R^d$, $n = 1, 2, \ldots$ — некоторые векторные функции.

Возьмем фиксированный неслучайный начальный вектор $\hat{\theta}_0 \in R^d$ и положительные константы α, β. Рассмотрим алгоритм с двумя наблюдениями для построения последовательности точек наблюдений $\{x_t\}$ и оценок $\{\hat{\theta}_n\}$:

$$\begin{cases} x_{2n} = \hat{\theta}_{n-1} + \beta\Delta_n, \; x_{2n-1} = \hat{\theta}_{n-1}, \\ \hat{\theta}_n = \hat{\theta}_{n-1} - \dfrac{\alpha}{\beta} K_n(\Delta_n)(y_{2n} - y_{2n-1}). \end{cases} \tag{5}$$

Предположим, что F_n — σ-алгебра вероятностных событий, генерируемых случайными векторами $w_1, w_2, .., w_{2n}$ и элементами $\xi_1, \xi_2, ..., \xi_{2n}$, $\overline{v}_1, \overline{v}_2, ..., \overline{v}_{2n}$, если они случайны.

Предположим, что выполнены следующие условия:

1. Для любых $n = 1, 2, \ldots$,
   a. Δ_n, w_{2n-1}, w_{2n} и ξ_{2n-1}, ξ_{2n} (если они случайны) не зависят от σ-алгебры F_{n-1};
   b. Случайные векторы Δ_n и элементы (w_{2n-1}, w_{2n}) независимы;
   c. Если $\overline{v}_n = \overline{v}_{2n} - \overline{v}_{2n-1}$ случайный вектор, то $\overline{v}_2$ и Δ_n независимы.

Векторы Δ_n ограничены: $\|\Delta_n\| \leq c_\Delta < \infty$, и вектор функций $K_n(\cdot)$ наряду с одновременно возмущаемой симметричной функцией распределения $P_n(\cdot)$ удовлетворяет условиям:

$$\int K_n(x) P_n(dx) = 0, \; \int K_n(x) x^T P_n(dx) = I,$$
$$K_n(x)^2 \leq k < \infty, \; n = 1, 2, \ldots . \tag{6}$$

3. Верхняя граница оценивания

Последовательность оценок $\hat{\theta}_n$ имеет верхнюю границу $\bar{L} > 0$, если $\forall \varepsilon > 0 \exists N$, и для $\forall n > N$:

$$\sqrt{E\,\|\hat{\theta}_n - \theta_{2n}\|} \le \bar{L} + \varepsilon.$$

Сформулируем условия, касающиеся возмущений:

1. Смещение ограничено $\|\theta_t - \theta_{t-1}\| \le \delta\theta < \infty$ и для любой произвольной точки x: $E_{F_{n-1}}\varphi_{2n}(x)^2 \le a\,\|x - \theta_{2n-2}\|^2 + b$, где $\varphi_t(x) = f_{\xi_t}(x, w_t) - f_{\xi_{t-1}}(x, w_{t-1})$;
2. Наблюдение шума $\{v_n\}$ удовлетворяет: $|v_{2n} - v_{2n-1}| \le c_v < \infty$.

Следующая теорема показывает асимптотическую эффективность верхней границы оценки по алгоритму (5).

Теорема 1. Пусть все условия выполнены, обозначим

$$k = \mu - k\alpha\left(c_\Delta^2 M^2 - \frac{2\alpha}{\beta^2}\right),$$

$$l = 2\beta M c_\Delta^2 + 2k\alpha\left(\frac{c_v^2 + 2b}{\beta^2} + c_\Delta^2\left(3M^2 c_\Delta^2 + 2g\right)\right).$$

Если константа α достаточно мала: $2k\alpha < 1$ и

$$\alpha < \frac{\mu}{k c_\Delta^2 M_2}, \tag{7}$$

то последовательность оценок, обеспечиваемая алгоритмом (5) имеет ассимптотически эффективную верхнюю границу, которая равна

$$\bar{L} \le 2\left(\frac{\delta_0}{k\alpha} + \delta_0\right)^2 - 2\delta_0^2 + \frac{l}{k}. \tag{8}$$

4. Балансировка загрузки вычислительных узлов

Пусть вычислительная сеть имеет m узлов. На каждой итерации *i* система получает пакет заданий известного размера z_i. Предположим, что полученное задание можно разбить на m подзадач u_j, $j = 1, \ldots, m$,

$$\sum_{j=1}^{m} u_j = z_i,$$

для каждого узла, тогда время выполнения подзадачи на узле *j* определяется как $t_j(u_j) = x_j u_j$, где $x_j \in R$ — значение, обратное величине производительности узла *j*.

Необходимо минимизировать время выполнения полученного пакета заданий z_i:

$$T(u) = \|t(u)\|_\infty = \max_{j=1\ldots m} t_j(u_j) \to \min_u, \tag{9}$$

где

$$u = \left(u_1, u_2, \ldots, u_m\right)^T, \quad t = \left(t_1(u_1), t_2(u_2), \ldots, t_m(u_m)\right)^T.$$

Когда известны производительности узлов, наилучшей стратегией управления является пропорциональное распределение задач, при котором

$$x_1 u_1 = x_2 u_2 = \ldots = x_m u_m.$$

Стратегия, в которой $u = U(\theta, z_i)$ называется «балансировка загрузки». Здесь мы ввели обозначение $\theta = (x_1, x_2, \ldots, x_m)^T$.

В действительности производительности узлов неизвестны. Более того, они могут быть искажены из-за выполнения предыдущих задач $\theta_i = \theta + w_i$, или изменяться во времени $\theta_i = \theta_{i-1} + \xi_i$, где $w_i, \xi \in R_m$ — векторы независимых случайных переменных.

В таком случае, на каждой итерации i можно воспользоваться приближенными оценками $\hat{\theta}_i$, характеризующими производительности узлов, которые определяются таким образом, что $\|\hat{\theta}_n - \theta_i\|^2 \to \min$ по некоторым обоснованным соображениям, и вычислять u_i на итерации i как $u_i = U(\hat{\theta}_i, z_i)$.

Один из рациональных способов оценки качества:

$$F(\hat{\theta}_i) = E\left(\frac{\hat{\theta}_i - \theta_i, u_i}{z_i}\right)^2 = E\left(1 - \frac{t_{i1}}{z_i}\right)^2 \to \min_{\hat{\theta}_i}, \tag{10}$$

где $t_1 = \sum_{i=1}^{m} |t_i|$.

Для определения оптимального вектора θ^* воспользуемся поисковым алгоритмом стохастической аппроксимации (Simultaneous Perturbation Stochastic Approximation, SPSA).

Алгоритм SPSA

```
Require: α > 0, β > 0, θ̂0 ∈ Rm
    procedure SPSA(αc, βc, N)
        α ← αc
        β ← βc
        θ̂0 ← rand()
        for i ← 1 to N do
            n←n+1
            zn ← rand()
            deltan ← randb()
```

```
        u2n-1 ← U( $\hat{\theta}_{n-1}$ , z2n-1)
        t2n-1 ← fork(u2n-1)
        y2n-1 ← $\left(1-\frac{t_{2n-11}}{z_{2n-1}}\right)^2$
        u2n ← U( $\hat{\theta}_{n-1}$ +βdeltan, z2n)
        t2n ← fork(u2n)
        y2n ← $\left(1-\frac{t_{2n1}}{z_{2n}}\right)^2$
        $\hat{\theta}_n$ ← $\hat{\theta}_{n-1}-\frac{\alpha}{\beta}$ deltan(y2n - y2n-1)
    end for
end procedure
```

5. Моделирование работы алгоритма

В текущем разделе продемонстрируем имитацию работы алгоритма, рассмотренного ранее. Предположим, что на итерации i пакет заданий $z_i = (t_i, d_i)$, где t_i — время поступления пакета на управляющий узел, d_i — время, необ-

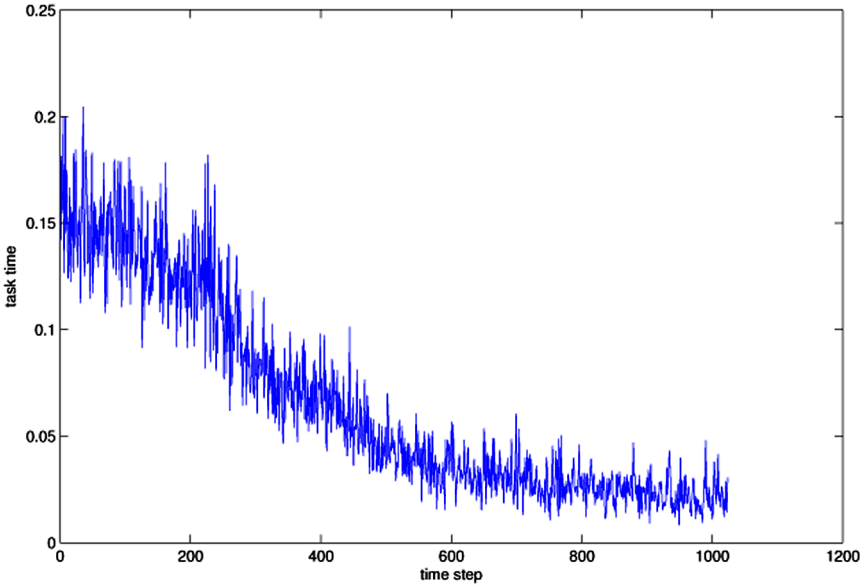


Рис. 1. Адаптация алгоритма к оптимальному параметру для 100 вычислительных узлов

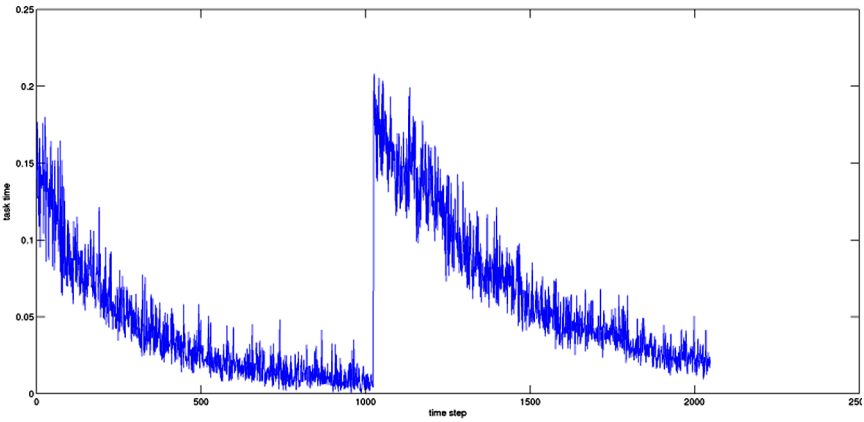


Рис. 2. Поведение алгоритма при меняющихся условиях

ходимое на выполнение всех заданий. Так как заранее неизвестно время, за которое выполнятся задания из пришедшего пакета, то будем считать, что d_i распределенная по Пуассону величина, зависящая от произведения mp_i, где p_i — среднее значение производительности узлов m на итерации i.

На рисунке 1 представлена зависимость времени выполнения одного пакета заданий z_i от временных тактов. Как можно заметить, с течением времени алгоритм сходится к оптимальному значению для текущих условий работы.

Так как заданные входные условия могут измениться в процессе работы алгоритма, смоделируем его работу в условиях изменения количества вычислительных узлов. Результат данного исследования представлен на рисунке 2.

Заключение

Применение рандомизированных алгоритмов позволяет достичь наилучшего результата, используя при этом простые методы решения задач. Применительно к проблеме балансировки нагрузки, появляется возможность рационального использования ресурсов на основе использования адаптирующихся алгоритмов. Как видно из раздела моделирования, предложенный алгоритм подстраивается под меняющиеся условия среды и не теряет при этом своих характеристик. В последующем планируется улучшение механизма выбора параметров α и β.

Литература

1. *H. Robbins* and *S. Monro*, “A stochastic approximation method,” The Annals of Mathematical Statistics, pp. 400–407, 1951.
2. *J. Kiefer* and *J. Wolfowitz*, “Stochastic estimation of the maximum of a regression function,” The Annals of Mathematical Statistics, vol. 23, no. 3, pp. 462–466, 1952.

3. *J. C. Spall*, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," IEEE Transactions on Automatic Control, vol. 37, no. 3, pp. 332–341, 1992.
4. *J. C. Spall*, Introduction to Stochastic Search and Optimization: Estimation, Simulation, and Control.
5. *O. Granichin* and *B. Polyak*, Randomized Algorithms of an Estimation and Optimization Under Almost Arbitrary Noises. Moscow: Nauka, 2003.
6. *H. Chen, T. E. Duncan*, and *B. Pasik-Duncan*, "A kiefer-wolfowitz algorithm with randomized differences," IEEE Transactions on Automatic Control, vol. 44, no. 3, pp. 442–453, 1999.
7. *O. Granichin*, "A stochastic recursive procedure with correlated noise in the observation, that employs trial perturbations at the input," Vestnik Leningrad University: Math, vol. 22, no. 1, pp. 27–31, 1989.
8. *B. T. Polyak* and *A. B. Tsybakov*, "Optimal order of accuracy of search algorithms in stochastic optimization," Problemy Peredachi Informatsii, vol. 26, no. 2, pp. 45–53, 1990.
9. *O. Granichin*, "Procedure of stochastic approximation with disturbances at the input," Automation and Remote Control, vol. 53, no. 2, part 1, pp. 232–237, 1992.
10. *O. Granichin*, "Randomized algorithms for stochastic approximation under arbitrary disturbances," Automation and Remote Control, vol. 63, no. 2, pp. 209–219, 2002.
11. *O. Granichin* and *O. Izmakova*, "A randomized stochastic approximation algorithm for self-learning," Automation and Remote Control, vol. 66, no. 8, pp. 1239–1248, 2005.
12. *N. Amelina, O. Granichin*, and *A. Kornivetc*, "Local voting protocol in decentralized load balancing problem with switched topology, noise, and delays," in Proc. of the 52st Conference on Decision and Control (CDC2013), pp. 4613–4618, 2013.
13. *S. Dhakal, B. Paskaleva, M. Hayat, Schamiloglu, E.*, and *C. Abdallah*, "Dynamical discrete-time load balancing in distributed systems in the presence of time delays," in Proc. of the 52st Conference on Decision and Control (CDC2013), pp. 5128–5134, 2013.
14. *A. Vakhitov, O. Granichin*, and *L. Gurevich*, "Algorithm for stochastic approximation with trial input perturbation in the nonstationary problem of optimization," Automation and Remote Control, vol. 70, no. 11, pp. 1827–1835, 2009.
15. *O. Granichin, L. Gurevich*, and *A. Vakhitov*, "Discrete-time minimum tracking based on stochastic approximation algorithm with randomized differences," in Proc. of the 48th IEEE Conference on Decision and Control (CDC2009), pp. 5763–5767, 2009.
16. *A. Vakhitov, V. Vlasov*, and *O. Granichin*, "Adaptive Control of SISO Plant with Time-Varying Coefficients Based on Random Test Perturbation".

АЛГОРИТМ ОРИЕНТИРОВАНИЯ СВЕРХЛЕГКОГО БПЛА ПО ДАННЫМ БОРТОВОГО ФОТО-ВИДЕОРЕГИСТРАТОРА[1]

В. К. Филатов

студент 5 курса кафедры системного программирования СПбГУ

E-mail: vova.filatov@gmail.com

Аннотация. В статье описан метод позиционирования БПЛА без использования систем спутникового наведения, с помощью камеры, установленной на борту. Метод основан на алгоритме ориентирования человека в неизвестной ему местности. Планируется спроектировать систему, которая сможет построить и запомнить карту маршрута БПЛА с выделением характерных объектов, и поможет ему не потеряться при следующих полетах по этой местности.

Введение

В современном мире для решения задач мониторинга территории все чаще применяют беспилотные летательные аппараты (БПЛА), которые могут выполнять поставленную им задачу (например, полет по маршруту по заданным точкам) автоматически. На таких аппаратах, как правило, установлены автопилот, инерциальная система навигации и система навигации ГЛОНАСС/GPS, что позволяет самолету ориентироваться в пространстве и выполнять задачу. В основном, БПЛА используются для решения военных задач, но за счет увеличения технологичности производства электронных компонент и уменьшения их стоимости БПЛА стали активно применятся в гражданских целях. Одной из задач, которые решают БПЛА — мониторинг территории заповедников, где актуальны такие вопросы как: исследования миграции животных, своевременное определение нарушений режима заповедника и др.

В больших дорогостоящих БПЛА, которые применяются в военных целях, для позиционирования в пространстве используется полный комплекс инерциальной системы в сочетании с ГЛОНАСС/GPS и системой поправок для увеличения точности. Точность таких систем позиционирования достигает 1–2 см. Из-за уменьшения стоимости легких БПЛА увеличивается доступность таких решений, но при этом в качестве систем навигации на таких БПЛА устанавливается только система ГЛОНАСС/GPS и магнитометр для определения курса движения. Данные со спутников GPS обновляются с частотой 1–5 Гц, что позволяет автопилоту часто оценивать курс движения на заданную ему точку и вносить поправки в курс. Точность граждан-

[1] Работа частично поддержана грантов РФФИ № 13-07-00250.

ских приемников ГЛОНАСС/GPS сейчас находится в пределах 5–10 метров, что позволяет довольно точно следовать по заданному маршруту.

На БПЛА может устанавливаться дополнительное оборудование, например видеокамера, для разведки местности. Эта камера может снимать видео в режиме реального времени, или делать фотографии в режиме timelaps (1 фото за устанавливаемый промежуток времени). Так на снимках можно видеть, куда мигрируют те или иные виды животных, увидеть посторонних людей на территории, искать изменения на местности.

Иногда аппарат может летать на территориях, где прием GPS сигнала может быть затруднен. Также в ходе полета у аппарата может сломаться блок навигации. В результате всего этого он может потерять ориентацию в пространстве и не вернуться в заданную точку, поэтому необходимо попробовать применить другие методы ориентирования. В качестве решения предполагается использовать фото-видеорегистратор, который беспилотник и так использует для других своих задач.

Ориентирование человека на местности

Иногда человек может оказаться на незнакомой территории и ему будет необходимо возвращаться домой. Чтобы выбраться из этой местности, ему необходимо анализировать, что происходит вокруг. У него может быть карта. По ней он прокладывает маршрут по объектам и начинает движение. Например, он видит на карте озеро, ставит азимут на него и идет туда, затем ставит азимут на следующий объект и так, пока по цепочке ориентиров не выйдет домой. Если же у него карты нет, то он может вспоминать, как он туда пришел. То есть, он может иметь в голове карту маршрута, по которому он пришел. Пока он шел, он запоминал отдельные ориентиры и двигался по ним, и чтобы вернуться домой, ему необходимо идти по ним в обратном направлении пока он не достигнет нужной цели. Это называется ориентированием человека в лесу. Можно попробовать привить похожее ориентирование БПЛА. Где карту он будет составлять сам себе, так как научить читать любую карту БПЛА пока еще сложно.

Подготовка карты местности с выделением объектов

Перед тем как летать по незнакомой местности, необходимо составить карту окружающей действительности. Для этого необходимо совершить тестовый облет по территории и все объекты, которые «увидит» беспилотник, сохранить на своей карте. Для начала нужно понять. как располагать объекты.

Фотограмметрия — технология дистанционного зондирования Земли, позволяющая определять геометрические, количественные и другие свойства объектов на поверхности земли по фотографическим изображениям, получаемым с помощью летательных аппаратов любых видов. С помощью

фотограмметрии по паре последовательных снимков, зная их углы съемки относительно осей Земли X, Y, Z, а также географические координаты можно определить координаты одинаковых точек[1]. За координаты объектов, будем принимать координаты их центра масс, чтобы как-то привязать объекты к пространственным координатам.

Также необходимо как-то выделять объекты на фотографиях. Благодаря библиотеке компьютерного зрения OpenCV([2]), это делается с помощью детектора границ Кенни, на вход которому необходимо подать отфильтрованное изображение в оттенках серого.

Путем применения этих двух областей, можно выделять объекты и сохранять их на свою карту.

Формирование базы данных объектов

Карта в понимании БПЛА — это некоторая база данных объектов, которые он увидел при первом облете или выделил на каких-нибудь известных сервисах. Нужно понять, в каком формате сохранять карту, чтобы к ней можно было легко и быстро обращаться, а также, чтобы по ней можно было сформировать полетное задание, при отсутствии GPS сигналов. Необходимо также как-то описать объекты, которые хранятся в этой базе, чтобы по ним потом можно было сформировать задание. В реальной жизни человек, работая с картой, оперирует с объектами такими понятиями:

1. Название объекта (дом, река, красный столб, и т. д.).
2. Характеристики объекта.
3. Координаты объекта на карте.
4. Контур объекта.

Для БПЛА описание объекта должно быть уникальным, так как, если указать ему лететь к столбу, он должен будет найти контур нужного столба и лететь, пока не увидит похожий через свой фото-видеорегистратор.

Возникает вопрос, в каком формате хранить объекты на карте. На сегодняшний день существует множество Геоинформационных систем, которые как-то хранят и используют геоданные, сделанные своими или чужими средствами картографирования земной поверхности. Создавать новую систему хранения геоданных нет необходимости, так как все существующие имеют большое количество описаний и библиотек для работы с ними. Все хранилища делятся на три типа:

- растровое хранение данных;
- векторное хранение данных;
- смешанное хранение.

В каждом хранении есть свои плюсы и минусы, ,но для решения задачи подойдет система хранения в виде шейпфайлов, так как файлы этого типа поддерживаются большинством ГИС и файлы можно открывать в различных

графических редакторах. Стандарт позволяет сохранять объекты в виде точек, линий и полигонов и «мультиобъектов» в файле.

Программой создается один полный шейпфайл для хранения информации объектов (.shp для хранения геоточек объектов, .dbf для хранения описания). В описании хранится название объекта (должно быть уникальным), и координаты объектов. Также создается один дополнительный файл .shp для хранения контуров объектов. Все вместе является хранилищем объектов для одной карты.

Формирование маршрутного задания

После того как БПЛА составил карту местности после своего первого облета (или серии полетов), можно попробовать научить его ориентироваться по какому-нибудь маршруту без GPS. Необходимо понять, как описывать маршрут в терминах объектов. Например, сказать ему «двигаться по дороге» мы не можем, так как не понятно, что должен делать БПЛА в таком случае. В качестве задания можно указать БПЛА двигаться к определенному объекту. В таком случае, он подгрузит контур объекта или его фотографию в свою память и будет ожидать появление этого контура на снимке через свой фотовидеорегистратор. В таком случае он будет работать как глаза у человека. Простейшим описанием маршрута от объекта к объекту может быть указание азимута. То есть файл с описанием может быть в виде «объект—азимут до объекта 1, объект 1—азимут до объекта 2, …». Все объекты в маршруте должны находится на карте, который сделал БПЛА.

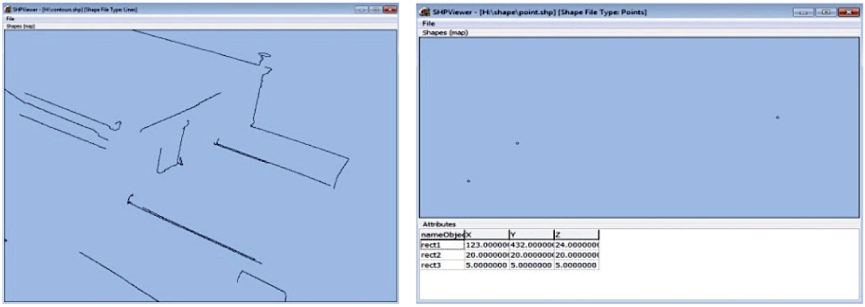
Реализация полета по заданию

Для реализации программы задача была разделена на несколько основных модулей:

1) обработка входных изображений с фотокамеры и выделение объектов и их географических координат;
2) сохранение объектов и их координат на своей карте местности исследования;
3) модуль составления маршрутов;
4) модуль движения по известному маршруту без использования GPS;

В первом и во втором пунктах на вход можно подавать пары изображений с известными углами относительно Земли. На выходе получается файл с контурами объектов, файл с описаниями объектов и файл с геоточками объектов.

Основной язык программирования C++, так как для работы с компьютерным зрением априори уже используют библиотеку OpenCV. Для работы с шейпфалами используется готовая библиотека Shapefile C Library [3], которая предлагает методы сохранения и открытия шейпфайлов.



В модуль составления маршрутов на вход можно подать текстовую строку с типа объект 1—азимут до объекта 2, объект 2—азимут до объекта 3, … В результате получится .dbf файл, который можно также посмотреть. Он отображает маршрут, который получился.

Модуль движения по известному маршруту позволяет подключиться к камере, подключенной к квадрокоптеру или к ноутбуку, и эмулирует действия, которые должен выполнять БПЛА. Сначала ждем пока в камере не появится объект1, затем смотрим азимут по маршруту, показываем его и летим по этому азимуту, пока в камере не появится контур объекта2, затем повторяем итерации пока не долетим до последнего объекта, который будет являться финишной точкой.

При успешном тестировании алгоритма, его можно полностью реализовать для БПЛА, используя трехуровневую архитектуру системы управления БПЛА (базовая станция—бортвой микрокомпьютер—автопилот), разработанную в СПбГУ Амелиным Константином [4].

Заключение

В работе построен алгоритм, который позволяет ориентироваться беспилотному летальному аппарату не по данным GPS, а по карте, которую построил он сам. Конечно, его еще сложно применять в полевых условиях, так как природные местности могут меняться часто, и контуры объектов, соответственно, будут меняться чаще, но технологии не стоят на месте, и в будущем, может быть, не надо будет задавать задание БПЛА и полетное задание он сможет формулировать себе сам. В ходе работы был также реализован прототип системы, который реализует данный алгоритм.

Литература

1. *Минько В. Ю.* Основные зависимости аналитической фотограмметрии.
2. OpenCV — open source computer vision library. URL: http://opencv.org/

3. Shapefile C Library — C programs for reading, writing and updating (to a limited extent) ESRI Shapefiles, and the associated attribute file (.dbf). URL: http://shapelib.maptools.org/
4. *Амелин К. С.* Математическое обеспечение микрокомпьютеров мобильных объектов с групповым взаимодействием / Диссертация на соискание ученой степени кандидата физико-математических наук (2012).
5. *Амелин К. С.* «Технология программирования легкого БПЛА для мобильной группы», Стохастическая оптимизация в информатике.

ПРИМЕНЕНИЕ ОНТОЛОГИЙ В МАС НА ПРИМЕРЕ АЛГОРИТМА МУРАВЬИНОЙ КОЛОНИИ

Д. Г. Найданов

Санкт-Петербургский государственный университет

E-mail: dmitry.naydanov@jetbrains.com

Р. Е. Шеин

Санкт-Петербургский государственный университет

E-mail: marso.des@gmail.com

Аннотация: Данная работа посвящена обзору онтологий как инструмента проектирования и реализации мультиагентных систем, а также исследования возможностей расширения применимости онтологий в реализации мультиагентных систем на примере модифицированного алгоритма муравьиной колонии. На сегодняшний день онтологии в МАС используются для работы с представлением агента о внешнем мире. В данной работе предлагается подход, использующий онтологии не только для работы со знаниями, но и дляреализации управления агента.

Ключевые слова: мультиагентные системы, онтологии, базы знаний, дескрипционные логики, алгоритмымуравьиной колонии

1. Введение

Онтологии — основанные на дескрипционных логиках системы представления знаний, получившие широкое распространение в решении задач каталогизации, классификации и представления дружественном для человека виде информации: существует множество проектов, использующих онтологии для представления знаний ([1–4] и другие). Помимо вышеуказанных применений, в которых базы знаний в первую очередь ориентированы на применение их человеком, на сегодняшний день всё чаще онтологии используются для хранения представлений о мире агентов в мультиагентных системах(далее МАС) [5–9]. Это позволяет применять алгоритмы вывода в дескрипционных логиках непосредственно на базе знаний агента, а также обеспечивает большую стандартизацию и упорядоченность знаний, что позволяет упростить коммуникацию между агентами. Данный обзорный доклад посвящен известным на текущий момент способам применения онтологий при проектировании и реализации МАС, проблемам и преимуществам представления знаний агентов в виде онтологий. Также, помимо часто встречающегося на сегодняшний день применения онтологий для представления знаний агента о внешнем мире, в работе рассматривается естественное распространение этой идеи непосредственно на механизм управления агентом

на примере внедрения онтологии действий для МАС типа муравьиной колонии для автоматического распознавания агентами успешных шаблонов поведения и внедрения их в деятельности всей МАС.

2. Дескрипционныелогики

В этом разделе мы определим основные понятия дескрипционных логик (далее — DL), которые применяются для представления знаний в онтологиях. В общем смысле, DL — язык представления знаний, позволяющий описывать понятия предметной области в формальном виде. Подробный обзор дескрипционных логик, включая их синтаксис и семантику, можно найти в [10].

Пусть C — множество концептов (аналог унарных предикатов), R — множество ролей (аналог бинарных предикатов), I — множество имен индивидов (т. е. объектов предметной области) DL. Тогда терминологией (или сокращенно *TBox*) называется набор утверждений вида $C \equiv D$ (эквивалентность) или CD (включение), где C и D — произвольные концепты; системой фактов (ABox) называется набор утверждений вида a: C и aRb, где a, b I есть индивиды, C — произвольный концепт, R — роль.

Описания концептов интерпретируется в классическом смысле теории моделей: интерпретация I — это пара (Y, f), где Y — непустое множество индивидов, а f — функция интерпретации, которая отображается множество концептов C в Y и множество имен ролей в YY.

Интерпретация I называется моделью для терминологии T, если для любого включения ADT верно $f(A)$ f (D) и для любой эквивалентности $A \equiv B$ верно $f(A) = f(B)$.

Концепт C называется выполнимым в терминологии T, если существует такая интерпретация (Y, f), являющаяся моделью T, что $f(C)$ непусто.

3. Использование онтологий

3.1. Способы применения онтологий в мультиагентных системах

Онтологии широко применяются на различных стадиях разработки и эксплуатации мультиагентных систем. Онтологии используют для проектирования различных МАС [11–12], а также при реализации МАС для решения задач управления продажами и арендой недвижимости [5], торговли на бирже [6], поддержания безопасности информационных ресурсов [7], информационного поиска [8–9] и других задач.

На стадии проектирования и разработки модели мультиагентной системы для систематизации знаний о предметной области и решаемой задаче используются основанные на онтологиях инструменты и методологии [11–12].

В реализациях мультиагентных систем онтологии применяются для систематизации предметной области и доступных агентам знаний. Можно выделить три качественно различных подхода к интеграции онтологии в работу мультиагентной системы: либо каждый агент хранит свою онтологию, содержащую доступные именно ему знания и понятия (будем называть такие МАС распределенными) [9], либо онтология едина для всех агентов и хранится централизованно (как правило, на специальном агенте, будем называть такие МАС централизованными) [5–7], либо онтология частично едина, а частично — распределена (будем называть такие МАС гибридными) [8].

Как правило, для работы с МАС применяются таксономические онтологии, фактически представляющие знания в виде иерархии.

3.2. Проблемы применения онтологий в мультиагентных системах

3.3.1. Выразительность или вычислительная сложность

Одним из ключевых преимуществ онтологий над остальными методами организации информации является возможность на основе имеющихся данных автоматически вывести корректный ответ на вопрос вида «выполняется ли концепт C для инвидидаA?» или «верно ли, что роль R осуществляется индивидами A и B?», т.е. разрешить запрос в дескрипционной логике, лежащей в основе онтологии. С увеличением выразительности логики, лежащей в основе применяемой онтологии, растёт алгоритмическая сложность вывода ответа на подобный вопрос. Однако, несмотря на то, что в общем случае задача разрешимости дескрипционной логики часто даже не полиномиальная по времени [15], на практике используются эвристические разрешающие алгоритмы, обеспечивающие разумное время вывода.

3.3.2. Коммуникации между агентами

В гетерогенной МАС при передаче данных из онтологии одного агента к другому в случае отсутствия передаваемых концептов в онтологии принимающего возникает конфликтная ситуация — данные не могут быть непосредственно добавлены в систему знаний принимающего агента. Для решения этой проблемы известны следующие подходы: интеграция онтологий(ontology integration [16]), настройка онтологий (ontology alignment [17]), сервисы преобразования онтологий (ontology mediation services [18–19]) и обсуждение онтологий (ontology negotiation [9]). Как показано в [9], первые три подхода не годятся для общего случая гетерогенной МАС с динамически меняющимися онтологиями в каждом агенте. Описанный же в [9] подход работает только для таксономических онтологий. Предположительно, данная проблема может быть разрешена применением рандомизированных алгоритмов [20].

4. Модифицированный алгоритм муравьинойколонии

4.1. Алгоритмы муравьиной колонии

Одним из возможных вариантов применения онтологий действий в мультиагентных системах является использование дескрипционной логики в качестве управляющей системы для агентов в т.н. алгоритме муравьиной колонии [22] .

Алгоритмы муравьиной колонии — семейство оптимизационных алгоритмов с ярко выраженной мультиагентной природой. Своим происхождением алгоритмы обязаны муравьям: поведение агентов копирует поведение муравьёв, ищущих ближайший к муарвейнику источник пищи: первый муравей находит источник пищи любым способом, а затем возвращается к гнезду, оставив за собой тропу из феромонов. Затем муравьи выбирают один из возможных путей, затем укрепляют его и делают привлекательным. Муравьи выбирают кратчайший маршрут, так как у более длинных феромоны сильнее испарились. Среди экспериментов по выбору между двумя путями неравной длины, ведущих от колонии к источнику питания, биологи заметили, что, как правило, муравьи используют кратчайший маршрут. В общем виде постановка задачи, решаемой алгоритмом муравьиной колонии, такова: агенты, передвигаясь по взвешенному графу, в каждый момент времени к каждому ребру которого приписано численное значение интенсивности «феромона» на этом ребре, должны найти экстремум некой целевой функции; при выборе пути агент с большей вероятностью выбирает пусть с большей интенсивностью «феромона», а при нахождении решения агент помечает пройденный им путь количеством «феромона», обратно пропорциональным стоимости решения. Данный алгоритм способен эффективно эвристически разрешать некоторые NP-полные задачи, например, задачу коммивояжера [23–25].

Для тестирования алгоритма был написан симулятор с моделью, обладающей следующими характеристиками:

1) время дискретно;
2) среда представляет собой прямоугольник известного размера, с дискретными координатами;
3) каждая точка среды обладает численной характеристикой, которая показывает, что находится в данной точке (0 — еда, 1 — непроходимое препятствие и т.д.);
4) часть среды является «муравейником». Кроме того, были рассмотрены различные модификации постановки задачи (конечность или бесконечность источников еды, необходимость возвращаться с едой в муравейник, «прозрачность» агентов и т.п.).

В качестве эталона на симуляторе был написан стандартный алгоритм. Следующий этап — реализация агентов, использующих онтологии действия. В качестве дескрипционной логики для такой онтологии была выбрана

ALC [15]. В терминах этой логики индивиды интерпретировались как действия, а концепты — как некоторые оценки того, приведет ли то или иное действие к достижению цели (например, концепт isPathToFood, который выдает true для действия, передвигающего агента к еде). При этом множество атомарных концептов неизменяемо (при этом мы можем порождать новые составные концепты по правилам логики), а все индивиды порождаются по заранее заданным шаблонам. Также у агентов была возможность обмениваться своими системами фактов (целиком или частично), при этом для каждого факта, полученного от другого агента и содержащего информацию о месте нахождения еды, в зависимости от удаленности места определялась пороговая вероятность, с которой факт добавлялся в онтологию принимающего агента.

Сравнение стандартного алгоритма и алгоритма на основе онтологии действий на ряде модельных примеров показало в среднем сопоставимые результаты в терминах количества единиц пищи, собранной за ограничиенной время. Однако, в связи с отсутствием на данный момент метрики для сравнения сложности агентов, а также слишком большой вариативностью постановки задачи, такое сравнение имеет ориентировочный характер. Для получения существенных данных требуется сравнение алгоритмов на некой конкретной практической задаче, что является следующим этапом нашей работы.

5. Заключение

На сегодняшний день онтологии всё чаще применяются и как инструмент проектирования и разработки агентов, и как средство представления знаний об окружающем мире для самого агента. Рассмотренный в данной работе пример показывает акутальность введения онтологии действий в управлении агента для повышения гибкости и автономности поведения агента, а также распределённости МАС. Не использующий заранее запрограммированных в агента конкретных моделей поведения алгоритм на основе онтологии действий показал результаты, сопоставимые с классическим алгоритмом муравьиной колонии, кроме того, нетрудно видеть, что сам алгоритм муравьиной колонии является частным случаем алгоритма, основанного на онтологии действий при наличии в онтологии действий только действий «идти по следу феромона за пищей» и «возвращаться в муравейник, оставляя след феромона», то есть, рассматриваемый в данном докладе подход может применяться для большей стандартизации и обобщения уже известных алгоритмов поведения агентов МАС.

Перспективным направлением применения онтологий в МАС являются группы беспилотных летательных аппаратов. В современном мире беспилотные летательные аппараты (англ. Unmanned Aerial Vehicles, БПЛА) приобретают все большую популярность в качестве недорогих инструментов для исследования территорий, разведки и воздушной съемки. В основной

массе современные БПЛА не обладают автономностью (обычно управляются оператором с пульта). Есть успешные разработки одиночных БПЛА, которые под управлением автопилота способны в автономном режиме облететь территорию по заданному маршруту, проводя фотосъемку или выполняя другие задачи. Анализ большинства задач, решаемых с использованием одиночных БПЛА, показывает, что они могут более эффективно решаться группой, так как у группы взаимодействующих летательных аппаратов появляются дополнительные полезные свойства.

Таким образом, для беспилотных летательных аппаратов в ближайшее время могут стать актуальными проблемы группового взаимодействия[21]. Одним из возможных способов решения этой проблемы и планирования стратегии поведения каждого отдельного агента является использование онтологии действий как основы системы управления, применяющей автоматический вывод на основе онтологического знания как естественный и заведомо корректный способ выбора поведения для агента.

Также применение онтологий обеспечит стандартизацию представления знаний, что, предположительно, упростит обмен информацией между гетерогенными агентами (по модели, аналогичной [7]), а также позволит при часто встречающейся в реальных проектах ситуации невозможности связи между агентами восстановить или предсказать наперёд поведение другого агента с некоторой точностью на основе известных частей его онтологии.

Литература

1. Yahoo! Web Directory (http://dir.yahoo.com/)
2. OpenCYC (http://www.opencyc.org/)
3. GO (http://www.geneontology.org/)
4. SNOMED (http://www.ihtsdo.org/snomed-ct/)
5. http://ausweb.scu.edu.au/aw07/papers/refereed/yang/paper.html
6. *Weir Ying, Anjalee Sujanan*i. Design and development of financial application using ontology-based multi-agent systems. Computing and Informatics, Vol. 28, 2009, 635–654.
7. *Ryan Ribeiro de Azevedo, Eric Rommel Galvão Dantas, Fred Freita, Cleyton Rodrigues, Marcelo J. Siqueira C. de Almeida, Wendell Campos Veras* and *Rubean Santos*. An Autonomic Ontology-Based Multiagent System for Intrusion Detection in Computing Environments. International Journal for Infonomics (IJI), Volume 3, Issue 1, March 2010.
8. *Emmanuel Solidakis, Nikolaos Konstantinou, Emily-Sirin Pashou, Anthi Papa-konstantinou* and *Nikolas Mitrou*. A Decentralized Multi-Agent Ontology-Based System-for Information Retrieval (2005).
9. *Jurriaan van Diggelen, Robbert-Jan Beun, Frank Dignum, Rogier M. van Eijk* and *John-Jules Meyer*. Ontology negotiation in heterogeneous multi-agent systems: The ANEMONE system. Applied Ontology 2 (2007) 267–303 IOS Press.
10. *Baader F., Calvanese D., McGuinness D., Nardi D. & Patel-Schneider P*. Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press., 2003.

11. http://www.open.org.au/Conferences/oopsla2004/PapersAO/3-Girardi.pdf
12. *Jonathan DiLeo, Timothy Jacobs* and *Scott DeLoach*. Integrating Ontologies into Multiagent Systems Engineering. Fourth International Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS2002). 15–16 July 2002, Bolgona (Italy).
13. http://www.w3.org/TR/rdf-concepts/
14. http://xmlhack.ru/texts/06/rdf-quickintro/rdf-quickintro.html
15. *Staab Steffen* and *Rudi Studer*. Handbook on ontologies. Berlin: Springer, 2009. Print. Sure, Y., Staab, S. & Studer, R. (2004). On-to-knowledge methodology (OTKM). In S. Staab & R. Studer (Eds.), Handbook on ontologies (Chapter 6, pp. 117–132). Springer.
16. *Sure Y., Staab S. & Studer R*. (2004). On-to-knowledge methodology (OTKM). In S. Staab & R. Studer (Eds.), Handbook on ontologies (Chapter 6, pp. 117–132). Springer.
17. *Ciocoiu M., Gruniger M. & Nau D*. (2001). Ontologies for integrating engineering applications. Journal of Computing and Information Science in Engineering, 1(1), 12–22.
18. *Wiederhold G. & Genesereth M*. (1997). The conceptual basis for mediation services. IEEE Expert: Intelligent Systems and Their Applications,12(5), 38–47.
19. FIPA (2000). FIPA Ontology Service Specification.
20. *Граничин О. Н., Поляк Б. Т*. Рандомизированные алгоритмы оценивания и оптимизации при почти произвольных помехах. М.: Наука, 2003. 291 с.
21. *Амелин К. С., Бакановский М. В., Граничин О. Н*. и др. Адаптивная мультиагентная операционная система реального времени // Стохастическая оптимизация в информатике. 2013. Т. 1. С. 3–16.
22. *Marco Dorigo, Luca Maria Gambardella*. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem // IEEE Transactions on Evolutionary Computation. Vol. 1. No. 1. April 1997.
23. *A. Colorni, M. Dorigo* and *V. Maniezzo*. Distributed optimization by ant colonies // Proc. ECAL91— Eur. Conf. Artificial Life. New York: Elsevier, 1991, pp. 134–142.
24. An investigation of some properties of an ant algorithm // Proc. Parallel Problem Solving from Nature Conference (PPSN 92) New York: Elsevier, 1992, pp. 509–520.
25. *M. Dorigo, V. Maniezzo* and *A. Colorni*. The ant system: Optimization by a colony of cooperating agents // IEEE Trans. Syst, Man, Cybern. B, vol. 26, no. 2, pp. 29–41, 1996.

ИСПОЛЬЗОВАНИЕ СКРЫТЫХ МАРКОВСКИХ МОДЕЛЕЙ ПРИ РАЗРАБОТКЕ ИНТЕРФЕЙСА ВЗАИМОДЕЙСТВИЯ С ПК ПРИ ПОМОЩИ ДВИЖЕНИЙ ГОЛОВЫ

И. Н. Калитеевский
E-mail: i.kalit@yandex.ru

В. Н. Калитеевский
E-mail: vkalit@gmail.com

Аннотация. В настоящее время проводятся обширные исследования в области новых интерфейсов взаимодействия между человеком и компьютером. Одно из актуальных направлений — интерфейс на основе машинного зрения.

Работа посвящена реализации и исследованию скрытых марковских моделей для создания интерфейса основанного на распознавании движений головы, воспринимающихся как жесты на примере расширения для браузера.

Введение

Способы взаимодействия компьютера и пользователя почти не изменились со времён появления первых персональных компьютеров. Клавиатура и мышь, сразу став основными устройствами ввода, практически не получили развития. Управление голосом широкого применения не нашло. Большой шаг вперед сделали производители смартфонов, внедрив touch-интерфейс, оказавшийся естественным и интуитивно понятным, однако, для персональных компьютеров он не подходит.

В настоящее время компьютерная индустрия проявляет большой интерес к интерфейсам, основанным на машинном зрении. Действительно, такие интерфейсы содержат значительный потенциал, для реализации и внедрения которого должны быть выполнены следующие требования:

- **Безошибочность.** Нажатие на клавишу не должно быть неоднозначно трактовано. Неоднозначностей не должно возникать и при распознавании жестов. Нужно использовать только те жесты, которым можно гарантировать стабильное распознавание;
- **Естественность жестов.** Например, приближение головы — явный знак, что человек хочет рассмотреть поближе какой-то мелкий объект, уход из поля зрения — можно трактовать как возможность перейти в режим пониженного энергопотребления, или как команду «пауза» при просмотре кино и т. д. И наоборот, вынуждать пользователя, запоминать неестественные жесты, например, дважды кивать для того чтобы свернуть окно представляется нецелесообразным;

- **Стандартизация.** В целях экономии процессорного времени, а так же для того, чтобы у каждой программы не было своего уникального списка жестов, нужна единая служба, встроенная в операционную систему, к которой могли бы обращаться другие приложения.
  Была выбрана следующая карта жестов:

Жест	Команда
Кивок	Переход на домашнюю страницу браузера
Наклон головы влево + возврат	Переход вверх страницы
Наклон головы вправо + возврат	Переход вниз страницы
Поворот головы влево + возврат	Переход на предыдущую страницу
Поворот головы вправо + возврат	Переход на следующую страницу
Приближение	Увеличение содержимого на 30%
Удаление	Уменьшение содержимого на 30%

Скрытые марковские модели

Движения головы продолжаются определенный интервал времени и являются непрерывным процессом, однако мы будем рассматривать дискретную модель и представлять жесты в виде последовательности определенных положений головы в пространстве.

Теоретически любая последовательность может быть сгенерирована некоторым параметризованным случайным процессом. Моделирование такого процесса может быть организовано с помощью скрытой Марковской модели, параметры которой могут быть настроены на основе обучающих последовательностей [1, 2].

Дискретные скрытые марковские модели (СММ) описывают стохастические процессы, состоящие из множества состояний, каждое из которых связано с другим стохастическим процессом. Формально могут быть описаны следующим образом [3, 4, 5]:

1. N — общее количество состояний в модели. Переход в любое выбранное состояние возможен из любого состояния всей системы (в том числе и само в себя). Алфавит ненаблюдаемой последовательности мы обозначим как $\Omega = \{\omega_1, ..., \omega_N\}$.
2. M — количество возможных символов в наблюдаемой последовательности, размер алфавита наблюдаемой последовательности. Алфавит наблюдаемой последовательности мы обозначим как $O = \{o_1, ..., o_M\}$.
3. Вероятностное распределение смены состояний $A = \{\alpha_{ij}\}$, где α_{ij} представляет вероятность смены состояния из ω_i во время t в состояние ω_j во время $t+1$.

$$\alpha_{ij} = P(\omega_j(t+1) \mid \omega_i(t)),\ 1 \le i,\ j \le N.$$

4. Вероятностное распределение выбора части распознаваемого жеста $B = \{\beta_{il}\}$, где β_{il} представляет вероятность выбора части жеста o_l во время t в состоянии ω_i.

$$\beta_{il} = P(o_l(t) \mid \omega_i(t)),\ \sum_{l=1}^{M} \beta_{il} = 1 \text{ для } \forall i.$$

5. Первоначальное распределение состояний $\pi = \{\pi_i\}$.

$$\pi_i = P(\omega_i(1)),\ 1 \le i \le N.$$

Полный набор параметров СММ будем обозначать как $\lambda = \{A, B, \pi\}$.

Подготовка признаков

Предварительная работа перед обучением и распознаванием делится на следующие этапы:

- Выделение жестов из непрерывного потока.
- Подготовка входных параметров.
- Извлечение признаков.

С помощью библиотеки Intel Perceptual Computing SDK будем получать координаты положения частей лица, среди которых нос, глаза и рот. Введем следующие обозначения: точка A — координата центра рта, точка B — координата середины отрезка, соединяющего глаза, тогда α есть отклонение угла между AB и горизонтальной осью, d фиксирует изменение расстояния между точками A и B, а A_x — горизонтальное изменение положения точки A. Параметры $\langle \alpha, d, A_x \rangle$ и возьмем в качестве базовой тройки признаков.

Для детектирования начала жеста высчитывается суммарное изменение каждого из трех признаков за последние десять кадров. В случае преувеличения одного из признаков некоторого порогового значения начинается покадровая запись жеста и продолжается до возвращения параметров в исходное положение. Затем берется среднее изменение каждого из параметров на участках жеста, равных количеству состояний скрытой марковской модели и в качестве извлеченных признаков подаются на вход алгоритму обучения.

Обучение СММ

Обучение скрытой марковской модели является одной из основных задач СММ и формально формулируется следующим образом [6]:

Дано: наблюдаемая последовательность $O = \{o_1, ..., o_M\}$. Подобрать параметры модели $\lambda = \{A, B, \pi\}$ таким образом, чтобы максимизировать вероятность $P(o|\lambda)$.

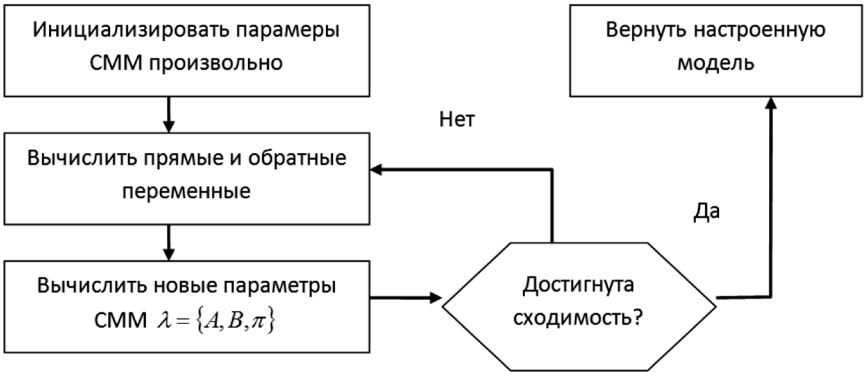
Решается данная задача с помощью алгоритма Баума|v.| — |v.|Уелша [7]. Этот метод рассматривает значение вероятности перехода из состояния ω_i в ω_j как порядок между ожидаемым количеством переходов из состояния ω_i в ω_j и ожидаемым общим количеством переходов из ω_i. Таким же способом оценка вероятности выбора части жеста o_l в состоянии ω_i получается за счет вычисления порядка между ожидаемым количеством раз, когда будет выбрана часть жеста o_l в состоянии ω_i и ожидаемым общим количеством раз в состоянии ω_i. Оценка начальной вероятности в состоянии ω_i есть ожидаемая частота в ω_i во время $t=1$. Эти частоты могут быть посчитаны через вероятности перехода из $\omega_i(t)$ в $\omega_i(t+1)$ как

$$v_{ij}(t) = P(\omega_i(t), \omega_j(t+1)|o, \lambda) = \frac{\delta_i(t)\alpha_{ij}\beta_{jo(t+1)}\eta_j(t+1)}{P(o, \lambda)}.$$

Используя $v_{ij}(t)$, становится возможным определить значения параметров СММ:

$$\pi_i = \sum_{j=1}^{N} v_{ij}(1), \quad \alpha_{ij} = \frac{\sum_{t=1}^{T-1} v_{ij}(t)}{\sum_{t=1}^{T-1}\sum_{j=1}^{N} v_{ij}(t)}, \quad \beta_{il} = \frac{\sum_{t=1,\, o(t)=o_l}^{T}\sum_{j=1}^{N} v_{ij}(t)}{\sum_{t=1}^{T}\sum_{j=1}^{N} v_{ij}(t)}.$$

Схема работы алгоритма Баума—Уелша:



Этап распознавания

Задача распознавания скрытой марковской моделью имеет следующую формальную постановку:

Дано: наблюдаемая последовательность $O = \{o_1, ..., o_M\}$ и модель $\lambda = \{A, B, \pi\}$. Необходимо подобрать последовательность состояний системы

$\Omega = \{\omega_1, ..., \omega_N\}$, которая лучше всего соответствует наблюдаемой последовательности.

Решается путем применения алгоритма Витерби [8], суть которого заключается в том, чтобы сохранять путь до текущего наилучшего состояния ω_j во время $t-1$, который имеет наибольшую вероятность получения наблюдаемой последовательности $\{o(1), o(2), \ldots, o(t)\}$.

Эта вероятность может быть вычислена рекурсивно

$$\zeta_i(t) = \left(\max_j \zeta_j(t-1)\alpha_{jt} \right) \beta_{io}(t).$$

Соответствующее наилучшее состояние сохраняется в переменной $\zeta_i(t)$.

Алгоритм Витерби:

1. Инициализация, $t=1$, $\zeta_i(1) = \pi_i \beta_{io(1)}$, $1 \le i \le N$, $\zeta_i(1) = 0, 1 \le i \le N$.

2. Рекурсия ($t = 2, \ldots, T$). $\zeta_i(t) = \left(\max_j \zeta_j(t-1)\alpha_{jt} \right) \beta_{io(t)}$, $1 \le i \le N$, $\zeta_i(t) = \arg\max_j \zeta_j(t-1)\alpha_{ji}$, $1 \le i \le N$.

3. Остановка. $\zeta^* = \max_j \zeta_j(T)$, $\omega^*(T) = \arg\max_j \zeta_j(T)$.

4. Обратная связь ($t = T-1, T-2, \ldots, 1$). $\omega^*(t) = \zeta_{\omega^*(t+1)}(t+1)$.

Выводы и планы на будущее

До перехода к скрытым марковским моделям был реализован алгоритм, основанный на идее EM-метода, измеряющий отклонение наблюдаемого движения от некоторой эталонной траектории для каждого жеста. В таблице ниже приводится сравнение вероятностей правильного детектирования жестов двумя алгоритмами.

	Алгоритм, основанный на идее EM-метода	**СММ**
Кивок	60 %	80 %
Наклон влево	70 %	90 %
Наклон вправо	70 %	90 %
Поворот влево	70 %	90 %
Поворот вправо	70 %	90 %
Приближение	95 %	95 %
Удаление	95 %	95 %

При использовании скрытых марковских моделей наблюдается существенный подъем вероятности правильного детектирования движений.

В дальнейшие планы входит более подробный сравнительный анализ инструментов и методов распознавания образов с учетом приобретенного опыта, исследование возможностей рандомизации для уменьшения системных требований.

Литература

1. *M. Jones* and *P. Viola*, «Fast multi-view face detection», Mitsubishi Electric Research Laboratories, Tech. Rep. 096, 2003.
2. *Yang J., Xu Y.* Hidden Markov Model for Gesture Recognition: Technical Report CMU.
3. *Граничин О. Н., Поляк Б. Т.* Рандомизированные алгоритмы оценивания и оптимизации при почти произвольных помехах. М.: Наука. 2003.
4. *Avros R., Granichin O., Shalymov D., Volkovich Z., Weber G.-W.* Randomized algorithm of finding the true number of clusters based on Chebychev polynomial approximation (Chapter 6) // Data Mining: Found. & Intell. Paradigms, D. E. Holmes, L. C. Jain (Eds.), Berlin Heidelberg: Springer-Verlag, ISRL 23, vol. 1, 2012, pp. 131–155.
5. *Граничин О. Н.* Обратные связи, усреднение и рандомизация в управлении и извлечении знаний // Стохастическая оптимизация в информатике. 2012. Том. 8. Вып. 2. С. 3–48.
6. *Местецкий Л. М.* «Математические методы распознавания образов», МГУ, ВМиК, Москва, 2002–2004., с. 42–44.
7. *Rabiner L.* A tutorial on hidden Markov models and selected applications in speech recognition . Proceedings of the IEEE , 77 : 257–286, 1989.
8. *Viterbi A.* Error bounds for convolutional codes and an asymptotically optimal decoding algorithm . IEEE Transactions on Information Theory, IT-13 : 260–269, 1967.

МЕТОД ДИСТАНЦИОННОГО МОНИТОРИНГА САНИТАРНОГО СОСТОЯНИЯ ЛЕСА

Е. В. Короткова
студентка ИИТУ СПбГПУ 6 курс
E-mail: e.korotkova@geoscan.aero

А. В. Самочадин
к.т.н. ИИТУ СПбГПУ
E-mail: samochadin@soft-consult.ru

Аннотация. В работе предлагается метод выделения деревьев на аэрофотоснимках и их классификации по санитарным показателям.

Введение

В работе решается задача дистанционного мониторинга санитарного состояния леса. Основной целью мониторинга является оперативное обнаружение отклонений от нормативов санитарных показателей и их оценка [6]. Примерами детектируемых таким образом патологий могут быть сухостои, ветровалы, повреждения деревьев вредителями. Рассматриваемый вид мониторинга, в общем случае, осуществляется путем сравнительного анализа аэрокосмоснимков интересующей местности. Сначала производится поиск признаков, по которым можно было бы судить о появлении патологий. На основании найденных признаков делается заключение о патологиях, затем вычисляются количественные оценки повреждений леса, произошедших за интересующий период времени или впоследствии каких-либо событий. Также на основании полученных оценок, в дальнейшем, можно прогнозировать развитие патологий.

Анализ снимков может быть визуальным, измерительным, автоматизированным и автоматическим. На практике преобладает применение измерительно-визуальных методик, которые известны своей ресурсозатратностью. В рамках работы будет предложена экспертная система (далее ЭС), предназначенная для автоматизации процесса обнаружения лесных патологий на снимках. В качестве материалов для формирования базы знаний ЭС будет использоваться методическое пособие «Дистанционные методы контроля лесопатологического состояния лесов», предоставленное организацией «Севзаплеспроект», заинтересованной в проведений данной работы [10, 11].

Известные способы решения проблемы

Для поиска патологических изменений на снимках леса используется технология создания мультивременного композита. Она заключается в совме-

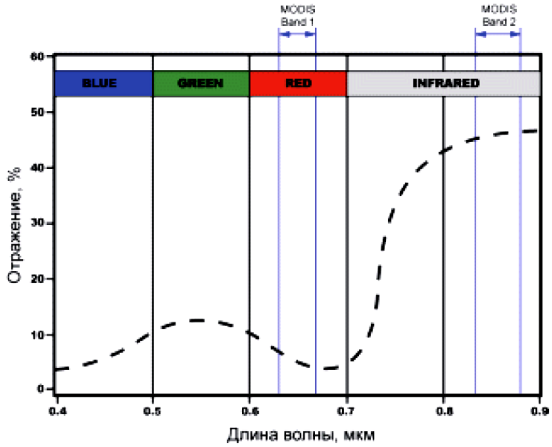


Рис. 1. Отражение волн в разных частях спектра растениями

щении спектральных каналов двух изображений на одном снимке. Для этого используются 4 спектра: 3 видимых (BGR) и ближний инфракрасный. Данная техника основана на том, что хлорофил-содержащие объекты имеют маленькое значение отраженной составляющей в области «красного» спектра и большое в области ближнего инфракрасного (см. Рис. 1) [1].Вегетационным индексом (ВИ) называется показатель, рассчитанный из значений в упомянутых диапазонах, например разностный ВИ — это разность значений, полученных в инфракрасном и красном диапазонах [8]. По сути, яркость пикселя в мультивременном композите определяется разностью вегетационных индексов объектов, соответствующих этому пикселю на двух снимках. Чем больше разность между ними, тем ярче будет пиксель, в случае убывания биологической массы, и наоборот.

Данная технология автоматизирует процесс выделения цветом области изменения биологической массы, например, сухостоев и ветровалов. Затем эксперт производит визуальную классификацию или автоматизированную, определяя пороговое значение с помощью эмпирических методик [6].

Недостатками этого подхода являются:

- точность метода сильно зависит от разности вегетации растительности в момент съемки;
- большое количество ложных срабатываний.

Постановка задачи

Спроектировать ЭС для классификации изменений на снимках. Система должна принимать решение для двух входных наборов разновременных снимков одной географической области, сделанных в период вегетации ра-

стительности, подлежащей мониторингу. Выходом системы будет структура, ставящая каждой точке изображений в соответствие наличие изменения (бинарная характеристика) и его тип. Система должна определять следующие типы деревьев: здоровое, ветровал, старый сухостой, новый сухостой, ослабленное, сильно ослабленное. В качестве входных параметров будут использоваться: цвет, форма горизонтальной проекции кроны, средний диаметр, форма тени, просматриваемость полога, наличие сухих ветвей, сомкнутость полога. Обеспечить расширяемость списка входных параметров и выходных классов.

Предлагаемые этапы решения задачи

1. Формирование 2 векторов входных данных.
   a. Аэрофотоснимки:
      АФС получается с использованием аэрофотосъемочного комплекса, в состав которого входят беспилотный летательный аппарат (БПЛА), камера и программное обеспечение для построения маршрута фотографирования для БПЛА.
      Снимки имеют хорошее геопространственное разрешение до 5 см/пикс, а благодаря встроенным функциям камер по корректировке дисторсии линз, присутствующие в них оптические искажения незначительны.
      Каждый снимок имеет соответствующие элементы внешнего ориентирования: углы камеры в момент снимка, высота и координаты.
   b. Данные о предыдущей таксации:
      Данные о предыдущей таксации является шейп-файл (.shp) с векторными географически-привязанными объектами, соответствующими группам леса, однородным по ряду признаков («выдел»), а также соответствующая ему таблица свойств этих групп (.dbf), таких как: возраст, порода, средняя высота, плотность, категория лесопатологического состояния и т. д.
2. Предобработка снимков.
   a. Приведение снимков к единой системе координат:
      Так как аэрофотоснимки, сделанные с БПЛА, имеют отличные от нуля углы ориентирования, то перед работой со снимками их необходимо привести к одной плоскости. Но, если рельеф сложный и углы фотографирования значительные, то применение простого аффинного[2] преобразования к снимку приведет к тому, что на плоскости окажутся пиксели изображения, которых не было бы при надирной съемке (ортогональной к горизонтальной плоскости в данной точке) (см. Рис. 2). В рамках работы была написана программа на языке Java с использованием библиотеки WorldWind[12], позволяющая приводить изображения к единой системе координат и строить пирамиду

разномасштабных изображений. Новизной реализованного метода является использование априорной модели рельефа SRTM [7], чтобы получать ортогональные проекции снимков с учетом положения пикселей изображения на сложном рельефе.

b. Построение пирамиды разномасштабных изображений (scalefree).
c. Переведение снимков в цветовое пространство HSV.
d. Выделение деревьев на снимке.

3. Формирование вектора входных параметров.
   a. Формирование максимально возможного набора признаков:
   Предлагается для классификации разности изображений, помимо базового подхода, основанного на спектральных характеристиках, использовать пространственный контекст, т.е. текстурные признаки на этапе сегментации в предобработке [5]. При использовании данного подхода ожидается уменьшение количества ложных срабатываний алгоритма и уменьшение зависимости точности алгоритма от разности вегетации растительности на симках (в рамках периода вегетации).
   b. Отсеивание взаимно коррелированных признаков.
   c. Определение весовых коэффициентов параметров.
4. Классификация деревьев.
   a. По критерию минимального расстояния до среднего классов в пространстве признаков с использованием метрики расстояния Махаланобиса, учитывающей корреляцию между признаками
   b. Дерево принятия решений. Так как оно позволяет использовать априорные экспертные правила классификации.
5. Объединение участков со схожими признаками
6. Определение областей существенных изменений

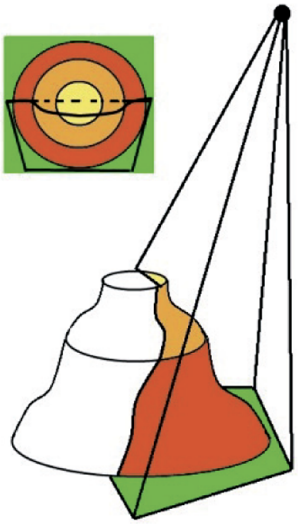


Рис. 2. Пирамида зрения камеры при наклонной съемке неровной поверхности

Заключение

Предполагается, что данный проект будет интересен организациями, которые занимаются лесным мониторингом. Для получения снимков они используют спутники и вертолеты гражданской авиации. Этот подход обладает следующими недостатками: высокая стоимость съемки, минимальное

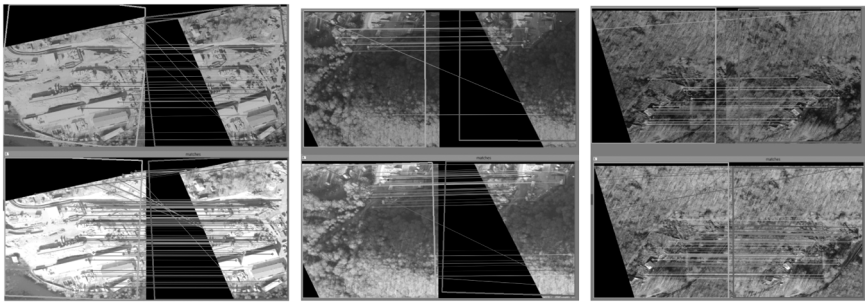


Рис. 3. На верхних рисунках приведен результат выравнивания снимков без использования информации о рельефе, на нижних — с использованием

(наилучшее возможное) геопространственное разрешение — десятки см/пикс. Экономически выгодным решением в некоторых случаях является использование БПЛА. При этом их использование позволяет получать более детальные снимки. Поэтому основной целью данной работы является показать преимущество использования снимков, полученных с БПЛА в целях лесопатологической таксации. Преимущество заключается в том, что на снимках высокого разрешения (пикс/см) информативность морфологических признаков дешифрирования повышается и позволяет более точно классифицировать объекты на снимках.

Использование данных о рельефе в решении задачи приведения снимков к единой системе координат дает выигрыш перед простым использованием аффинных преобразований. Гипотетически, максимальный выигрыш достигается при обработке снимков, полученных при больших углах съемки, а также при большей неоднородности рельефа. В качестве оценки качества для данного алгоритма предлагается интегрированная мера расхождения между одинаковыми областями на разных снимках по горизонтали и вертикали в абсолютных географических координатах, присвоенных этим областям алгоритмами. Для оценки истинного значения был использован алгоритм компьютерного зрения, основанный на поиске особых точек изображений методом SURF и методе клиппинга для определения общих областей на снимках. На верхних рисунках приведен результат выравнивания снимков без использования информации о рельефе, на нижних — с использованием (см. Рис. 3).

Литература

1. *Chi-Farn Chen*, N.-T. S.-B.-R.-Y. (2013). Multi-Decadal Mangrove Forest Change Detection and Prediction in Honduras, Central America, with Landsat Imagery and a Markov Chain Model. Remote Sensing.
2. *Dennis Morgan*, E. F. (2001). Aerial Mapping: Methods and Applications, Second Edition. CRC Press.

3. *Fabio Pacifici*, F. D. (2007). An Innovative Neural-Net Method to Detect Temporal Changes in High-Resolution Optical Satellite Imagery. IEEE Transactions on Geoscience and Remote Sensing.
4. *Richard J. Radke*, S. A.-K. (2005). Image Change Detection Algorithms: A Systematic Survey. IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 14, NO. 3.
5. *Zoltan Kato*, T.-C. P. (2006). A Markov random field image segmentation model for color textured images. Image and Vision Computing.
6. *В. М. Жирин*, к. Ц. (2004). Методы мониторинга вредителей и болезней леса. Москва: ВНИИЛМ.
7. *Дубинин, М*. (2004). Описание и получение данных SRTM. Получено из http://gis-lab.info/qa/srtm.html
8. *Дубинин, М*. (2006). Вегетационные индексы. Получено из gis-lab: http://gis-lab.info/qa/vi.html#sel=
9. *Дубинин, М*. (2006). Почвенная линия и ее определение. Получено из http://gis-lab.info/qa/vi-soilline.html
10. Определение состояния лесов. (29 12 2007 г.). Приложение № 2 к Приказу Рослесхоза № 523.
11. Приложение № 1 к Приказу Рослесхоза от 29.12.2007 № 523 «Руководство по проектированию, организации и ведению лесопатологического мониторинга». (б.д.).
12. WorldWind Java SDK, open sourse project, http://worldwind.arc.nasa.gov/java/

РАНДОМИЗАЦИЯ В ЗАДАЧАХ МАШИННОГО ОБУЧЕНИЯ

В. Н. Шац
Санкт-Петербург
E-mail: vlnash@mail.ru

Аннотация. В работе рассматривается дополнительные возможности индексного метода для решения задачи обучения в случае неколичественных признаков. Предложена схема рандомизации, при которой такие признаки трансформируются в количественные. Приведены примеры, в которых рандомизация резко повышает качество решения, снижая в ошибки до нескольких процентов.

Введение

Индексный метод машинного обучения реализует вычислительные принципы, которые свойственны биологическим системам [1, 2]. Новый метод отличается от существующих простотой и универсальностью алгоритма и приводит к снижению объема вычислений на несколько порядков.

Согласно методу значения признаков объектов описываются целыми числами, именуемыми индексы. Они находятся путем квантования или оцифровки признаков, и их набор описывает объект. Правило классификации объектов следует из сопоставления частот этих наборов для объектов различных классов.

Таким образом, анализ связей между группой объектов одного класса и исследуемым объектом опирается непосредственно на закон диалектики о всеобщей связи явлений [3]. Поэтому не вводятся допущения эвристического характера о близости значений той или иной детерминированной функции признаков для объектов одного класса. Тем самым обеспечивается большая достоверность результатов.

Расчеты показали, что процент ошибок для задач с количественными признаками близок к нулю [3]. Однако для других типов признаков качество решения является более низким. Работа посвящена анализу причин этого различия и способу повышения точности решения для неколичественных признаков.

Основные расчетные зависимости метода

Индексный метод в первую очередь служит для решения следующей типовой задачи обучения с учителем [2]. Задана выборка объектов q, которые характеризуются M признаками, и их распределение по непересекающимся классам $i \in (1, Mi)$. Требуется найти классы объектов тестовой выборки X. Согласно [1] этот метод применим также для типовой задачи обучения без учителя, отличающейся тем, что распределение объектов по классам неизвестно.

Кратко рассмотрим основные положения метода. Вводятся обозначения:

q_k^s – k-ый признак объекта номер $s \in (1, Ms)$, $k \in (1, M)$;

Ms — длина выборки;

$r_{s,k} = m$ — индекс признака q_k^s;

$m \in \{1, 2, ...\}$ — для неколичественного признака находится при оцифровке по номеру варианта в выборке;

$m \in \{1, 2, ..., n\}$ — для количественного признака находится при квантовании при числе интервалов квантования $n > 2$;

$l_{k,m}$ — количество объектов в упорядоченной паре (k, m) обучающей выборки;

$l_{k,m}^i$ — значение $l_{k,m}$, вычисленное только для объектов класса i;

T_i — количество объектов класса i;

ω_i — список объектов класса i;

$h_{k,m}^i = \dfrac{l_{k,m}^i}{T_i}$ — частота пары (k, m);

$g_{k,m}^i = \dfrac{l_{k,m}^i}{l_{k,m}}$ — относительная частота пары (k, m).

Поскольку частоты $h_{k,m}^i$ и $g_{k,m}^i$ дают оценки для условной вероятности $p(s \in \omega_i | r_{s,k} = m)$, то они одинаковы для объектов тестовой и обучающей выборки. Тогда для любого объекта s вероятность

$$p(s \in \omega_i) \cong f_s^i,$$

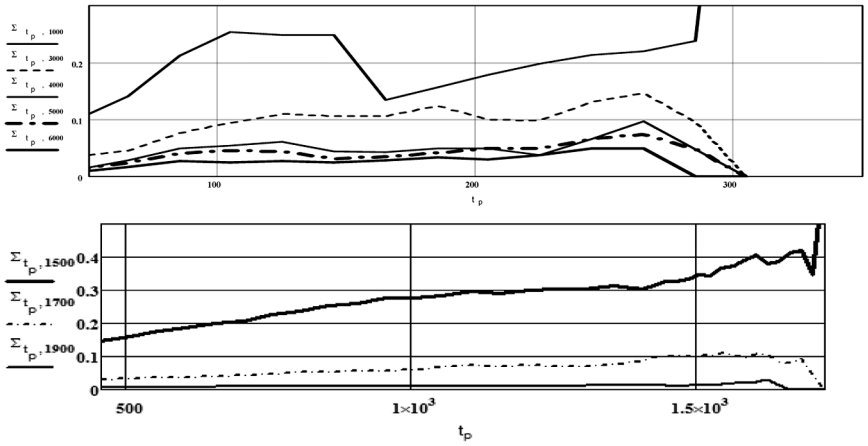


Рис. 1. Графики влияния n на точность решения при изменении соотношения длин обучающей и тестовой выборок

где средняя частота признаков объекта f_s^i в зависимости от особенностей выборки вычисляется по одной из следующих формул: $f_s^i = \frac{1}{M}\sum_{k=1}^{M} h_{k,m}^i$ или $f_s^i = \frac{1}{M}\sum_{k=1}^{M} g_{k,m}^i$.

Согласно методу максимума правдоподобия расчетный индекс класса объекта *s* равен

$$I(s) = \arg\max_{i\in(1,Mi)} f_s^i.$$

Таким образом, индексный метод предусматривает простейшую технологию классификации объекта. Объект моделируется конечным набором упорядоченных пар (*k*, *m*) и по данным обучающей выборки находятся частоты пар для каждого класса. Класс изучаемого объекта соответствует наибольшему значению средней частоты его элементов.

Особенности количественных признаков

Как следует из приведенных зависимостей, величины $h_{k,m}^i$ и $g_{k,m}^i$ дают частоту объектов группы из $l_{k,m}^i$ объектов, но используются только для объекта *s*, у которого $r_{s,k} = m$. Соответствующее осреднение характеристик вносит погрешность, которая снижается с уменьшением шага квантования Δ_k.

Преобразование количественных признаков в их индексы основано на предположении о том, что объект является элементом многомерного континуума. В этом случае признак может принимать любые значения на протяжении своего размаха. Тогда для любой последовательности *n* величины $l_{k,m}^i$ образуют последовательность случайных величин. Поэтом решение будут зависеть от вероятностной сходимости $I(s)_n$.

При $n \to \infty$ шаг Δ_k стремится к нулю, а $l_{k,m}^i$ достигнет предельного значения $a_{k,m}^i \sim 1$, равного количеству объектов класса *i*, имеющих одинаковые значения признака. Тогда количество ошибок обучения стремится к минимуму. В реальных задачах нижняя граница значений *n*, для которых ошибки обучения возникают не более, чем у 5% объектов, изменялась от 10 до 30000.

Отметим, что высокое качество обучения не гарантирует низкий процент ошибок при решении задачи, так как обучение только первый этап ее решения. На втором этапе нужно найти частоты $h_{k,m}^i$ или $g_{k,m}^i$ для всех пар (*k*, *m*) объекта тестовой выборки. Однако из *n* возможных значений *m* не более T_i имеют $l_{k,m}^i > 0$, у остальных $l_{k,m}^i = 0$. Поэтому при больших *n* указанные частоты образуют при каждом *k* разреженный вектор. Тогда может возникнуть ситуация, когда не определены частоты для той или иной пары (*k*, *m*). Создается эффект «разреженной» информации [4].

Преобразование неколичественных признаков

Можно считать, что выборка представлена некоторым экспертом, который разбил объекты на классы, выявил признаки, характеризующие объекты, установил для них шкалы измерений и измерил их с определенной погрешностью. Поэтому матрица данных «отображает» свойства реального набора объектов со случайной погрешностью [5].

Согласно методу, в случае количественных признаков значения q_k^s при каждом n заменяются их случайными приближениям $q_{k,m}$. По существу, метод относится к числу рандомизированных, когда отклонения от замеренных величин случайным образом учитывают комбинации погрешностей указанного отображения. Отметим, что для неколичественных признаков значения $l_{k,m}^i$ фиксированы и могут превышать половину Ms.

Вместе с тем, индексный метод основан на анализе частот признаков, которые представляют в вероятностном пространстве события, характеризующие свойства объекта. В отличие от метрических методов здесь величина признака играет роль идентификатора, поскольку учитываются только отношение эквивалентности между признаками, а не их величина.

Учитывая эти соображения, будем считать, что q_k^s, а также полученная при оцифровке величина $q_{k,m}^s$ дают «нечеткую» информацию о величине признака реального объекта. Фактическое значение признака, измеренное согласно принятой для него шкалы, соответствует случайной величине $\tilde{q}_k^s = m + v_s$. Здесь v — случайная величина, которая служит для «обогащения» данных [4]. Соотношение между реальными значениями признака и $\tilde{q}_k^s$ или q_k^s остается неопределенным.

Величину $\tilde{q}_k^s$ будем рассматривать как значение k-го количественного признака объекта. При квантовании получим $\tilde{q}_k^s \to q_{k,\tilde{m}}^s$, где $\tilde{m}$ — это значение индекса признака после рандомизации.

Таким образом, неколичественные признаки после оцифровке подвергаются рандомизации сначала при приведении признаков к количественному типу, а затем в ходе процесса квантования.

Численные результаты

Величина v варьировалась при расчетах. Устойчивые результаты были получены, когда в качестве нее принималась случайная величина, равномерно распределенная на отрезке (0,1).

Для иллюстрации эффекта рандомизации на рис. 1 приведены результаты решения задач «Car Evaluation» и «Haberman»s Survival» (соответственно верхний и нижний рисунки) с неколичественными признаками [6]. Первые t объектов заданных выборок рассматривались как обучающая выборка, остальные относились к тестовой.

Определялись частоты ошибок $\sigma_{t,n}$ и $\Sigma_{t,n}$ обучающей и тестовой выборки выборок при изменении t. Соответствующие графики строились по результатам вычислений для 17 и 36 вариантах значений t для верхнего и нижнего рисунков соответственно.

Расчеты показали, что для каждой задачи можно добиться предельно высокого качества обучения, когда $\sigma_{t,n} \sim 0$, не прибегая к рандомизации. Однако, качество решения при любых n сохраняется достаточно низким, поскольку $\Sigma_{t,n}$ находится в диапазоне 0.3–0.6.

Влияние рандомизации показывают графики изменения $\Sigma_{t,n}$. Из них следует, что благодаря рандомизации уровень ошибок снижается до $\Sigma_{t,n} \sim 0$.

Заключение

В работе проанализированы особенности механизма вычислений, который использует индексный метод при различных типах признаков. Показано, что в случае количественных признаков рандомизация органически входит в состав метода. Оказалось, что дополнительная рандомизация позволяет любой тип данных свести к количественному типу. В этом случае уменьшая шаг квантования можно снизить ошибки решения до приемлемого уровня.

Таким образом, индексный метод может обеспечить высокое качество решения задач обучения при любом типе признаков.

Литература

1. *Шац В. Н.* Двухуровневая метрика и новая концепция машинного обучения // Стохастическая оптимизация в информатике. 2013. Т. 9. Вып. 1. С. 128–143. www.math.spbu.ru/user/gran/optstoch.htm
2. *Шац В. Н.* Индексный метод машинного обучения // Сборник научных трудов научно-технической конференции «Нейроинформатика-2014». Ч. 1. М.: НИЯУ МИФИ, 2014. С. 21–30.
3. *Пугачев В. С.* Теория вероятностей и математическая статистика: Учеб. пособие. 2-е изд. М.: ФИЗМАТЛИТ, 2002. С. 496.
4. *Граничин О. Н.* Compressive Sensing. Рандомизация измерений и l_1 оптимизация // Стохастическая оптимизация в информатике. 2009. Т. 5. С. 3–23. www.math.spbu.ru/user/gran/soi5/Granichin5.pdf
5. *Джарратано Д., Райли Г.* Экспертные системы: Принципы разработки и программирования. 4-е изд.: Пер. с англ. М.: ООО «И. Д. Вильямс», 2007. С. 1152.
6. *Asuncion A., Newman D. J.* (2007). UCI Machine Learning Repository. Irvine CA: University of California, School of Information and Computer Science.

Параллельные алгоритмы и вэйвлетная обработка числовых потоков

Демьянович
Юрий Казимирович

д.ф.-м.н., профессор
заведующий кафедрой параллельных алгоритмов СПбГУ

МОДЕЛИРОВАНИЕ ГРЕБЕНЧАТЫХ СТРУКТУР ВЭЙВЛЕТОВ И ИХ РАСПАРАЛЛЕЛИВАНИЕ

Б. Д. Воробьёв

студент 543 группы математико-механического факультета СПбГУ,
Кафедры параллельных алгоритмов

E-mail: borisvorobyoff@yahoo.com

Аннотация. Рассмотрена интерференционная картина сплайн-вэйвлетного разложения в случае тесной гребенчатой структуры расположения элементарных гнезд на неравномерной сетке. Получены алгоритмы декомпозиции и реконструкции, дана оценка величины вэйвлетного потока в случае, когда исходный поток представляет собой последовательность отсчетов гладкого сигнала. Разработана компьютерная программа, реализующая полученные алгоритмы. Проведён анализ возможностей распараллеливания компьютерной программы.

Введение

Последнее время для обработки цифровых потоков широко используются вэйвлетные разложения. Исследовании вэйвлетных разложений числовых информационных потоков ведут к созданию гибкого аппарата сплайн-вэйвлетных аппроксимаций с учетом свойств гладкости и скорости изменения потоков. Для этого требуется использовать сплайновые аппроксимации различных порядков, неравномерные сетки, разложения со свойствами локализации в отдельных частях рассматриваемой области и применять гнездовые разложения для экономии вычислительных ресурсов с учетом возможностей параллельных вычислительных систем. Указанные аспекты нашли отражение в последних исследованиях (см. работы [1–2]). Вывод алгоритмов упомянутых разложений из аппроксимационных соотношений предопределяет высокое качество сжатия информации.

Цель данной работы состоит в создании компьютерной программы и анализе возможностей распараллеливания по формулам из работ [1–3].

Предварительные сведения

Для любого натуралного числа m введем обозначения:

$$J_m \stackrel{\text{def}}{=} \{0,1,\ldots,m\},\ J'_m \stackrel{\text{def}}{=} \{-1,0,1,\ldots,m\}.$$

Пусть N — натуральное число. На отрезке $[a, b]$ рассмотрим сетку

$$X: a = x_0 < x_1 < \ldots < x_{N-2} < x_{N-1} < x_N = b, \tag{1}$$

обозначим

$$S_j \stackrel{\text{def}}{=} (x_j, x_{j+1}) \cup (x_{j+1}, x_{j+2}),\ j \in J_{N-2},$$

$$G \stackrel{\text{def}}{=} (x_{0,} x_1) \cup (x_{1,} x_2) \cup \ldots \cup (x_{N-1}, x_N),$$

$$S_{-1} \stackrel{\text{def}}{=} (x_{0,} x_1),\ S_{N-1} \stackrel{\text{def}}{=} (x_{N-1}, x_N).$$

Система векторов $\{a_i\}_{i \in J'_m}$ пространства $\mathbb{R}^2$ называется полной цепочкой (см. [1]), если $det(a_{j-1}, a_j) \neq 0$ при $j \in J_m$.

Пусть $A = \{a_i\}_{i \in J'_{N-1}}$ — полная цепочка двумерных векторов.

Рассмотрим двухкомпонентную вектор-функцию $\varphi(t)$, непрерывную на отрезке $[a, b]$. Предположим, что ее компоненты линейно независимы на любом интервале $(a', b') \subset G$.

Зададим функции $\omega_j, t \in G, j \in J'_{N-1}$ с помощью аппроксимационных соотношений

$$\sum_{j \in J'_{N-1}} a_j \omega_j(t) = \varphi(t) \forall\, t \in G,$$

$$\omega_j(t) \equiv 0 \ \forall\, t \in G \setminus S_j,\ j \in J'_{N-1}.$$

Пусть s и r — натуральные числа, и $s < r$. Из сетки (1) удалим группу узлов с начётными номерами в количестве $r - s$; а именно, удалим узлы $x_{2s+1}, x_{2s+3}, \ldots, x_{2r-1}$, так что укрупнённая сетка имеет вид

$$\widetilde{X}: a = \widetilde{x}_0 < \widetilde{x}_1 < \ldots < \widetilde{X}_{\widetilde{N}-1} < \widetilde{X}_{\widetilde{N}} = b,$$

где $\tilde{N} = N - r + S$, а $\tilde{x}_i = x_i$ при $0 \le i \le 2s$, $\tilde{x}_{2s+i} = x_{2s+2i}$ при $2s \le i \le r - s$, $\tilde{x}_{s+r+i} = x_{2r+i}$ при $r - s \le i \le N - 2r$.

Каждый из удаленных узлов представляет собой элементарное (т.е. одноузловое) гнездо (см. [1]), причем каждая пара соседних гнезд разделена лишь одним узлом; такое расположение гнезд называем тесным.

Положим

$$\widetilde{S}_j \stackrel{\text{def}}{=} (\tilde{x}_j, \tilde{x}_{j+1}) \cup (\tilde{x}_{j+1}, \tilde{x}_{j+2}),\ j \in J_{N-2},$$

$$\widetilde{G} \stackrel{\text{def}}{=} (\tilde{x}_{0,} \tilde{x}_1) \cup (\tilde{x}_{1,} \tilde{x}_2) \cup \ldots \cup (\tilde{x}_{\widetilde{N}-1}, \tilde{x}_{\widetilde{N}}),$$

$$\widetilde{S}_{-1} \stackrel{\text{def}}{=} (\tilde{x}_{0,} \tilde{x}_1),\ \widetilde{S}_{\widetilde{N}-1} \stackrel{\text{def}}{=} (\tilde{x}_{\widetilde{N}-1}, \tilde{x}_{\widetilde{N}}).$$

Кроме того, введём обозначения

$$I^h \stackrel{\text{def}}{=} \{-1, 0, \ldots, 2s-2\},\ I^m \stackrel{\text{def}}{=} \{2s-1, \ldots, s+r-1\}$$

$$I^t = \{s+r, \ldots, \widetilde{N}-1\}.$$

Очевидно, что $I^h \cup I^m \cup I^t = J'_{\tilde{N}-1}$.

Рассмотрим цепочку векторов $\tilde{A} \stackrel{\text{def}}{=} \{\tilde{a}_{-1}, \tilde{a}_0, \ldots, \tilde{a}_{\tilde{N}-1}\}$, предполагая, что выполнено условие

(A) Цепочка векторов $\tilde{A}$ полная и справедливы соотношения

$$\tilde{a}_j = a_j \text{ при } j \in I^h, \ \tilde{a}_j = a_{2j+2s-1} \text{ при } j \in I^m, \ \tilde{a}_j = a_{j+r-s} \text{ при } j \in I^t.$$

В дальнейшем предполагаем, что условие (A) выполнено.

Совокупность $\{X, A, \tilde{X}, \tilde{A}\}$ будем называть *двухинтегральной гребенчатой структурой*.

Построим систему функций $\omega_j, t \in G, j \in J'_{\tilde{N}-1}$ с помощью аппроксимационных соотношений

$$\sum_{j \in J'_{\tilde{N}-1}} \tilde{a}_j \tilde{\omega}_j(t) = \varphi(t) \, \forall t \in \tilde{G},$$

$$\tilde{\omega}_j(t) \equiv 0 \, \forall t \in \tilde{G} \setminus \tilde{S}_j, \ j \in J'_{\tilde{N}-1}.$$

В дальнейшем считаем, что все рассматриваемые функции сужены на множество G.

Рассмотрим пространства

$$S \stackrel{\text{def}}{=} Cl_p L\{\omega_j\}_{j \in N-1} \text{ и } \tilde{S} \stackrel{\text{def}}{=} Cl_p L\{\tilde{\omega}_j\}_{j \in \tilde{N}-1},$$

где $L\{...\}$ — линейная оболочка функций, указанных в фигурных скобках, а Cl_p — замыкание в топологии поточечной сходимости. Заметим, что ввиду предположений относительно вектор-функции $\varphi(t)$ функции $\omega_j(t)$ линейно независимы на множестве G и, следовательно, являются базисом пространства S; отсюда имеем $\dim S = N+1$. Аналогично, вектор-функции $\tilde{\omega}_j(t)$ являются базисом пространства $\tilde{S}$, откуда $\dim \tilde{S} = \tilde{N}+1$.

Далее, в [3] рассматривается связь между функциями $\omega_j(t)$ и $\tilde{\omega}_j(t)$. В частности, доказывается теорема, утверждающая, что $\tilde{\omega}_j(t) \equiv \omega_j(t)$ при $j \in I^h$,

$$\tilde{\omega}_j(t) \equiv \omega_{r-s+j}(t) \text{ при } j \in I^t .$$

Вэйвлетное разложение

Рассмотрим систему функционалов $\{\tilde{g}_i\}_{i \in J'_{\tilde{N}-1}}$, биортогональную к системе $\{\tilde{\omega}_j\}_{j \in J'_{\tilde{N}-1}}$, $\langle \tilde{g}_i, \tilde{\omega}_j \rangle = \delta_{i,j}$ со свойством

$$supp\, \tilde{g}_i \subset [\tilde{x}_i, \tilde{x}_i + \varepsilon) \, \forall \varepsilon > 0, i \in J_{\tilde{N}-1}, \ supp\, \tilde{g}_{-1} \subset (\tilde{x}_0, \tilde{x}_0 + \varepsilon)$$

Кроме того, рассмотрим оператор P проектирования пространства S на подпространство $\tilde{S}$, задаваемый формулой

$$Pu \stackrel{\text{def}}{=} \sum_{j \in J'_{\tilde{N}-1}} \langle \tilde{g}_j, u \rangle \tilde{\omega}_j \; \forall \, u \in S \tag{2}$$

и введём оператор $Q = I - P$, где I — тождественный в S оператор.

Пространством вэйвлетов (всплесков) называется пространство $W = QS$. Итак, получаем прямое разложение $S = S + W$ — *сплайн-вэйвлетное разложение* пространства S.

Пусть $u \in S$; используя соотношения (2) и сплайн-вэйвлетное разложение S, получаем два представления элемента u:

$$u = \sum_{j \in J'_{N-1}} c_j \omega_j, \; u = \sum_{j \in J'_{\tilde{N}-1}} a_i \tilde{\omega}_i + \sum_{j \in J'_{N-1}} b_j \omega_j,$$

где $a_i \stackrel{\text{def}}{=} \langle \tilde{g}_i, u \rangle, b_j, c_j \in setR.$

Тогда *формулы реконструкции* имеют вид:

$$c_j = \sum_{i \in J'_{(\tilde{N})-1}} a_i p_{i,j} + b_j \; \forall \; j \in J'_{N-1}. \tag{3}$$

А *формулы декомпозиции*:

$$b_j = c_j - \sum_{i \in J'_{(\tilde{N})-1}} \sum_{j' \in J'_{N-1}} c_{j'} q_{i,j'} \, pi, j \; \forall \; j \in J'_{N-1}. \tag{4}$$

$$a_i = \sum_{j' \in J'_{N-1}} c_{j'} q_{i,j'} \; \forall \, i \in J'_{\tilde{N}-1}. \tag{5}$$

Здесь $p_{i,j}$ и $q_{i,j}$ — элементы матриц P и Q, носящих название матрица вложения и матрица продолжения, соответственно. Эти матрицы имеют особый вид, подробно процесс их построения описан в работе [3].

Введём вектор-столбцы $a \stackrel{\text{def}}{=} (a_{-1}, \ldots, a_{\tilde{N}-1})^T$, $b \stackrel{\text{def}}{=} (b_{-1}, \ldots, b_{N-1})^T$, $c \stackrel{\text{def}}{=} (c_{-1}, \ldots, c_{N-1})^T$ и перепишем формулы декомпозиции (4), (5) в матричном виде:

$$b = c - P^T Q C, \; a = Qc$$

Вектор a называется *основным потоком*, а вектор b — *вэйвлетным потоком* при сплайн-вэйвлетном разложении *исходного потока c*.

Ненулевые компоненты вэйвлетного потока b имеют вид:

$$b_{2q-1} = -u_{2q-1,2q-3} c_{2q-3} - u_{2q-1,2q-2} c_{2q-2} + c_{2q-1}$$

$$\text{при } \; q \in \{ s+1, s+2, \ldots, r \}, \tag{6}$$

$$b_{2q} = -u_{2q,2q-3}\, c_{2q-3} - u_{2q,2q-2}\, c_{2q-2} + u_{2q,2q-1}\, c_{2q-1}$$
$$\text{при } q \in \{ s+1, s+2, \ldots, r-1 \} , \qquad (7)$$

где $u_{i,j}$ — элементы матрицы $U \stackrel{\text{def}}{=} P^T Q$.

Интерференция на гребенчатой структуре

Далее в работе [3] доказывается следующая важная

Теорема: При $q \in \{ s+1, s+2, \ldots, r-1 \}$ для компонент вэйвлетного потока справедливы равенства

$$b_{2q} = \kappa_q b_{2q-1} , \qquad (8)$$

где κ_q от исходного потока c не зависит и определяется формулой

$$\kappa_q \stackrel{\text{def}}{=} \frac{det(a_{2q}, a_{2q+1})}{det(a_{2q-1}, a_{2q+1})} .$$

Линейная зависимость между компонентами числового потока называется интерференцией, а пропорциональность соседних компонент с коэффициентом, не зависящим от исходного потока, называется стоячей волной (см. [2]).

Доказанная теорема показывает, что порождение вэйвлетов первого порядка на двухинтервальной гребенчатой структуре $\{X, A, \tilde{X}, \tilde{A}\}$ сопровождается образованием системы стоячих волн; таким образом, размерность пространства вэйвлетных потоков совпадает с числом удаляемых узлов (т. е. с числом $r - s$).

Полученный результат означает, что при передаче или вэйвлетного потока b мы можем в два раза сократить его объём, отказавшись от запоминания каждого второго элемента. Для этого нам достаточно один раз вычислить значение κ_q и использовать его для восстановления элементов b_{2q} по сохранённым значениям элементов b_{2q-1} с помощью формулы (8).

Об аппроксимационных свойствах вэйвлетного потока

Предположим, что $\varphi \in C[a, b]$. Для непрерывности рассматриваемых сплайнов необходимо и достаточно, чтобы $a_j = \varphi_{j+1}$, здесь $\varphi_j \stackrel{\text{def}}{=} \varphi(x_j)$.

Пусть вектор-функция $\varphi(t)$ имеет вид $\varphi(t) = (1, f(t))^T$, где $f \in C[a, b]$. Тогда $det(\varphi(t'), \varphi(t'')) = f(t'') - f(t')$. Если $\varphi(t) = (1, t)^T$, то $f(t) = t$. Таким образом, в случае равномерной сетки $x_j = jh, h > 0$ имеем

$$det(a_i, a_j) = (j - i)h . \qquad (9)$$

Ранее в работе [3] доказывалось, что

$$u_{2q+1,2q-1}=\frac{det\left(a_{2q+1},a_{2q}\right)}{det\left(a_{2q-1},a_{2q}\right)}\ , \tag{10}$$

$$u_{2q+1,2q}=\frac{det\left(a_{2q-1},a_{2q+1}\right)}{det\left(a_{2q-1},a_{2q}\right)}\ . \tag{11}$$

Согласно формулам (6) — (7), (9) — (11) получаем

$$b_{2q-1}=c_{2q-3}-2c_{2q-2}+c_{2q-1}\ , \tag{12}$$

$$к_q=1/2\ . \tag{13}$$

Компьютерная реализация эффекта интерференции

Для демонстрации эффекта интерференции на гребенчатой структуре при тесном расположении гнезд была написана программа на языке C#. Принцип ее работы таков:

Шаг 1: Рассматриваем равномерную сетку $X: x_0<x_1<\ldots<x_N$, где $x_i=ih$, $h>0$, N — количество элементов. С помощью заранее заданной функции $F\in C^2$ порождаем элементы исходного потока c: $c_i=F(x_i)$. Выбираем числа s и r, задающие количество и расположение удаляемых из сетки узлов.

Шаг 2: Используя формулы (8), (12), (13), вычисляем элементы вэйвлетного потока b.

Шаг 3: Строим матрицу вложения P и матрицу продолжения Q (соответствующие формулы приведены в работе [3]).

Шаг 4: Вычисляем основной и вэйвлетный потоки (a и b, соответственно) в сплайн-вэйвлетном разложении исходного потока c, используя формул декомпозиции (4), (5).

Шаг 5: Вычисляем заново исходный поток c, используя формулы реконструкции (3).

Шаг 6: Проводим ряд сравнений, чтобы убедиться, что результаты, полученные при использовании в рассчетах эффекта интерференции, не отличаются от результатов, полученных при использовании традиционных формул декомпозиции и реконструкции. Для этого сравниваем элементы вэйвлетного потока b, вычисленные на шаге 2, с соответствующими элементами, полученными на шаге 4, а также элементы исходного потока c, порожденного на первом шаге, с соответствующими элементами, вычисленными на шаге 5.

Согласно формулам (8) и (12), при применении эффекта интерференции вычисление каждой пары элементов b_{2q-1}, b_{2q} может происходить независимо, что позволяет распараллелить этот процесс.

Литература

1. *Демьянович Ю. К.* Всплески и минимальные сплайны: Курс лекций. Спб., 2003.
2. *Демьянович Ю. К., Ходаковский В. А.* Введение в теорию вэйвлетов. Учеб. Пособие. Спб.: Петербургский государственный университет путей сообщения, 2008.
3. *Демьянович Ю. К., Дронь В.* О. О вэйвлетном гребне на нерегулярной сетке // Проблемы математического анализа, 2011.

ОБ АРХИТЕКТУРЕ ПАРАЛЛЕЛЬНОЙ СИСТЕМЫ[1]

Ю. К. Демьянович
Санкт-Петербургский государственный университет
Email: Yuri.Demjanovich@gmail.com

Аннотация. Рассмотривается общая схема архитектуры параллельной вычислительной системы, которая предназначена для реализации численных методов, основанных на аппроксимации пространствами локальных функций.

Математические модели многих часто встречающихся задач имеют локальный характер, ибо определяются локальными физическими законами (связанными с рассмотрением малых величин (длин, площадей, объемов и т. п.) и с последующим переходом к пределу). Применение подобных законов с соответствующими предельными переходами приводит к начально-краевым задачам математической физики. К таким задачам относится большинство суперсложных задач (задачи прогноза климата, состояния окружающей среды, состояния ядерных боезарядов и т. п.).

Наиболее эффективные методы решения упомянутых задач сводятся к построению приближения в конечномерном подпространстве локальных функций (см., например, [1]). Типичным примером таких методов является широко распространенный метод конечных элементов (МКЭ). Локальные функции применяются для обработки числовых информационных потоков, высокая плотность которых требует больших вычислительных мощностей (сюда относятся обработка двумерных и трехмерных информационных потоков, связанных с натурными экспериментами, с томографией, с обработкой информационных потоков от космических аппаратов, с передачей двумерных и трехмерных изображений, с распознаванием образов и т. п.). Методы отображения полученных конечномерных задач на вычислительную систему (т. е. способы программной реализации этих задач) рассматриваются во многих работах (см., например, [2–3]).

Для эффективного решения подобных задач необходимы вычислительные системы, архитектура которых согласована с применяемыми методами. Отметим в этой связи мультитредовую архитектуру (см. [3]); однако, прямой связи с применяемыми методами эта архитектура не имеет.

Цель данной работы рассмотреть общую схему архитектуры, которая согласована с численными методами, основанными на аппроксимации пространствами локальных функций.

[1] Работа частично поддержана грантами РФФИ 13-01-00096.

На некотором n-мерном дифференцируемом многообразии $\mathcal{T}$ (для удобства в качестве $\mathcal{T}$ можно рассмотреть, например, тор или сферу) введем клеточное подразделение $\mathcal{K} \overset{\text{def}}{=} \{K_j\}_{j \in J}$, т. е. рассмотрим непересекающиеся множества K_j, гомеморфные n-мерному открытому шару, объединение замыканий которых совпадает с $\mathcal{T}$,

$$\mathcal{T} = \bigcup_{j \in J} \bar{K}_j,$$

J — некоторое конечное множество индексов. Множества K_j называются *клетками*.

Пусть $\mathbb{N}$ — множество натуральных чисел, $\mathbb{N}_0 \overset{\text{def}}{=} \mathbb{N} \cup \{0\}$, $p \in \mathbb{N}_0$ и $s \in \mathbb{N}$.

Определение 1. Клетки называются инцидентными, если их замыкания пересекаются. Две клетки называются *p-инцидентными*, если пересечение их замыканий имеет размерность j, где $j \geq p, j \in \mathbb{N}0$.

Выделим в множестве $\mathcal{K}$ некоторое подмножество $\mathcal{K}^*$, среди клеток которого нет инцидентных, т. е.

$$\mathcal{K}^* = \{K \mid K \in \mathcal{K}\}, \text{ причем } \bar{K}' \cap \bar{K}'' = \varnothing \quad \forall K', K'' \in \mathcal{K}^*.$$

Определение 2. Клетки множества $\mathcal{K}^*$ будем называть *помеченными клетками*.

Если известно, что клетка помеченнная, то ее будем обозначать символом K^*.

Определение 3. Две клетки K' и K'' называются (p, s) — *связанными*, если существует последовательность клеток

$$\mathcal{C}_{(p,s)}(K', K'') \overset{\text{def}}{=} \{K' \overset{\text{def}}{=} K^{(0)}, K^{(1)}, K^{(2)}, \ldots, K^{(s)} \overset{\text{def}}{=} K''\}, \tag{1}$$

каждые две соседних клетки в которой p-инцидентны. Последовательность (1) называется (p, s) — *связующей цепочкой* (или просто (p, s) — *цепочкой*) для клеток K' и K'', число s — *длиной цепочки*, а число p называется *гарантированной мощностью цепочки*.

Для заданных клеток K' и K'' и фиксированной пары чисел (p, s) может быть несколько цепочек вида (1).

Множество всех цепочек вида (1) обозначим $\mathcal{C}_{(p,s)}(K', K'')$. Это множество не пусто разве лишь для конечного количества целочисленных значений s и p. Очевидно, что множество

$$\mathcal{C}^{(p)}(K', K'') \overset{\text{def}}{=} \bigcup_{s \in \mathbb{N}} \mathcal{C}_{(p,s)}(K', K''),$$

представляет собой множество цепочек одинаковой гарантированной мощности, связывающих клетки K' и K'', а множество

$$\mathcal{C}_{(s)}(K', K'') \overset{\text{def}}{=} \{\mathcal{C}_{(p, s)}(K', K'') | \, p \in \mathbb{N}_0\}$$

является множеством цепочек одинаковой длины (возможно, различных гарантированных мощностей), связывающих клетки K' и K''.

Введем числа $s^{(p)}_{\min}(K', K'')$ и $s^{(p)}_{\max}(K', K'')$ по формулам

$$s^{(p)}_{\min}(K', K'') \overset{\text{def}}{=} \min_{s \in \mathbb{N}} \{s | \mathcal{C}_{(p, s)}(K', K'') \neq \varnothing\},$$

$$s^{(p)}_{\max}(K', K'') \overset{\text{def}}{=} \max_{s \in \mathbb{N}} \{s | \mathcal{C}_{(p, s)}(K', K'') \neq \varnothing\},$$

Число $s^{(p)}_{\min}(K', K'')$ называется *минимальной p-удаленностью* клеток K' и K'', а число $s^{(p)}_{\max}(K', K'')$ — *максимальной p-удаленностью* клеток K' и K''; полезными оказываются также числа

$$s_{\min}(K', K'') \overset{\text{def}}{=} \min_{p \in \mathbb{N}_0} s^{(p)}_{\min}(K', K''),$$

$$s_{\max}(K', K'') \overset{\text{def}}{=} \max_{p \in \mathbb{N}_0} s^{(p)}_{\max}(K', K''),$$

называемые минимальной и максимальной удаленностью клеток K' и K'' соответственно.

Отметим некоторые свойства p-инцидентности и (p, s) — связанности. Пусть $p_1 \leq p$. Тогда

1) если две клетки p-инцидентны, то они и p_1-инцидентны,
2) если цепочка (1) — (p, s) — связующая для клеток K' и K'', то она является и (p_1, s) — связующей для этих клеток,
3) если две клетки (p, s) — связаны, то они и (p_1, s) — связаны,
4) если две клетки $(p, 1)$ — связаны, то они p-инцидентны,
5) справедливы включения

$$\mathcal{C}_{(p, s)}(K', K'') \subset \mathcal{C}_{(p_1, s)}(K', K''), \qquad \mathcal{C}^{(p)}(K', K'') \subset \mathcal{C}^{(p_1)}(K', K'').$$

Архитектуру предлагаемой вычислительной системы согласуем с клеточным подразделением следующим образом.

Будем считать, что каждая клетка представляет собой фрагмент памяти (далее термины «фрагмент памяти» и «клетка» отождествляем), а коммуникации между (p, s) — связанными фрагментами памяти K' и K'' происходят по каналам ассоциированным с (p, s) — связующими цепочками длины s и мощности p.

Как известно, процессор (вместе с соответствующим оборудованием) можно рассматривать как фрагмент памяти с большим быстродействием. Будем считать, что каждая помеченная клетка представляет собой процессор

(дальше термины «процессор» и «помеченная клетка» отождествляются). Если помеченная клетка (процессор) фиксирована, то по отношению к ней любая другая клетка может рассматриваться как кэшпамять. Более традиционно каждой помеченной клетке K^* соотнести некоторую группу $\mathcal{G}(K^*)$ «близлежащих» к ней (в смысле упомянутой топологии) непомеченных клеток, например, клеток, связанных с K^* цепочками длины $s \le s_0(K^*)$, где $s_0(K^*)$ — некоторое фиксированное число. Совокупность клеток $\mathcal{G}(K^*)$, связанных цепочками длины s с клеткой K^*, $s \le s_0(K^*)$, называется кэшем s-го уровня для процессора K^*.

Определение 4. Объединение $M(K^*) \stackrel{\text{def}}{=} \mathcal{G}(K^*) \cap \{K^*\}$ называется *вычислительным модулем*, а $\mathcal{G}(K^*)$ — его кэшем. Объединение всех вычислительных модулей называется *вычислительной системой*.

Существенным моментом, отличающим предлагаемую архитектуру от обычно рассматриваемых, является то, что кэши различных вычислительных модулей могут пересекаться; в частности, априори фиксированная клетка K может принадлежать кэшам нескольких вычислительных модулей.

Определение 5. Число вычислительных модулей, которым принадлежит данный фрагмент памяти $K \in \mathcal{K}$, называется кратностью накрытия этого фрагмента и обозначается $\varkappa(K)$. Число $\varkappa_0 \stackrel{\text{def}}{=} \max_{K \in \mathcal{K}} \varkappa(K)$ называется *кратностью вычислительной системы*.

Вычислительная система с рассмотренной архитектурой называется *локально кратной вычислительной системой*.

Если кратность накрытия можно менять добавлением или удалением вычислительных модулей, то вычислительную систему можно назвать *локально кратной вычислительной системой с изменяемой кратностью накрытия*.

Закончим изложение этого подхода замечанием о том, что для практических целей было бы важно иметь однородную локально кратную вычислительную систему, т. е. такую у которой кратность накрытия одинакова:

$$\varkappa(K) = \varkappa_0 \qquad \forall K \in \mathcal{K}.$$

Анализ численного решения сложных задач показывает, что при разработке архитектуры суперкомпьютеров очень важно учитывать свойства алгоритмов решения систем линейных алгебраических уравнений. Ввиду локального характера всех упомянутых задач требования к архитектуре суперкомпьютеров могут быть в значительной степени конкретизированы; они дают архитектурные решения, которые могут привести к созданию суперкомпьютеров, занимающих промежуточное положение между компьютерами с разделяемой памятью и компьютерами с распределенной памятью: память разделена на фрагменты, к каждому фрагменту имеют прямой доступ несколько вычислительных модулей; в свою очередь, каждый вычислитель-

ный модуль имеет прямой доступ к нескольким фрагментам памяти. Такая архитектура полностью соответствует локальным аппроксимациям, методам конечных элементов, сплайнам и вэйвлетам, применяемым при решении перечисленных во введении суперсложных задач; можно надеятсься, что этот подход приведет к существенному повышению реальной производительности при решении упомянутых задачах.

Литература

1. *Демьянович Ю. К.* Локальная аппроксимация на многообразии и минимальные сплайны. СПб., 1994. 356 с.
2. *Эндрюс Г. Р.* Основы многопоточного, параллельного и распределенного программирования. М., 2003. 512 с.
3. *Корнеев В. В.* Вычислительные системы. М., 2004. 512 с.
4. *Демьянович Ю. К.* Проблемы распараллеливания в некоторых локальных задачах // Приборостроение. 2009. Т. 52. № 10. С. 41–49.

КОМПЬЮТЕРНАЯ РЕАЛИЗАЦИЯ ВЫЧИСЛЕНИЙ НЕКОТОРЫХ ЭЛЕМЕНТАРНЫХ ФУНКЦИЙ

В. С. Богданов
студент 5-го курса кафедры информатики СПбГУ
E-mail: vladimir.bogdano@mail.ru

М. Ю. Быц
студент 5-го курса кафедры информатики СПбГУ
E-mail: maxim.90svet@gmail.com

Научный руководитель:
Ю. К. Демьянович

Введение

В настоящее время, ключевыми задачами, решающимися на многоядерных суперкомпьютерах с параллельной архитектурой, являются: расчёт климатической модели, решение систем трёхмерных нелинейных уравнений с частными производными, уравнений с большим количеством неизвестным и т. д. Каждая из этих задач требует многократного вычисления элементарных функций и от того, насколько оно будет эффективно осуществляться, зависит общая производительность системы.

В данной работе, проанализированы ключевые подходы к вычислению элементарных тригонометрических и обратных тригонометрических функций. Исследованы алгоритмы, представляющие тот или иной подход к вычислению, с точки зрения скорости вычислений и их устойчивости к ошибкам округления.

В работе рассмотрены следующие подходы:

- аппроксимация с помощью ряда Тейлора;
- приближение цепными дробями;
- итерационные методы (метод Вольдера).

Методы

Аппроксимация с помощью рядов Тейлора

Пусть функция $f(x)$ имеет производные в окрестности точки x_0 *всех* порядков до $(n-1)$ включительно, а также (n) — ую производную в самой точке x_0. Тогда для функции $f(x)$ можно составить полином следующего вида [2]:

$$p_n(x) = f(x_0) + \frac{f'(x_0)}{1!}(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \ldots + \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n.$$

Для его записи используют сокращенную форму:

$$p_n(x) = \sum_{i=0}^{n} \frac{f^{(i)}(x)}{i!}(x - x_0)^i.$$

Полином $p(x)$ даёт лишь приближение функции $f(x)$, таким образом, требуется оценка погрешности:

$$r_n = f(x) - p_n(x).$$

Для вычисления методической погрешности тригонометрических функций использована Лагранжева форма дополнительного члена формулы Тейлора:

$$r_n(x) = \frac{f^{(n+1)}(\theta x)}{(n+1)!}x^{n+1}.$$

Для вычисления методической погрешности обратных тригонометрических функций использована интегральная форма дополнительного члена формулы Тейлора:

$$r_n(x) = \int_{x_0}^{x} \frac{f^{(n+1)}(t)}{n!}(x - t)^n \, dt.$$

Для реализации вычисления тригонометрических функций требуется найти оценку для $r_n(x_0)$. Например, для функции $\sin(x)$, учитывая её ограниченность, можно провести следующую грубую оценку методической погрешности:

$$0 < r_n(x_0) < \frac{|x_0|^{2n+1}}{(2n+1)!}.$$

Метод Вольдера

С помощью методов «цифра за цифрой», вычисление значений элементарных тригонометрических функций сводится к последовательному выполнению двух операции: сложения и сдвига [1].

В настоящее время существует множество алгоритмов, представляющих метод «цифра за цифрой». В данной работе мы остановили свой выбор на алгоритме Вольдера, так-как при том, что он разработан ещё во второй половине XX века, интерес к нему в научном сообществе до сих пор не ослабевает.

Рассмотрим плоскость A на которой задана система координат OXY. Отложим на плоскости единичную окружность с центром в начале координат. Требуется найти координаты вектора $\vec{v}$, который образует с положительным направлением оси OX угол α ($0° \le \alpha \le 90°$).

Отложим на оси *OX* единичный вектор $\vec{w}_0$, и путём поворотов его на фиксированные углы (как по, так и против часовой стрелки) попытаемся совместить с вектором $\vec{v}$. Возможность полного совмещения зависит, как от величины угла α, так и от набора фиксированных углов, задающих ротации вектора $\vec{w}_0$.

Координаты вектора $\vec{w}_{i+1} = (x_{i+1}, y_{i+1})$, получающегося из вектора $\vec{w}_0$, путём последовательного его поворота на углы $\alpha_0, \alpha_1, \ldots, \alpha_i$:

$$x_{i+1} = x_i \cos\alpha_i - y_i \sin\alpha_i,$$

$$y_{i+1} = y_i \cos\alpha_i + x_i \sin\alpha_i,$$

где $\vec{w}_i = (x_i, y_i)$, вектор который получается из вектора $\vec{w}_0$, путём поворота на углы $\alpha_0, \alpha_1, \ldots, \alpha_{i-1}$.

Мы можем таким образом подобрать углы $\alpha_0, \alpha_1, \ldots, \alpha_{n-1}$, чтобы угол между векторами $\vec{v}$ и $\vec{w}_i$ уменьшался при возрастании $i = 0 \ldots n$.

Выбирая углы $\alpha_i = \pm \operatorname{arctg}(2^{-i})$, знак которых зависит от направления вращения вектора, а само направление на *j*-ом шаге определяется разностью углов α и $\sigma_0\alpha_0 + \sigma_1\alpha_1 + \ldots + \sigma_{j-1}\alpha_{j-1}$, получим:

$$\sigma_i = \operatorname{sign}\{z_i\},$$

$$x_i = \cos\alpha_i (x_i - \sigma_i \times 2 \times y_i),$$

$$y_{i+1} = \cos\alpha_i (y_i + \sigma_i \times 2^{-i} \times x_i),$$

$$z_{i+1} = z_i - \sigma_i \times \operatorname{arctg}(2^{-i}).$$

где $z_0 = \alpha$, $\sigma_0 = 1$.

Алгоритмы

Аппроксимация с помощью рядов Тейлора

Для $\arcsin(x)$ разложение в формулу Тейлора при $x_0 = 0$ примет следующий вид:

$$f(x) \approx 0 + x\left(1 + x\left(0 + x\left(\frac{1}{6} + x\left(0 + \ldots + \frac{(2n)!}{4^n \cdot (n!)^2 \cdot (2n+1)} x\right)\right)\right)\right),$$

где

$$a_0 = 0,\ a_1 = 1,\ a_2 = 0,\ a_3 = \frac{1}{6},\ \ldots,\ a_n = \frac{(2n)!}{4^n \cdot (n!)^2 \cdot (2n+1)}.$$

Предварительно рассчитав коэффициенты $\alpha_0, \alpha_1, \ldots, \alpha_n, \ldots$, внесём ненулевые их значения в память ЭВМ. Будем рассматривать промежуток аппроксимации [0; 0,8); разбив его на восемь равных частей: [0; 0,1), [0,1; 0,2), …, [0,7; 0,8).

Входные данные: x_a — точка, в которой вычисляется значение функции $f(x)$, P_0 — требуемая погрешность вычислений.

Исходные данные: $A_{f(x)}[i]$ — одномерный массив, содержащий коэффициенты разложения функции $f(x)$ в ряд Тейлора в окрестностях точки $x_0 = 0$; $B_{f(x)}[j, k]$ — двумерный массив, содержащий количество итераций m, необходимых для достижения заданной точности P_0, где $j \in \boldsymbol{Z}$, $1 \le j$ и соответствует требуемой погрешности ($10^{-1}, 10^{-2}, \ldots, 10^{-j}, \ldots$), а $k \in \boldsymbol{Z}$, $0 \le k \le 7$ и соответствует первому десятичному знаку после запятой аргумента функции $f(x)$: $[x_a \times 10]$.

Выходные данные: *res* — значение функции $f(x)$ в точке x_a, рассчитанное с погрешностью P_0.

Алгоритм 1. Вычисление $\arcsin(x)$[1]

A1: $n = B_{\arcsin(x)}\left[P_0, [x_a \times 10]\right]$;

A2: $mul = x_a \times x_a$;

A3: $i = 0$; $res = A_{\arcsin(x)}[n]$;

A4: $i = i + 1$; $(i = n) \Rightarrow (\rightarrow$ A6);

A5: $res = res \times mul + A_{\arcsin(x)}[n - i]$; $\rightarrow$ A4;

A6: $res = res \times x_a$; $\rightarrow 0$.

Разобьём формулу Тейлора на две части следующим образом, если n четное:

$$f(x) - o(x^n) = \left(a_0 + \ldots + a_n x^n\right) + \left(a_1 x + \ldots + a_{n-1} x^{n-1}\right),$$

и таким:

$$f(x) - o(x^n) = \left(a_0 + \ldots + a_{n-1} x^{n-1}\right) + \left(a_1 x + \ldots + a_n x^n\right),$$

если n нечетное. Продолжим разбиение для случая, когда n четное (для нечетного n разбиение проводится аналогично):

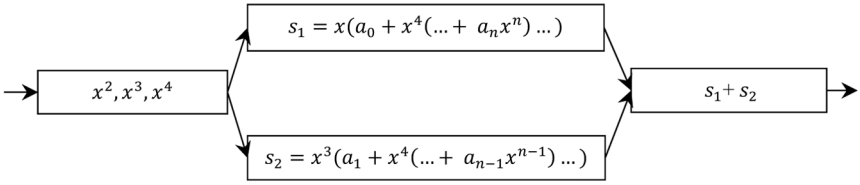
$$f(x) \approx \left(a_0 + x^2(a_2 + \ldots + a_n x^2)\ldots\right) + x\left(a_1 + x^2(a_3 + \ldots + a_{n-1} x^2)\ldots\right).$$

Для $\arcsin(x)$ в соответствии формула примет следующий вид:

$$f(x) \approx x\left(a_0 + x^4(a_2 + \ldots + a_n x^4)\ldots\right) + x^3\left(a_1 + x^4(a_3 + \ldots + a_{n-1} x^4)\ldots\right).$$

[1] Алгоритм, как и следующий, написан на псевдокоде. $(a) \Rightarrow (b)$ — означает условный переход: если верно a, выполняется b; $\rightarrow An$ — означает переход к шагу An; $\rightarrow 0$ — завершение алгоритма.

Теперь вычисление можно разбить на два потока по следующей схеме:



Метод Вольдера

Входные данные: x_a — значение аргумента функции $f(x)$, P_0 — требуемая погрешность вычислений.

Исходные данные: $B_{f(x)}[j]$ — одномерный массив, содержащий количество итераций m, требуемых для достижения указанной точности P_0 при вычислении функции $f(x)$, где $j \in \boldsymbol{Z}$, $1 \le j$ и соответствует требуемой погрешности; $T[k]$ — одномерный массив, содержащий последовательность углов $\alpha_0, \alpha_1, \ldots, \alpha_k, \ldots$: $\alpha_k = \operatorname{arctg}(2^{-k})$, где $k \in \boldsymbol{Z}$, $0 \le k$.

Выходные данные: z — содержит значение $\operatorname{arctg}(x_a)$, рассчитанное с точностью P_0.

Алгоритм 2. Вычисление $\operatorname{arctg}(x)$ ($\gg$ — битовый сдвиг налево).

A1: $n = B_{\operatorname{arctg}(x)}[P_0]$; $i = -1$;

A2: $z = 0$; $x = 1$; $y = x_a$;

A3: $i = i + 1$; $\big(i = (n-1)\big) \Rightarrow \big(\rightarrow 0\big)$;

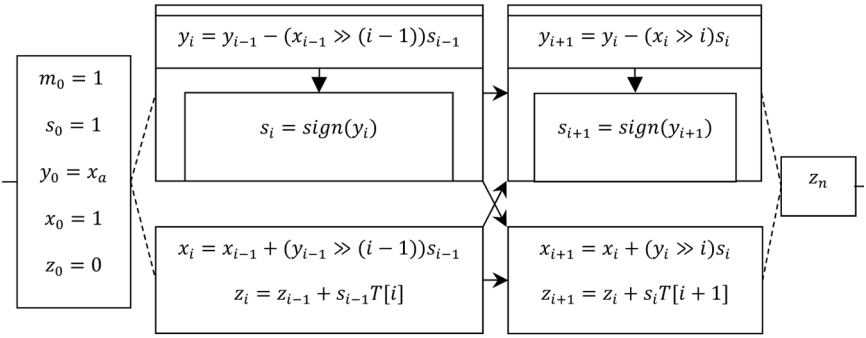
A4: $s = sign(y)$;

A5: $mid = x$; $x = x + s \times \big(y \gg i\big)$;

A6: $y = y - s \times \big(mid \gg i\big)$;

A7: $z = z + s \times T[i]$; $\rightarrow$ A3.

Распараллеливание осуществляется следующим образом:



Погрешность алгоритмов

Погрешности алгоритмизации возникают при реализации используемого алгоритма на вычислительной машине. Они могут различаться как по величине, так и по типу. Разделяют их на три группы:

1) методические;
2) инструментальные;
3) трансформированные.

Наиболее достоверными оценками погрешностей на вычислительных машинах являются их статистические характеристики[6]. Это обусловлено многими факторами, среди которых выделяются 2 основных:

1) статистический характер возникновения погрешностей;
2) результирующая погрешность вычислений состоит из большого числа независимых компонент.

Таким образом, функция распределения результирующей погрешности имеет вид, близкий к функции нормального распределения. Тогда будет выполняться следующее соотношение: $\Delta \leq 3\sigma$, где Δ — максимальная, а σ — среднеквадратическая оценка погрешности. Такая оценка будет выполняться с вероятностью 99,73%.

Заключение

На основе описанных выше методов были разработаны алгоритмы и реализованы в двух версиях: последовательной и параллельной. Проведена оценка их эффективности с точки зрения скорости вычислений и устойчивости к ошибкам. Последовательная версия алгоритма Тейлора оказалась немного медленнее стандартных средств C++, однако при этом, стоит отметить, что алгоритм гарантированно дает значение с требуемой точностью, чего нельзя сказать о встроенных функциях. Параллельная версия показала себя не эффективно для малой требуемой точности, так-как такая точность достигалась небольшим количеством итераций. Тем не менее, при возрастании числа итераций имеется прирост скорости работы сравнительно с последовательной версией. Так же с положительной стороны можно отметить, что существенная часть алгоритма Тейлора может выполняться независимо, таким образом, есть возможность распараллеливания на несколько потоков, при большом количестве итераций.

Алгоритм на основе метода Вольдера, оказался немного медленнее алгоритма Тейлора, но при этом вычисляется сразу две функции (например, $\sin x$ и $\cos x$), что позволяет эффективно применять его в задачах вроде поворота координат. Алгоритм устойчив к ошибкам округления. Существенным недостатком является нецелесообразность распараллеливания алгоритма, так как на каждом шаге двум процессорам приходится ждать друг друга и пе-

редавать сообщения. Это связано с тем, что рекуррентные соотношения для x и y зависимы.

Литература

1. *Байков В. Д., Смолов В. Б.* Аппаратурная реализация элементарных функций в ЦВМ. Л.: Изд-во Ленингр. ун-та, 1975.
2. *Фихтенгольц Г. М.* Основы математического анализа. Т. II. М.: Наука, Главная редакция физико-математической литературы, 1968.

О КАЛИБРОВОЧНЫХ СООТНОШЕНИЯХ ДЛЯ *B*φ-СПЛАЙНОВ ЧЕТВЕРТОГО ПОРЯДКА[1]

Мирошниченко И. Д.

ст. преп. кафедры параллельных алгоритмов,
Санкт-Петербургский государственный университет
E-mail: irina_mir_@mail.ru

Аннотация: Рассматриваются калибровочные соотношения для *B*φ-сплайнов четвертого порядка сплайновых пространств, получающихся при удалении группы узлов, а также соответствующие формулы декомпозиции и реконструкции. Результаты могут быть использованы для разработки параллельных методов реализации сплайн-вэйвлетных разложений для рассмотриваемых пространств.

1. Введение

В работе авторов (см. [1]) рассматривались необходимые и достаточные условия гладкости (вообще говоря, неполиномиальных) сплайнов четвертого порядка, была доказана единственность пространства сплайнов максимальной гладкости и их вложенность на измельчающихся сетках, найдены соответствующие калибровочные соотношения.

В работе (см. [2]) была установлена независимость вэйвлетного разложения от порядка удаления узлов. Однако, реализации процесса последовательного удаления большого количества узлов (с использованием плавающей арифметики) приводит к быстрому накоплению ошибок округления, поэтому такой подход оправдан лишь при вычислениях в реальном масштабе времени, когда запаздывание недопустимо.

В данной работе, как и в работе [3] рассматривается ситуация, когда возможно запаздывание при обработке поступающего числового потока. В этом случае возможно одновременное удаление групп последовательных узлов, что позволяет применить распараллеливание процесса вычислений. В предлагаемой статье продолжено рассмотрение калибровочных соотношений.

2. Основные соглашения

Пусть задана полная цепочка двумерных векторов $A_N \overset{def}{=} \{a_i\}_{i \in J'_{N-1}}$, определены функции $\omega_j(t)$, $t \in G_N$, $j \in J'_{N-1}$ с помощью аппроксимационных соотношений (согласно определениям, приведённым в работе [2])

$$\sum_{j \in J'_{N-1}} a_j \omega_j(t) = \varphi(t), \forall t \in G_N \qquad \omega_j(t) \equiv 0, \forall t \in G_N / S_j, j \in J'_{N-1},$$

[1] Работа частично поддержана грантами РФФИ 10-01-00297 и 10-01-00245.

то есть на множестве $G_N N$ определены функции $\omega_{-1}(t)$, $\omega_j(t)$, $\omega_{N-1}(t)$.

Рассмотрим пространтво $B\varphi$-сплайнов четвертого порядка $\mathbb{S}_4^*(X, \varphi) \subset C^3(\alpha, \beta)$ на конечном или бесконечном интервале (α, β) вещественной оси R^1.

Обозначим $\mathfrak{S}_4(X, \varphi)$ множество пространств сплайнов четвертого порядка при фиксированной сетке X и при фиксированной вектор-функции $\varphi(t)$ на множестве **A** всех полных цепочек A_N.

$$\mathfrak{S}_4(X, \varphi) = \{S_4(X, A, \varphi) | \forall A_N \in A\}.$$

Справедливы следующие теоремы.

Теорема 1. Во множестве $\mathfrak{S}_4(X, \varphi)$ существует единственное пространство класса $C^3(\alpha, \beta)$; таким пространством является пространство $S_4^*(X, \varphi)$.

Теорема 2. Каждая функция $\omega_j^*(t)$ определяется значениями вектор-функции $\varphi(t)$ на множестве $\mathbf{supp}\,\omega_j^*$.

3. О калибровочных соотношениях

Укрупним исходную сетку X, выбросив узел $x_k + 1$.

Положим $\tilde{x}_j = x_j$ при $j \le k$, $\tilde{x}_j = x_{j+1}$ при $j \ge k+1$ и рассмотрим новую сетку $\tilde{X} \overset{\text{def}}{=} \{\tilde{x}_j\}_{j\in\mathbb{Z}}$ $\tilde{X}: \ldots < \tilde{x}_{-1} < \tilde{x}_0 < \tilde{x}_1 < \ldots$.

Положим $\tilde{\varphi}_j \overset{\text{def}}{=} \varphi'(\tilde{x}_j)$, $\tilde{\varphi}_j' \overset{\text{def}}{=} \varphi'(\tilde{x}_j)$ и определим векторы $\tilde{d}_k \in R^5$

$$\tilde{d}_k^T x \overset{\text{def}}{=} \det(\tilde{\varphi}_k, \tilde{\varphi}_k', \tilde{\varphi}_k'', \tilde{\varphi}_k''', x) \quad \forall k \in \mathbb{Z} \quad \forall x \in \Re^5$$

Введем векторы $\tilde{a}_j^*$ с помощью символического определителя

$$\tilde{a}_j^* \overset{def}{=} \begin{pmatrix} \tilde{\varphi}_{j+1} & \tilde{\varphi}_{j+1}'' & \tilde{\varphi}_{j+1}' & \tilde{\varphi}_{j+1}''' \\ \tilde{d}_{j+2}^T\tilde{\varphi}_{j+1} & \tilde{d}_{j+2}^T\tilde{\varphi}_{j+1}' & \tilde{d}_{j+2}^T\tilde{\varphi}_{j+1}'' & \tilde{d}_{j+2}^T\tilde{\varphi}_{j+1}''' \\ \tilde{d}_{j+3}^T\tilde{\varphi}_{j+1} & \tilde{d}_{j+3}^T\tilde{\varphi}_{j+1}' & \tilde{d}_{j+3}^T\tilde{\varphi}_{j+1}'' & \tilde{d}_{j+3}^T\tilde{\varphi}_{j+1}''' \\ \tilde{d}_{j+4}^T\tilde{\varphi}_{j+1} & \tilde{d}_{j+4}^T\tilde{\varphi}_{j+1}' & \tilde{d}_{j+4}^T\tilde{\varphi}_{j+1}'' & \tilde{d}_{j+4}^T\tilde{\varphi}_{j+1}''' \end{pmatrix}$$

Рассмотрим аппроксимационные соотношения

$$\sum_{j'\in\mathbb{Z}} \tilde{a}_j^* \tilde{\omega}_{j'}^*(t) \equiv \varphi(t) \quad \forall t \in (\alpha, \beta); \quad \operatorname{supp}\tilde{\omega}_j = [\tilde{x}_j, \tilde{x}_{j+5}] \quad \forall j \in \mathbb{Z}$$

Пространство *B*φ-сплайнов на новой сетке обозначим

$$\mathbb{S}_4^*(\tilde{X},\varphi) \overset{def}{=} Cl_p L(\{\tilde{\omega}_j^*\}_{j\in\mathbb{Z}}).$$

Теорема 3. Если сетка $\tilde{X}$ столь мелкая, что цепочка $\tilde{A}^* \overset{def}{=} \{\mathrm{a}^*\}_{j\in\mathbb{Z}}$ полная, то пространство *B*φ-сплайнов на сетке *X* содержит пространство Bφ-сплайнов на сетке $\tilde{X}$:

$$\mathrm{S}_4^*(\tilde{X},\varphi) \subset \mathrm{S}_4^*(X,\varphi).$$

Из этой теоремы следует справедливость тождеств, которые называют калибровочными соотношениями:

$$\tilde{\omega}_i(\mathrm{t}) \equiv \sum_{j\in\mathbb{Z}} p_{i,j}\omega_j(\mathrm{t}) \qquad \forall\,\mathrm{t}\in(\alpha,\beta) \quad \forall\,\mathrm{i}\in\mathbb{Z},$$

где p_{ij} при $I \le k-5$ и $i \ge k+1$ определяются из соотношений

$$\tilde{\omega}_j^*(\mathrm{t}) \equiv \omega_j^*(\mathrm{t}), \quad \tilde{\mathrm{a}}_l^* \equiv a_j^* \text{ при } j \le k-5,$$

$$\tilde{\omega}_j^*(\mathrm{t}) \equiv \omega_{j+1}^*(\mathrm{t}), \quad \tilde{\mathrm{a}}_l^* \equiv a_{j+1}^* \text{ при } \mathrm{j}\ge\mathrm{k}+1,$$

$$\tilde{a}_{k-4}^*\tilde{\omega}_{k-4}^*(\mathrm{t}) + \tilde{a}_{k-3}^*\tilde{\omega}_{k-3}^*(\mathrm{t}) + \tilde{a}_{k-2}^*\tilde{\omega}_{k-2}^*(\mathrm{t}) + \tilde{a}_{k-1}^*\tilde{\omega}_{k-1}^*(\mathrm{t}) + \tilde{a}_k^*\tilde{\omega}_k^*(\mathrm{t}) \equiv$$
$$\equiv a_{k-4}^*\omega_{k-4}^*(\mathrm{t}) + a_{k-3}^*\omega_{k-3}^*(\mathrm{t}) + a_{k-2}^*\omega_{k-2}^*(\mathrm{t}) + a_{k-1}^*\omega_{k-1}^*(\mathrm{t}) + a_k^*\omega_k^*(\mathrm{t})$$
$$\forall\,\mathrm{t}\in(\alpha,\beta) \quad \forall\,\mathrm{i}\in\mathbb{Z}.$$

а при $j \in J_k$ эти числа определяются применением формул Крамера к системе

$$\tilde{a}_{k-4}^*\tilde{\omega}_{k-4}^*(\mathrm{t}) + \tilde{a}_{k-3}^*\tilde{\omega}_{k-3}^*(\mathrm{t}) + \tilde{a}_{k-2}^*\tilde{\omega}_{k-2}^*(\mathrm{t}) + \tilde{a}_{k-1}^*\tilde{\omega}_{k-1}^*(\mathrm{t}) + \tilde{a}_k^*\tilde{\omega}_k^*(\mathrm{t}) \equiv$$
$$\equiv a_{k-4}^*\omega_{k-4}^*(\mathrm{t}) + a_{k-3}^*\omega_{k-3}^*(\mathrm{t}) + a_{k-2}^*\omega_{k-2}^*(\mathrm{t}) + a_{k-1}^*\omega_{k-1}^*(\mathrm{t}) + a_k^*\omega_k^*(\mathrm{t})$$
$$\forall\,\mathrm{t}\in(\alpha,\beta) \quad \forall\,\mathrm{i}\in\mathbb{Z}.$$

Литература

1. *Ю. К. Демьянович, И. Д. Мирошниченко*. Калибровочные соотношения для Bφ-сплайнов четвертого порядка // Проблемы математического анализа. 2011. Вып. 60. С. 12–24.
2. *Ю. К. Демьянович, И. Д. Мирошниченко*. Гнездовые сплайн-вэйвлетные разложения // Проблемы математического анализа. 2012. Вып. 64. С. 51–61.
3. *И. Д. Мирошниченк*о. Структура сплайн-вэйвлетныхразложений // Материалы всероссийской научной конференции по проблемам информатики «СПИСОК-2013». Спб. С. 230–236.

О ВЭЙВЛЕТНОМ РАЗЛОЖЕНИИ

М. В. Парахин

аспирант 1 курса математико-механического факультета СПбГУ,
Кафедры параллельных алгоритмов

E-mail: Mikhail.Parakhin@gmail.com

Аннотация. Проведен анализ алгоритма обработки числовых информационных потоков с помощью сплайн-вэйвлетных преобразований эрмитова типа третьей высоты, разработана компьютерная реализация алгоритма, проведен анализ возможностей распараллеливания.

Введение

Данная работа посвящена разработке и изучению свойств сплайн-вэйвлетных разложений с высокой точностью приближения гладких цифровых потоков. Они приводят к эффективному сжатию и к достаточно точному результату, так как учитывают гладкость обрабатываемого потока цифровой информации. Для случаев, когда исходный поток интерпретируется как значения гладкой функции на некоторой сетке, разработаны сплайн-вэйвлетные разложения лагранжева типа. В тех случаях, когда исходный поток распадается на два и более потоков — на поток значений функции и на поток значений ее производных в узлах сетки, построены сплайн-вэйвлетные разложения эрмитова типа. Цель данной работы — анализ алгоритма обработки числовых информационных потоков с помощью сплайн-вэйвлетных преобразований эрмитова типа третьей высоты из работы [1], разработка компьютерной реализации алгоритма и анализ возможностей распараллеливания.

1. Минимальные сплайны эрмитова типа

Рассмотрим восьмикомпонентную вектор-функцию $\varphi(t)$ класса $C^3(\alpha, \beta)$, ее компоненты будем обозначать квадратными скобками:

$$\varphi(t) = ([\varphi]_0(t), [\varphi]_1(t), [\varphi]_2(t), \ldots, [\varphi]_7(t))^T.$$

Предположим, что выполнено следующее условие:

$$W_C \stackrel{\text{def}}{=} \det(\varphi'''(x), \varphi''(x), \varphi'(x), \varphi(x), \varphi'''(y), \varphi''(y), \varphi'(y), \varphi(y)) \neq 0 \qquad (C)$$

для всех $x \neq y$; $x, y \in (\alpha, \beta)$.

Пусть X — сетка вида

$$X: \ldots < x_{-1} < x_0 < x_1 < \ldots;$$

$$\alpha \stackrel{\text{def}}{=} \lim_{j \to -\infty} x_j, \quad \beta \stackrel{\text{def}}{=} \lim_{j \to +\infty} x_j.$$

Обозначим

$$G \overset{\text{def}}{=} \bigcup_{j\in\mathbb{Z}} (x_j,\, x_{j+1}),$$

$$\varphi_k \overset{\text{def}}{=} \varphi(x_k), \quad \varphi'_k \overset{\text{def}}{=} \varphi'(x_k), \quad \varphi''_k \overset{\text{def}}{=} \varphi''(x_k), \quad \varphi'''_k \overset{\text{def}}{=} \varphi'''(x_k).$$

Функции

$$\omega_{4j-3}(t), \quad \omega_{4j-2}(t), \quad \omega_{4j-1}(t), \quad \omega_{4j}(t), \quad t \in G, \quad j \in \mathbb{Z},$$

определим из аппроксимационных соотношений

$$\sum_j \varphi'''_{j+1}\omega_{4j-3}(t) + \varphi''_{j+1}\omega_{4j-2}(t) + \varphi'_{j+1}\omega_{4j-1}(t) + \varphi_{j+1}\omega_{4j}(t) = \varphi(t)$$

при условиях

$$\operatorname{supp}\omega_{4j-3} \subset [x_j,\, x_{j+2}], \quad \operatorname{supp}\omega_{4j-2} \subset [x_j,\, x_{j+2}],$$

$$\operatorname{supp}\omega_{4j-1} \subset [x_j,\, x_{j+2}], \quad \operatorname{supp}\omega_{4j} \subset [x_j,\, x_{j+2}].$$

При фиксированных $k \in \mathbb{Z}$ и $t \in (x_k, x_{k+1})$ из (1.1) получаем

$$\varphi'''_k\omega_{4k-7}(t) + \varphi''_k\omega_{4k-6}(t) + \varphi'_k\omega_{4k-5}(t) + \varphi_k\omega_{4k-4}(t) + \varphi'''_{k+1}\omega_{4k-3}(t) +$$

$$+ \varphi''_{k+1}\omega_{4k-2}(t) + \varphi'_{k+1}\omega_{4k-1}(t) + \varphi_{k+1}\omega_{4k}(t) = \varphi(t).$$

Благодаря свойству (C), система (1.2) однозначно разрешима, а для явного представления функций $\omega_{4k-j}(t), j=0, 1, \ldots, 7$, при $t \in (x_k, x_{k+1})$ можно использовать формулы Крамера (для краткости эти представления не выписываем).

Ввиду условия (C), система функций

$$\varphi_0(t),\, \varphi_1(t),\, \varphi_2(t),\, \ldots,\, \varphi_7(t)$$

линейно независима на любом интервале вещественной оси, отсюда следует линейная независимость рассматриваемых сплайнов.

Теорема 1.1. Пусть $\varphi(t) \in C^3(\alpha, \beta)$, и пусть выполнено условие (C). При любом $q \in \mathbb{Z}$ функции $\omega_{4q-3}(t)$, $\omega_{4q-2}(t)$, $\omega_{4q-1}(t)$, и $\omega_{4q}(t)$ могут быть продолжены на весь интервал (α, β) до функций класса $C^3(\alpha, \beta)$. Кроме того,

$$\omega^{(i)}_{4q-s}(x_j) = \delta_{q+1,j}\delta_{s,i}, \tag{1.3}$$

где $i=0, 1, 2, 3$; $j=q, q+1, q+2$; $s=0, 1, 2, 3$; $q \in \mathbb{Z}$.

Доказательство. Вычисляя соответствующие односторонние пределы от функций

$$\omega_{4q-3}(t),\, \omega_{4q-2}(t),\, \omega_{4q-1}(t),\, \omega_{4q}(t)$$

и их первых, вторых и третьих производных в узлах x_q, x_{q+1}, x_{q+2} приходим к утверждению теоремы.

Введем обозначение

$$S_{\varphi}^{3}(X) = \{u \mid u = \sum_{i} c_i \omega_i \quad \forall c_i \in R,\ i \in \mathbb{Z}\}.$$

Пространство $S_{\varphi}^{3}(X)$ называется пространством *сплайнов эрмитова типа третьей высоты*, а множество функций $\omega_j(t)$ — *главным базисом пространства* $S_{\varphi}^{3}(X)$.

2. Калибровочные соотношения

Добавляя ξ к сетке X, $\xi \in (x_k, x_{k+1})$ получим новую сетку $\overline{X}$. Обозначим

$$\overline{x}_j = \begin{cases} x_j, & j \le k, \\ x_{j-1}, & j \ge k+2, \end{cases}$$

$$\overline{x}_{k+1} = \xi.$$

Построим новые базисные функции $\overline{\omega}_j(t)$, $j \in \mathbb{Z}$, аналогичным методом. Положим

$$\overline{\varphi}_j \overset{\text{def}}{=} \varphi(\overline{x}_j), \quad \overline{\varphi}'_j \overset{\text{def}}{=} \varphi'(\overline{x}_j), \quad \overline{\varphi}''_j \overset{\text{def}}{=} \varphi''(\overline{x}_j), \quad \overline{\varphi}'''_j \overset{\text{def}}{=} \varphi'''(\overline{x}_j),$$

Формулы для функций $\overline{\omega}_j(t)$ получаются из формул для $\omega_j(t)$ заменой φ_q, φ'_q, φ''_q, φ'''_q, x_q, x_{q+1}, x_{q+2} на $\overline{\varphi}_q$, $\overline{\varphi}'_q$, $\overline{\varphi}''_q$, $\overline{\varphi}'''_q$, $\overline{x}_q$, $\overline{x}_{q+1}$, $\overline{x}_{q+2}$ соответственно. Как и выше, справедливы равенства

$$\overline{\omega}_{4q-s}^{(i)}(\overline{x}_j) = \delta_{q+1,j}\delta_{s,i}, \tag{2.1}$$

где $i = 0, 1, 2, 3$; $j = q, q+1, q+2$; $s = 0, 1, 2, 3$; $q \in \mathbb{Z}$. Нетрудно видеть, что при $j \le 4k-8$ верно равенство

$$\omega_j(t) = \overline{\omega}_j(t),$$

а при $j \ge 4k+1$ имеем

$$\omega_j(t) = \overline{\omega}_{j+4}(t).$$

Установим формулы для представления функций ω_{4k-7}, ω_{4k-6}, ω_{4k-5}, ω_{4k-4}, ω_{4k-3}, ω_{4k-2}, ω_{4k-1}, ω_{4k} через функции $\overline{\omega}_j(t)$ и проведя некоторые преобразования докажем следующее утверждение.

Теорема 2.1. Если выполнено условие (C) , то при $t \in (\alpha, \beta)$

$$\omega_i(t) = \sum_{j \in Z} \mathfrak{p}_{ij} \overline{\omega}_j(t), \tag{2.2}$$

где

$$\mathfrak{p}_{ij} = \delta_{ij}, \quad j \le 4k-4, \tag{2.3}$$

$$\mathfrak{p}_{ij} = \delta_{i,j-4}, \quad j \geq 4k+1, \tag{2.4}$$

$$\mathfrak{p}_{i,4k-s} = 0, \quad s = 0,1,2,3 \quad (i \leq 4k-8) \vee (i \geq 4k+1), \tag{2.5}$$

$$\mathfrak{p}_{i,4k-s} = \omega_i^{(s)}(\xi), \quad s = 0,\ 1,\ 2,\ 3,\ \ 4k-7 \leq i \leq 4k. \tag{2.6}$$

Замечание. Введем обозначение

$$S_\varphi^3(\overline{X}) = \{v \mid v = \sum_i d_i \overline{\omega}_i \ \ \forall d_i \in \mathbb{R},\ i \in \mathbb{Z}\}.$$

Из предыдущего видно, что $S_\varphi^3(\overline{X}) \supset S_\varphi^3(X)$.

3. Биортогональная система функционалов и их значения на функциях $\overline{\omega}_j(t)$

Над пространством $C^3(\alpha, \beta)$ рассмотрим систему линейных функционалов $\{g^{(i)}\}_{i \in Z}$, определяемых соотношениями

$$\langle g^{(4q-s)}, u \rangle \stackrel{\text{def}}{=} u^{(s)}(x_{q+1}), \quad s = 0,\ 1,\ 2,\ 3. \tag{3.1}$$

Имеем

$$\langle g^{(i)}, \omega_j \rangle = \delta_{i,j}.$$

Введем матрицу

$$\mathfrak{Q} \stackrel{\text{def}}{=} (\mathfrak{q}_{ij})_{i,j \in \mathbb{Z}},$$

где

$$\mathfrak{q}_{ij} \stackrel{\text{def}}{=} \langle\ (\ ), \overline{\omega}_j \rangle. \tag{3.2}$$

Справедливы следующие соотношения

$$\mathfrak{q}_{ij} = \begin{cases} \delta_{ij}, & j \leq 4k-4, \\ \delta_{i,j-4}, & j \geq 4k+1, \end{cases} \tag{3.3}$$

$$\mathfrak{q}_{i,4k-3} = \mathfrak{q}_{i,4k-2} = \mathfrak{q}_{i,4k-1} = \mathfrak{q}_{i,4k} = 0 \quad \forall i \in \mathbb{Z}. \tag{3.4}$$

Обозначим $\mathfrak{P} \stackrel{\text{def}}{=} (\mathfrak{p}_{ij})_{i,j \in \mathbb{Z}}$. Покажем, что матрица $\mathfrak{Q}$ является левой обратной к $\mathfrak{P}^T$. Действительно, из (2.2) следует

$$\omega(t) = \mathfrak{P}\overline{\omega}(t),$$

где

$$\omega(t) \stackrel{\text{def}}{=} (\ldots, \omega_{-1}(t), \omega_0(t), \omega_1(t), \ldots)^T$$

и

$$\overline{\omega}(t) \stackrel{\text{def}}{=} (\ldots, \overline{\omega}_{-1}(t), \overline{\omega}_0(t), \overline{\omega}_1(t), \ldots)^T.$$

Транспонированием соотношения получаем равенство вектор-строк

$$(\omega)^T(t) = (\overline{\omega})^T(t)\mathfrak{P}^T.$$

Умножение равенство на вектор-столбец $g \overset{\text{def}}{=} (g^{(i)})_{i\in\mathbb{Z}}$, получим $I = \mathfrak{Q}\mathfrak{P}^T$.

4. Вэйвлетное разложение. Формулы декомпозиции и реконструкции

Вэйвлетное разложение пространства $S^3_\varphi(\overline{X})$ определим равенством

$$S^3_\varphi(\overline{X}) = PS^3_\varphi(\overline{X}) \dotplus W,$$

где P — оператор проектирования $S^3_\varphi(\overline{X})$ на подпространство $S^3_\varphi(X)$ — задается формулой

$$P\tilde{u} \overset{\text{def}}{=} \sum_i a_i \omega_i$$

при $a_i = \langle g^{(i)}, \tilde{u}\rangle$ для всех $\tilde{u} \in S^3_\varphi(\overline{X})$,

$$W \overset{\text{def}}{=} QS^3_\varphi(\overline{X}), \quad Q = I - P,$$

I — тождественный оператор. Пространство W называется *пространством вэйвлетов*.

Пусть $\tilde{u} \in S^3_\varphi(\overline{X})$. Тогда

$$\tilde{u} = \sum_k c_k \overline{\omega}_k = \sum_i a_i \omega_i + \sum_j b_j \overline{\omega}_j = \sum_j \left(\sum_i a_i p_{ij} + b_j \right) \overline{\omega}_j.$$

Поэтому

$$c_k = \sum_i a_i p_{ik} + b_k.$$

Коэффициенты b_j выражаются через коэффициенты c_k следующим образом:

$$b_j = c_j - \sum_i p_{ij} \langle g^{(i)}, \sum_k c_k \overline{\omega}_k \rangle = c_j - \sum_i p_{ij} \sum_k c_k \langle g^{(i)}, \overline{\omega}_k \rangle.$$

Итак,

$$a_i = \sum_{i'} \mathfrak{q}_{ii'} c_{i'}, \tag{4.1}$$

$$b_j = c_j - \sum_{i'} \left(\sum_i \mathfrak{p}_{ij} \mathfrak{q}_{ii'} \right) c_{i'}. \tag{4.2}$$

Формулы декомпозиции. Для вэйвлетного разложения справедливы следующие соотношения:

$$a_i = \begin{cases} c_i, & i \le 4k-4, \\ c_{i+4}, & i \ge 4k-3, \end{cases} \qquad b_j = 0, \quad (i \le 4k-4) \vee (j \ge 4k+1), \tag{4.3}$$

$$b_{4k-i} = c_{4k-i} - c_{4k-7} w^{(i)}_{4k-7}(\xi) - c_{4k-6} w^{(i)}_{4k-6}(\xi) - c_{4k-5} w^{(i)}_{4k-5}(\xi)$$
$$- c_{4k-4} w^{(i)}_{4k-4}(\xi) - c_{4k+1} w^{(i)}_{4k-3}(\xi) - c_{4k+2} w^{(i)}_{4k-2}(\xi)$$
$$- c_{4k+3} w^{(i)}_{4k-1}(\xi) - c_{4k+4} w^{(i)}_{4k}(\xi), \quad i = 0, 1, 2, 3.$$

Пространство W четырехмерно, и его базисом служат функции $\overline{\omega}_{4k-3}(t)$, $\overline{\omega}_{4k-2}(t)$, $\overline{\omega}_{4k-1}(t)$ и $\overline{\omega}_{4k}(t)$.

Формулы реконструкции. Пусть известны коэффициенты a_i, b_{4k-3}, b_{4k-2}, b_{4k-1}, b_{4k} в разложениях проекций элемента $\tilde{u} \in S^3_\varphi(\overline{X})$ на пространство $S^3_\varphi(X)$ и W,

$$P\tilde{u} = \sum_i a_i \omega_i,$$

$$Q\tilde{u} = b_{4k-3}\overline{\omega}_{4k-3} + b_{4k-2}\overline{\omega}_{4k-2} + b_{4k-1}\overline{\omega}_{4k-1} + b_{4k}\overline{\omega}_{4k}.$$

Тогда коэффициенты c_k в формуле

$$\tilde{u} = \sum_k c_k \overline{\omega}_k$$

имеют вид

$$_i = \begin{cases} a_i, & i \le 4k-4, \\ a_{i-4}, & i \ge 4k+1, \end{cases}$$

$$c_{4k-i} = b_{4k-i} + a_{4k-7} w^{(i)}_{4k-7}(\xi) + a_{4k-6} w^{(i)}_{4k-6}(\xi) + a_{4k-5} w^{(i)}_{4k-5}(\xi)$$
$$+ a_{4k-4} w^{(i)}_{4k-4}(\xi) + a_{4k-3} w^{(i)}_{4k-3}(\xi) + a_{4k-2} w^{(i)}_{4k-2}(\xi)$$
$$+ a_{4k-1} w^{(i)}_{4k-1}(\xi) + a_{4k} w^{(i)}_{4k}(\xi), \quad i = 0, 1, 2, 3.$$

5. Компьютерная реализация алгоритма и анализ возможностей распараллеливания

Дан исходный числовой поток, который представляет из себя векторы значений функции φ, а также ее первой, второй и третьей производных в точках сетки X. По формулам декомпозиции на основе значений исходного потока вычисляются значения основного и вэйвлетного потока. По формулам реконструкции производится восстановление значение исходного потока на основе значений основного и вэйвлетного потоков и сравнение восстановленного потока с исходным для оценки погрешности.

Компьютерная реализация сплайн-вэйвлетного разложения базируется на формулах декомпозиции и реконструкции и вычислительный процесс по этим формулам может происходит независимо. Значения основного и вэйвлетного потоков можно вычислять независимо, для этого используются только значения исходного потока. Таким образом можно разбить исходный поток на блоки и передавать блоки для обработки различным параллельным вычислительным модулям. В случае, если накладные расходы на распараллеливание достаточно малы, ускорение линейно.

Заключение

Проведен анализ алгоритма обработки числовых информационных потоков с помощью сплайн-вэйвлетных преобразований эрмитова типа третьей высоты, разработана компьютерная реализация алгоритма, проведен анализ возможностей распараллеливания.

Литература

1. *Ле Т. Н. Б., Демьянович Ю. К.* Вэйвлетное разложение сплайнов эрмитова типа третьей высоты. Проблемы математического анализа. Вып. 44. 2010.
2. *Демьянович Ю. К., Зимин А. В.* О всплесковом разложении сплайнов эрмитова типа. Проблемы математического анализа. Вып. 35. 2007.
3. *Демьянович Ю. К.* Всплески и минимальные сплайны. СПб., 2003.

Теория и практика кодирования информации

Крук
Евгений Аврамович
д.т.н., профессор
декан факультета информационных систем и защиты информации
заведующий кафедрой
безопасности информационных систем СПбГУАП

Абрамов
Андрей Юрьевич
научный сотрудник кафедры безопасности
информационных систем ГУАП

АНАЛИЗ ПРИМЕНЕНИЯ РАЗРЕЖЕННОГО КОДИРОВАНИЯ В ЗАДАЧЕ ВОССТАНОВЛЕНИЯ РЕГИОНОВ ИЗОБРАЖЕНИЙ

В. А. Ястребов

асп. кафедры инфокоммуникационных систем

E-mail: victor.yastrebov1@yandex.ru

А. И. Веселов

асс. кафедры инфокоммуникационных систем

E-mail: felix@vu.spb.ru

Аннотация. В статье рассматривается задача маскирования визуальных искажений, возникающих в процессе передачи данных по сети. Анализируется эффективность методов, основанных на использовании разреженного кодирования для борьбы с такими искажениями. Приводятся результаты работы рассматриваемых алгоритмов на предложенной модели ошибок в канале.

Введение

В зависимости от используемого метода передачи, ошибки, возникающие при передаче данных по сети, могут варьироваться от случайных битовых ошибок в потоке данных вплоть до потерь отдельных сетевых пакетов. Наиболее широко распространенным способом борьбы с ошибками является применение помехоустойчивого кодирования. В случае если возникшие ошибки не удается полностью исправить, то используется ретрансляция данных. При этом необходимо, чтобы существовала обратная связь между передатчиком (кодером) и получателем (декодером) данных.

Однако в некоторых случаях обратной связи не существует, или по ряду причин воспользоваться ей не представляется возможным. Например, при обработке видеоданных в режиме реального времени, ретрансляция потерянных данных не является приемлемым решением, поскольку она может привести к недопустимым задержкам во времени.

В связи с этим, актуальной задачей является обработка ошибок в канале передачи данных только на стороне декодера. В таких системах восстановление потерянных регионов осуществляется на основе анализа успешно декодированных областей изображения.

Существует несколько основных направлений к восстановлению потерянных блоков данных изображения, например: методы, основанные на решении выпуклых задач [1], методы, анализирующие геометрические свойства фигур [2], спектральные методы [3] и т. д. В настоящее время активно развиваются алгоритмы, основанные на разреженном представлении данных.

В данной статье рассматривается их применение в задаче восстановления регионов изображения.

Описание рассматриваемой модели обработки и передачи изображений по сети

Введем в рассмотрение модель системы обработки и передачи изображений по сети, построенную на основе следующих допущений. Кодер (передатчик) считается зафиксированным и никакие изменения в него вносить нельзя. Между кодером и декодером налажена пакетная передача данных, при этом обратная связь между ними отсутствует. Таким образом, декодер не может запросить у кодера повторной отправки каких-либо пакетов.

Рис. 1. Пример изображения с искаженными макроблоками

Также вводится модель ошибок, основанная на следующих допущениях:

- любые ошибки в пакете будут обнаружены;
- наличие ошибки в пакете приводит к его полной потере;

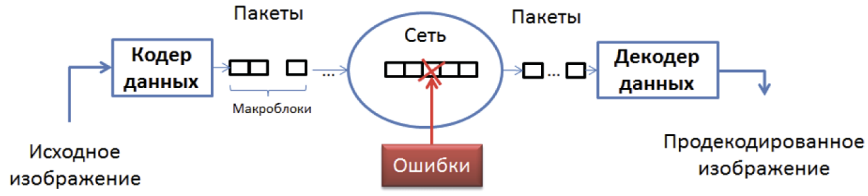
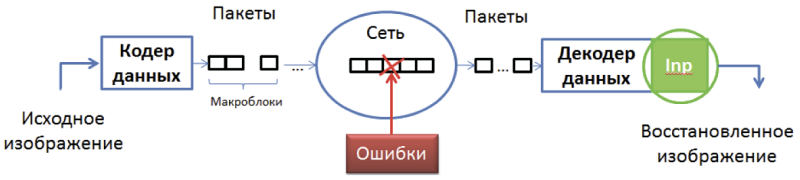


Рис. 2. A — модель кодера и модель ошибок; B — рассматриваемая модификация декодера

A



B

- потеря пакета приводит к искажению региона (макроблока) изображения;
- макроблоки могут иметь различный размер;
- количество искаженных макроблоков может варьироваться.

Пример изображения с искаженными макроблоками приведен на рис. 1. Белым цветом выделены регионы, требующие восстановления.

Рассматриваемая модель кодера и декодера данных приведена на рис. 2, *A*. Анализируемый модуль на стороне декодера выделен на рис. 2, *B*.

Понятие разреженного представления данных

Анализируемый подход к восстановлению регионов изображения основан на алгоритме шумоподавления, использующем разреженное представление (кодирование) данных [4].

Задачей разреженного кодирования является представление исходного сигнала вектором, содержащим малое количество ненулевых элементов. Компоненты вектора являются координатами сигнала в пространстве, определяемом словами (*атомами*) из переполненного словаря, включающего в себя набор базисных векторов, а также набор дополнительных векторов, являющихся линейными комбинациями базисных.

Формальная постановка задачи разреженного представления данных приведена на рис. 3, где n — размерность исходного сигнала; k — размер словаря; L — максимально допустимое количество ненулевых элементов в $\bar{\alpha}$; $\| \bullet \|_0 - l_0$ — норма вектора (количество ненулевых элементов).

Для представления изображения в разреженном виде оно разбивается на пересекающиеся блоки (*патчи*). Каждый блок раскладывается по словарю так, что в разложении используется небольшое число атомов, линейная комбинация которых хорошо его аппроксимирует. Пиксели искаженного макроблока могут входить в состав нескольких патчей. В результате восстановления каждого патча по словарю одному и тому же искаженному пикселю может быть поставлено в соответствие несколько аппроксимирующих значений, взвешенная сумма которых определяет итоговое значение пикселя.

$\bar{x} \in \Re^{n\times 1}$ - исходный сигнал $\quad \bar{\alpha} \in \Re^{k\times 1}$ - его представление

$D \in \Re^{n\times k}$ - словарь $\quad L$ - пороговое значение

$$\bar{x} = D\bar{\alpha}$$

$$\|\bar{\alpha}\|_0 < L \ll n$$

Рис. 3. Формальная постановка задачи разреженного кодирования

Алгоритмы, работающие с фиксированным словарем, принято называть *неадаптивными*. Алгоритмы, изменяющие в процессе работы исходный словарь, называют *адаптивными*.

Адаптивные алгоритмы восстановления изображений являются итеративными и основаны на использовании схемы KSVD [5]. Согласно этой схеме, на каждой итерации происходит последовательное изменение атомов текущего словаря так, чтобы ошибка аппроксимации патчей уменьшалась. Изменение атома осуществляется за счет использования SVD разложения.

Результаты проведенных экспериментов

В соответствии с приведенной выше моделью передачи данных и моделью ошибок в канале, было выполнено имитационное моделирование с целью оценки эффективности восстановления изображений, взятых из стандартного тестового множества [6].

Имитационное моделирование включало в себя ряд экспериментов, каждый из которых состоял из пяти тестов. Для заданного в тесте изображения фиксировался размер и количество искаженных макроблоков. Координаты искаженных макроблоков выбирались случайным образом. Результатом эксперимента являлось среднее значение PSNR восстановленных изображений.

В данной работе были рассмотрены два типа исходных словарей: словарь, состоящий из базисных функций дискретно-косинусного преобразо-

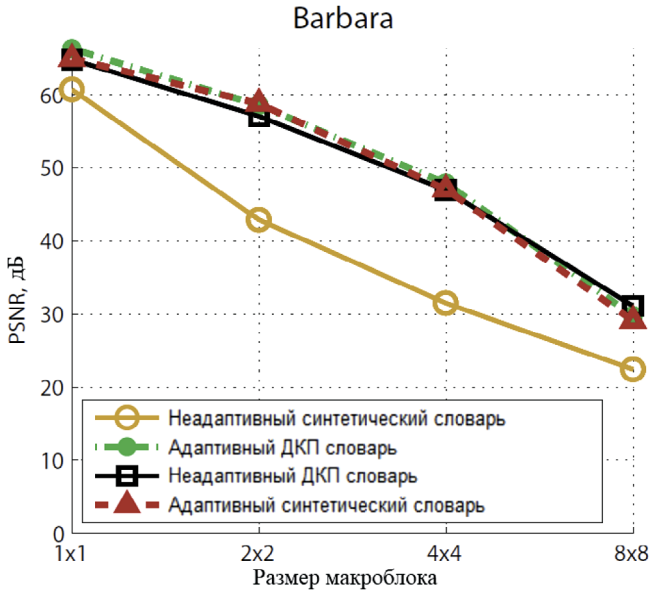


Рис. 4, *А*. Результаты обработки изображения barbara

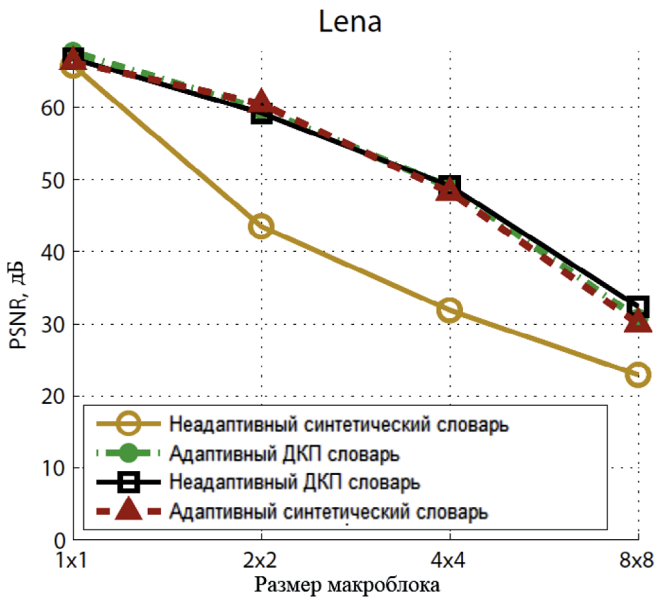


Рис. 4, *B*. Результаты обработки изображения lena

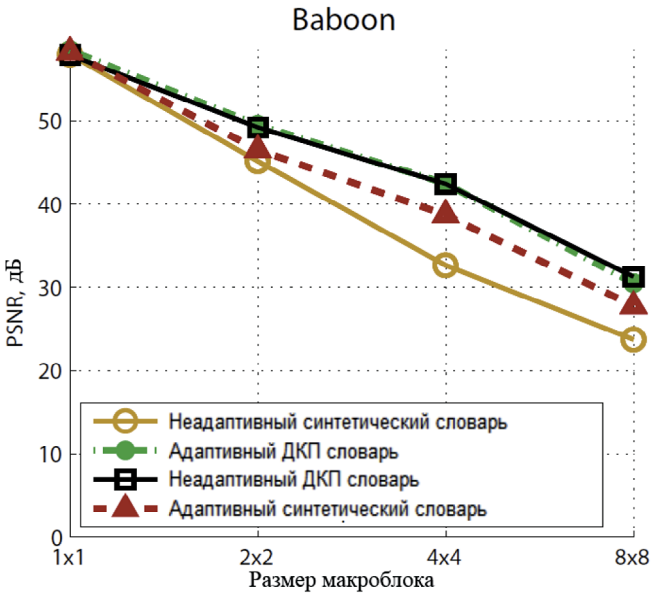


Рис. 4, *C*. Результаты обработки изображения baboon

вания (ДКП), а также синтетический словарь, представленный в работе [4]. Атомы синтетического словаря были получены в процессе обучения на большой выборке неискаженных изображений. В каждом эксперименте размер патча был зафиксирован и составлял 8×8 пикселей. В каждом словаре содержалось 256 атомов.

Параметры экспериментов:

- тестовые изображения: barbara, lena, baboon;
- количество искаженных макроблоков: 16, 32, 64, 128;
- размер макроблоков: 1×1, 2×2, 4×4, 8×8.

В результате проведенных экспериментов было установлено, что количество искаженных макроблоков не влияет на соотношение результатов экспериментов. В связи с этим на рис. 4 приведены результаты сравнения для тестовых изображений с максимальным рассматриваемым количеством искаженных макроблоков.

Из полученных результатов следует, что методы восстановления регионов изображений на основе применения адаптивного словаря показывают лучшее качество восстановления по сравнению с методами, использующими фиксированный словарь.

Заключение

Проведенный анализ показывает, что адаптивная схема восстановления, основанная на использовании словаря, состоящего из набора базисных векторов ДКП, позволяет восстанавливать искаженные регионы на изображении более точно, однако полученный выигрыш не является существенным. Вопросы, связанные с генерацией универсального синтетического словаря, требуют дополнительного анализа.

Литература

1. *H. Sun* and *W. Kwok*, «Concealment of damaged block transform coded images using projections onto convex sets,» IEEE Trans. Image Processing, vol. 4, pp. 470–477, Apr. 1995.
2. *W. Zeng* and *B. Liu*, «Geometric-structure-based error concealment with novel applications in block-based Low Bit Rate Coding,» IEEE Transactions on Circuits and Systems for Video Technology, pp. 648–665, June 1999.
3. *Z. Alkachouh* and *M. Bellanger*, «Fast DCT-based spatial domain interpolation of blocks in images,» IEEE Trans. Image Processing, vol. 9, pp.729–732, Apr. 2000.
4. *M. Elad* and *M. Aharon*, «Image denoising via learned dictionaries and sparse representation,» IEEE ComputerVision and Pattern Recognition, New York, Jun. 2006.
5. *M. Aharon, M. Elad, A. Bruckstein*, «The K-SVD: An algorithm for designing of overcomplete dictionaries for sparse representation,» IEEE, Trans. Signal processing, vol. 54, no. 11, Nov. 2006, pp. 4311–4322.
6. SIPI image database http://sipi.usc.edu/database/database.php?volume=misc&image=11 [дата просмотра: 21.04.2014].

О ЗАДАЧЕ УНИВЕРСАЛЬНОГО КОДИРОВАНИЯ ИСТОЧНИКОВ БЕЗ ПАМЯТИ

Н. Д. Егоров
аспирант кафедры инфокоммуникационных систем СПбГУАП
E-mail: negorov.91@gmail.com

М. Р. Гилмутдинов
к.т.н., доцент кафедры инфокоммуникационных систем СПбГУАП
E-mail: mgilm@vu.spb.ru

Аннотация. В данной работе рассматривается задача универсального кодирования источников без памяти. Производится обзор существующих методов универсального кодирования. Предлагаются способы оценки их эффективности. Производится сравнение рассмотренных методов с помощью предложенных способов оценки их эффективности.

Введение

В данный момент в связи с увеличением объемов передаваемой и используемой информации растет потребность в ее компактном представлении. В связи с этим кодирование реальных данных является актуальной задачей. Реальные источники данных в большинстве случаев можно рассматривать как источники с памятью и с неизвестной статистикой. Типовой алгоритм кодирования таких источников представлен на Рис. 1:

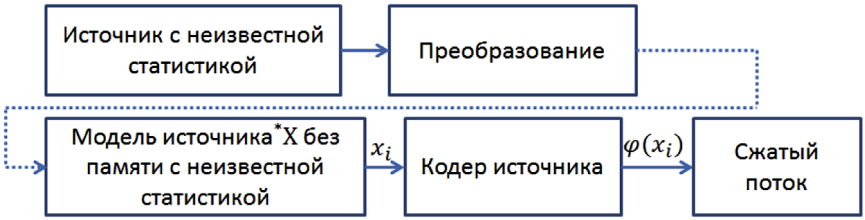


Рис. 1. Типовая схема кодирования реальных данных

где x_i — сообщение источника X.

Учет памяти источника выделяют как отдельную задачу, решаемую с помощью обратимых декоррелирующих преобразований (например, для изображений применяется подход, описанный в [1]) и контекстного моделирования [2]. Если сделать допущение, что выход блока «Преобразование» является источником без памяти, то можно совершить переход к задаче универсального кодирования источника без памяти. При этом решаемую задачу принято разделять на две подзадачи:

1. оценка вероятностных характеристик источника;
2. построение оптимального кода для источника с известной статистикой.

Задача о построении оптимальных кодов для источника с известной статистикой имеет множество теоретических подходов [3–5] для которых доказана оптимальность. В связи с этим основное внимание в статье будет уделено оценке вероятностных характеристик источника.

Решение описанной задачи зависит от стационарности источника. Источник считается стационарным, если его вероятностные характеристики не зависят от конкретного момента времени, и выполняется следующее условие:

$$p^{(t)}(x_i) = \text{const}, \quad (1)$$

где $p^{(t)}(x_i)$ — вероятность для символа x_i в момент времени t. Если условие (1) не выполняется, то источник считается нестационарным. Именно к таким источникам в большинстве случаев относят реальные данные. Из множества нестационарных источников выделяют кусочно-стационарные [6] — источники, чьи вероятностные характеристики являются локально стационарными. То есть последовательность данных, полученную от такого источника, можно представить в виде набора подпоследовательностей, для которых выполняется условие (1).

В данной статье производится обзор существующих методов оценки вероятностных характеристик источников без памяти. Предлагаются способы оценки эффективности для рассмотренных методов. Выполняется сравнительный анализ рассмотренных методов с помощью предложенных методик оценки их эффективности.

Обзор существующих методов оценки вероятностных характеристик источника

Различают два подхода [4], [5] к оценке параметров неизвестного источника: *двухпроходное* кодирование или кодирование *с задержкой* и *однопроходное* кодирование. При работе с реальными данными в большинстве случаев наличие задержки нежелательно, поэтому в работе будут рассматриваться подходы, применяемые для однопроходного кодирования.

Типовым методом оценки вероятности для стационарного источника является использование счетчиков для символов алфавита и их последовательное обновление после обработки очередного символа. При этом данный подход не предполагает адаптацию к изменениям вероятностных характеристик источника. Легко показать, что использование данного метода при кодировании нестационарных источников является неэффективным.

Для нестационарного источника при кодировании необходимо учитывать изменение его характеристик с течением времени. В работе рассматриваются следующие методы оценки вероятностных характеристик *двоичного* источ-

ника (на практике принято представлять недвоичный источник в виде набора двоичных источников с помощью процедуры *бинаризации*).

- *Счетчик с насыщением* — Saturated Counter (SC) [7]. Данный метод может быть представлен в виде конечного автомата с N состояниями. При нахождении в *i*-ом состоянии и появлении символа 1/0 осуществляется переход в $(i+1)/(i-1)$ состояние. Оценка вероятности для *i*-ого состояния осуществляется следующим образом:

$$\begin{cases} \Pr\{1\} = 0 \ \text{ если } \ i < N/2; \\ \Pr\{1\} = 1 \ \text{ если } \ i \geq N/2. \end{cases} \tag{3}$$

- *Счетчик с линейным выходом* — Linear Output Saturated Counter (LOSC) [8]. Отличие данного метода от SC состоит в альтернативной процедуре оценки вероятности символа:

$$\Pr\{1\} = i/N. \tag{4}$$

- *Счетчик со сбросом* — Reset Counter (RC) [9]. Данный метод предполагает наличие графа сложной структуры, где каждой вершине поставлена в соответствие свой набор вероятностей. На каждом шаге происходит переход в новую вершину, но через каждые R переходов происходит возвращение в начальное состояние.
- *Мнимое скользящее окно* — Imaginary Sliding Window (ISW) [10]. Оригинальный метод скользящего окна предполагает оценку вероятности символа исходя из статистики появлений символов источника в буфере, содержащим *N* последних обработанных символов. Мнимое скользящее окно предлагает имитацию данного буфер с помощью счетчика единиц n_1. На каждом шаге предполагается выполнение двух операций:

$$\begin{aligned} &1.\ n_1 = n_1 - 1 \text{ с вероятностью } \Pr\{1\}; \\ &2.\ n_1 = n_1 + 1 \text{ если новый символ "1".} \end{aligned} \tag{4}$$

- *Экспоненциальное забывание* — Exponential Decaying Machine (EDM) [11]. Идея данного метода заключается в ослаблении влияния ранее обработанных символов и может быть представлена в виде следующей формулы:

$$\hat{p}(1)_{t+1} = (1-\lambda)\,\hat{p}(1)_t + \lambda x_t, \tag{5}$$

где λ — коэффициент, управляющий скоростью забывания; $\hat{p}(1)_t$ — оценка вероятности символа «1» в момент времени *t*.

Также существуют эффективные методы оценок вероятностных характеристик, применяемые для кусочно-стационарных источников [6], [12]. Данные методы обладают сложностью, возрастающей с увеличением количества закодированных символов. Поэтому они считаются неприменимыми на практике и в работе не рассматриваются.

Способы оценки эффективности методов универсального кодирования

Для сравнения эффективности вышеперечисленных методов оценки вероятностных характеристик было предложено их применение к нестационарным источникам. Битовый поток, используемый для сравнения эффективности, формируется с помощью арифметического кодера [13]. Для моделирования выхода нестационарного источника используются:

— модели кусочно-стационарных источников;
— последовательности реальных данных.

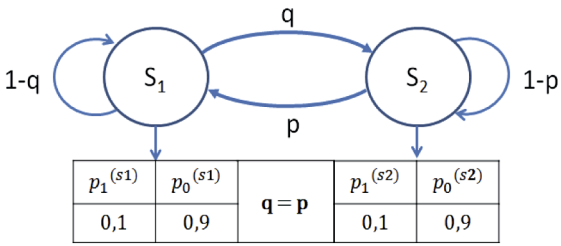
Для моделирования кусочно-стационарных источников использовалась модель Гилберта—Элиота, как показано на Рис. 2. В общем случае на базе данной модели формируется стационарный процесс. Но на выборках ограниченной длины и с малой заданной вероятностью перехода между состояниями модели, возможно получение последовательностей, близких по свойствам к выходу кусочно-стационарного источника.

В качестве последовательностей реальных данных возможно использование изображений. Но, так как в большинстве случаев изображение является источником без памяти, существует необходимость использования декоррелирующих преобразований для устранения зависимостей в обрабатываемых данных.

Результаты сравнительного анализа

При использовании универсального кодирования на практике возникает необходимость ограничения выделяемой под кодирование памяти. Данное ограничение для метода оценки вероятностных характеристик будет представлено параметром K — количеством состояний конечного автомата, реализующего рассматриваемый метод.

Проводилось сравнение эффективности описанных ранее методов при различных значениях «K». Для имитации выхода нестационарного источника использовалась Y-компонента изображения IMMGE_15.ppm из мно-



$p_1(s1)$	$p_0(s1)$	q = p	$p_1(s2)$	$p_0(s2)$
0,1	0,9		0,1	0,9

Рис. 2. Имитация выхода нестационарного источника с помощью модели Гилберта—Элиота

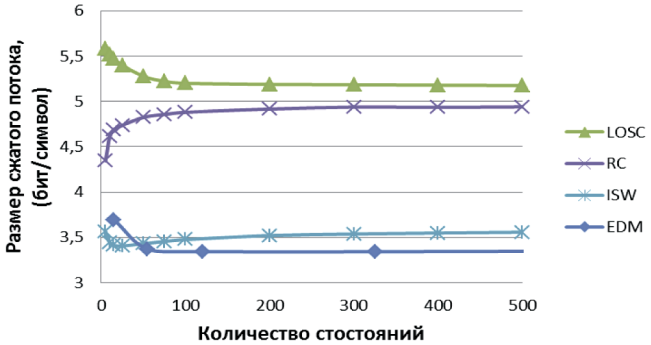


Рис. 3. Зависимость коэффициента сжатия от количества состояний для известных методов кодирования (для изображения)

жества Kodak [14]. В качестве декоррелирующего преобразования использовалось MED предсказание из стандарта сжатия JPEG-LS [15]. Рассмотренные ранее методы применялись для кодирования абсолютных значений получаемых ошибок предсказания. Результат проведенного сравнения представлен на Рис. 3. Из него видно, что для заданного нестационарного источника наиболее эффективными оказались ISW и EDM методы. Также видно, что эффективность данных методов зависит от параметра «K». И если для стационарного источника увеличение «K» повышает эффективность сжатия, то для нестационарного источника нахождение оптимального «K» является отдельной актуальной задачей.

В данной работе также исследовалось влияние частоты изменения характеристик источника на эффективность рассмотренных методов. Для имитации выхода нестациоарного источника использовалась модель Гилберта—Элиота с двумя состояниями, представленная на Рис. 2. В каждом

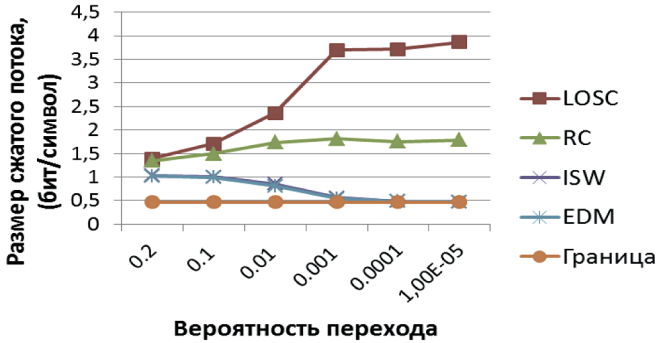


Рис. 4. Зависимость коэффициента сжатия от вероятности перехода состояния модели при $K = 500$ (для модели на базе Гилберта—Элиота)

из состояний производилась генерация выходной двоичной последовательности по заданному для этого состояния распределению. Для проводимого исследования были заданы фиксированное количество переходов $q = 1000$, вероятность перехода $p_{tr} \in [0.2; 10^{-5}]$ и параметр $K = 500$. Выбор данного значения объясняется проведенными практическими экспериментами, из которых следует, что при $K = 500$ рассматриваемые методы в среднем наиболее эффективны. Длина получаемой выборки вычислялась как $N = q/p_{tr}$.

Как видно из представленных на Рис. 4 результатов, при изменении частоты переходов эффективность представленных методов также меняется. В качестве отмеченной на графике нижней границы кодирования берется значение энтропии данного источника при его известных параметрах: $1/2H(S_1) + 1/2H(S_2)$. Результаты лидеров ISW и EDM сходятся к нижней границе кодирования при уменьшении частоты переходов.

Заключение

В данной работе была предложена методика оценки эффективности известных методов универсального кодирования для нестационарных источников. Сравнение, проведенное согласно предложенной методике, показало, что наиболее эффективными методами для кодирования нестационарных источников являются ISW [10] и EDM [11]. Также была обнаружена зависимость эффективности метода от количества используемой памяти, которое рассматривалось в качестве его управляющего параметра. Исходя из этого, был сделан вывод о том, что для конкретного нестационарного источника необходим поиск оптимального значения данного параметра. Поиск решения данной задачи является направлением будущих исследований.

Литература

1. *H. S. Malvar, G. J. Sullivan, S. Srinivasan*. «Lifting-based reversible color transformations for image compression» // SPIE Proceedings, Vol. 7073, № 707307.
2. *Д. Ватолин, А. Ратушняк, М. Смирнов, В. Юкин*. «Методы сжатия данных. Устройство архиваторов, сжатие изображений и видео» // ДИАЛОГ-МИФИ, 2002.
3. *Р. Галлагер*. «Теория информации и надежная связь» // М., Советское радио, 1974.
4. *Б. Д. Кудряшов*. «Теория информации» // Питер, 2009.
5. *Ю. М. Штарьков*. «Универсальное кодирование: теория и алгоритмы» // Москва Физматлит, 2013
6. *N. Merhav*. «On the Minimum Description Length Principle for Source with Piecewise Constant Parameters» // IEEE Trans. On Information Theory, Vol. 39, № 6, November 1993.
7. *E. Federovski*. «Branch prediction based on universal data compression algorithms» // Master»s thesis, Dept. Elec. Eng. — Syst., Tel-Aviv Univ., Tel-Aviv, Israel, 1998.
8. *E. Meron, M. Feder*. «Finite-Memory Universal Prediction of Individual Sequences» // IEEE Trans. On Information Theory, Vol. 50, No. 7, 2004.

9. *K. D. Rajwan*. «Universal Finite Memory Coding of Binary Sequences» // Master»s thesis, Dept. Elec. Eng. — Syst., Tel-Aviv Univ., Tel-Aviv, Israel, 2000.
10. *Б. Я. Рябко*. «Сжатие данных с помощью «мнимого скользящего окна»” // Проблемы передачи информации, том 32, вып. 2, 1996.
11. *E. Meron*. «Universal finite memory prediction, coding and estimation of individual sequences,» // Master»s thesis, Tel-Aviv University, 2003.
12. *I. Shamir, N. Merhav*. «Low Complexity Sequential Lossless Coding for Piecewise Stationary Memoryless Sources» // IEEE Trans. On Information Theory, Vol. 45, № 5, July 1999.
13. *Ian H. Witten, Radford M*. «Arithmetic Coding for Data Compression» // Communications of the ACM, 1987.
14. Тестовое множество Kodak // http://r0k.us/graphics/kodak/
15. Recommendation T.87: Information technology — lossless and near-lossless compression of continuous-tone still images — baseline // ISO/IEC 14495–1:1999.

МЕТОД РЕГУЛЯРИЗАЦИИ В ЗАДАЧЕ ВОССТАНОВЛЕНИЯ ИСКАЖЕННЫХ ИЗОБРАЖЕНИЙ

Д. В. Новиков

магистрант кафедры инфокоммуникационных систем, СПБГУАП

E-mail: dn@vu.spb.ru

Аннотация. В данной работе рассматривается задача разработки метода регуляризации для улучшения качества восстановления размытых и смазанных изображений. Производится обзор существующих методов регуляризации, выделяются их недостатки. Демонстрируется эффективность предложенного метода.

Введение

Ввиду несовершенства регистрирующих устройств, сформированное изображение представляет собой искаженную (нечеткую) копию оригинала. Основными причинами искажений, понижающих резкость изображения, являются: ограниченная разрешающая способность, неправильно выставленный фокус камеры, наличие искажающей среды (например, атмосферы), взаимное движение камеры и объекта относительно друг друга во время экспозиции и др. Устранение или ослабление искажений с целью повышения резкости относится к задаче восстановления изображений.

В процессе восстановления изображений с помощью итерационных методов зачастую возникает задача анализа качества изображения, полученного на каждой итерации алгоритма. Возникает необходимость оценить, достиг ли алгоритм желаемого результата, или необходимо продолжить процедуру восстановления. Такая оценка может выполняться с помощью регуляризационных компонент.

Модель процесса искажения изображения

В данном разделе описывается модель процесса искажения изображения, с которой работает предложенный метод регуляризации. Ввиду того, что изображения представлены в цифровой форме, все функции являются двумерными дискретными массивами отсчетов. Устройства регистрации можно описать в виде линейной системы с пространственно-инвариантными искажениями, таким образом, механизм искажения одинаков во всех точках (y, x). За y и x обозначены координаты пикселя в изображении по высоте и ширине соответственно. Условная запись линейной модели процесса формирования искаженного изображения выглядит так:

$$P = I \otimes K + N, \tag{1}$$

где P — функция интенсивности искаженного изображения, I — функция интенсивности оригинального изображения, K — ядро размытия (функция рассеяния точки), N — аддитивный двумерный белый Гауссовский шум. За $\otimes$ обозначена операция свертки.

Существующие подходы к процессу восстановления искаженных изображений

В данном разделе описываются типовые подходы к решению задачи восстановления изображения в рамках модели искажений (1). Можно выделить три основные группы алгоритмов восстановления изображений: алгоритмы решения системы алгебраических уравнений, алгоритмы фильтрации изображений в частотной области и итерационные алгоритмы [1].

Основным недостатком алгебраических алгоритмов является необходимость выполнения трудоемких операций обращения и умножения матриц огромных размеров. Основной недостаток алгоритмов фильтрации заключается в том, что они сильно подвержены влиянию краевых эффектов (появляются из-за неизбежной потери информации на краях изображения вследствие размытия или смаза), что зачастую влечет за собой неудовлетворительное качество восстановленных изображений.

Итерационные алгоритмы, которые рассматриваются в данной статье, характеризуются слабой чувствительностью к шуму на изображении. При использовании итерационных алгоритмов, как правило, влияние краевых эффектов на процесс восстановления изображения не столь значительно. Основными же плюсами итерационных алгоритмов являются:

- возможность учесть *априорную* информацию о восстанавливаемом изображении;
- возможность сделать компромиссный выбор между качеством восстановления и скоростью обработки.

Достаточно часто в итерационных методах решается оптимизационная задача следующего вида [2]:

$$(I,\ K) = \arg\min_{I,\ K} \lambda \,||\, I \otimes K - P||_2^2 + ||\nabla_X I||_\alpha + ||\nabla_Y I||_\alpha, \qquad (2)$$

где за $||\nabla_i I||_\alpha$ обозначена L_α норма от матрицы градиентов по оси i, сформированная на основе восстановленного изображения I. Слагаемые $||\nabla_X I||_\alpha$ и $||\nabla_Y I||_\alpha$ являются регуляризационными компонентами в данной оптимизационной задаче. Множитель λ определяет значимость веса ошибки аппроксимации размытого изображения по сравнению с весом регуляризационных слагаемых.

Задача (2) — это задача *слепой деконволюции,* потому что одновременно не известны как I, так и K. В процессе работы итерационного алгоритма,

как правило, происходит итеративное обновление I и K. Ввиду того, что K на любой итерации может плохо аппроксимировать реальное ядро размытия, необходимо на каждой итерации контролировать обновленное изображение I. Задача контроля Iрешается с помощью регуляризационных компонент.

Достаточно часто в регуляризационных компонентах α выбирают равным 1 или 2 [2]. Первый вариант носит название L_1 — регуляризация. Ее основной недостаток заключается в том, что восстановленное изображение, как правило, обладает невысокой резкостью [3]. Второй вариант L_2 — регуляризация или регуляризация Тихонова [4] — имеет недостаток в виде сильной чувствительности к шуму на изображении.

В статье [5] представлена регуляризационная функция:

$$g(I) = \frac{\left\|\nabla_X I\right\|_1 + \left\|\nabla_Y I\right\|_1}{\left\|\nabla_X I\right\|_2 + \left\|\nabla_Y I\right\|_2}. \quad (3)$$

Предполагается, что по мере повышения резкости изображения скорость роста нормы L_1 в числителе будет превосходить скорость роста нормы L_2 в знаменателе, таким образом, значение функции будет расти. По мере размытия изображения скорость убывания нормы L_2 в знаменателе превосходит скорость убывания нормы L_1 в числителе, таким образом, предполагается опять получить рост значений функции. Иными словами, ожидается, что функция будет давать минимум на изображении «хорошего качества».

Исследования показали, что встречаются случаи, когда (3) дает минимум на размытых изображениях. Это связано с тем, что после размытия изображений с сильными перепадами градиентов функции интенсивности (например, зашумленных), в полученных изображениях остаются достаточно сильные перепады градиентов. Это приводит к тому, что по мере размытия изображения норма L_2 убывает не так быстро, как ожидалось, в результате (3) будет минимальна на размытом изображении.

Предложенный способ регуляризации

В данном разделе описывается разработанный регуляризационый компонент и анализируются его свойства.

Разработанная регуляризационная функция имеет вид:

$$g(I) = \frac{\left\|\nabla_X I\right\|_1 + \left\|\nabla_Y I\right\|_1}{\left\| E(I) \right\|_2}, \quad (4)$$

где $E(I) = (\varphi(e_1), \varphi(e_2), \varphi(e_3), ..., \varphi(e_n))^T$ – это вектор ошибок предсказания пикселей изображения I, которые подвергаются операции клиппирования:

$$\varphi(e_i) = \begin{cases} |e_i|, & if\ (|e_i| < \text{threshold}) \\ 0, & otherwise \end{cases} \quad (5)$$

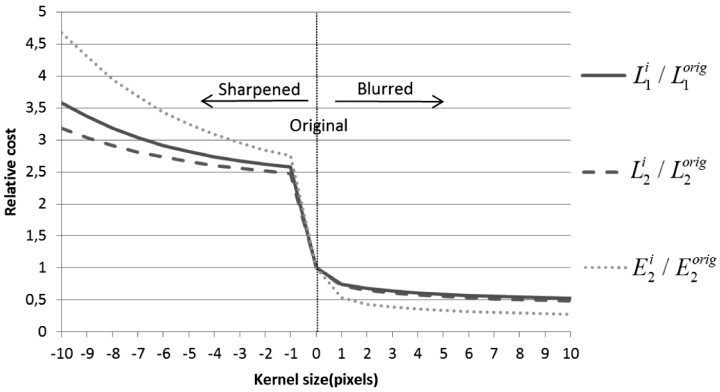


Рис. 1. Сравнение значений различных функций на изображениях с большой и малой резкостью

Для предсказания выбран метод MED из стандарта сжатия изображений JPEG-LS [6].

На рис. 1 представлено сравнение значений различных функций на изображениях с большой и малой резкостью. Для эксперимента было выбрано изображение lena. Положительные значения по оси абсцисс характеризуют размер фильтра, с помощью которого было размыто исходное изображение. Отрицательные значения по оси абсцисс характеризуют размер фильтра, с помощью которого была повышена резкость исходного изображения. По оси ординат отложены значения метрик, являющиеся отношениями полученных значений метрик для разных фильтров к значениям тех же метрик, посчитанных для оригинально изображения. За E_2 обозначена функция $\| E(I) \|_2$

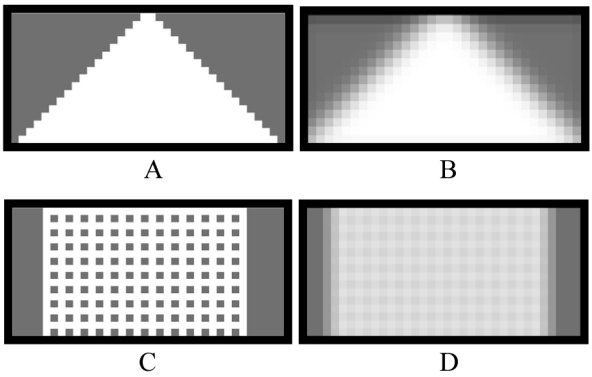


Рис. 2. Тестовые изображения для анализа значений регуляризационных функций. Изображения B, D являются размытыми копиями оригиналов A и C соответственно

для случая неклипированных ошибок. Из Рис. 1 видно, что функции на основе MED предсказания показывают наилучшую чувствительность к изменению резкости на изображении.

В процессе размытия изображения значения регуляризационных компонент L_1 и L_2 на различных участках изображения ведут себя по-разному. В Таблице 1 представлены значения регуляризационных функций для изображений на Рис. 2.

Таблица 1

Значение регуляризационных компонент для изображений на Рисунке 2

	Image A	**Image B**	**Image C**	**Image D**
L_1	8255	27565	62103	13086
L_2	1447	3453	3967	895
E_2	889	374	2214	88
L_1/L_2	5,7	7,98	15,65	14,6
L_1/E_2	9,29	73,7	28,05	148,7

Из таблицы видно, что в результате размытия изображения A значения норм L_1 и L_2 увеличились, с другой стороны, после размытия изображения C значения норм L_1 и L_2 уменьшились. Следствием этого является существенный рост функции (4) при размытии изображений A и C по сравнению с функцией (3). Функция (3) при размытии изображения C, вместо того, чтобы расти, снизилась с 15,65 до 14,6. Это связано с тем, что после размытия на изображении остались сильные градиенты функции интенсивности, которые не позволили обеспечить достаточную скорость убывания нормы L_2 в знаменателе для того, чтобы функция показала возрастающий характер. Тут наглядно видно преимущество предложенной функции на базе MED предсказания, т.к. она имеет минимум на обоих резких оригинальных изображениях.

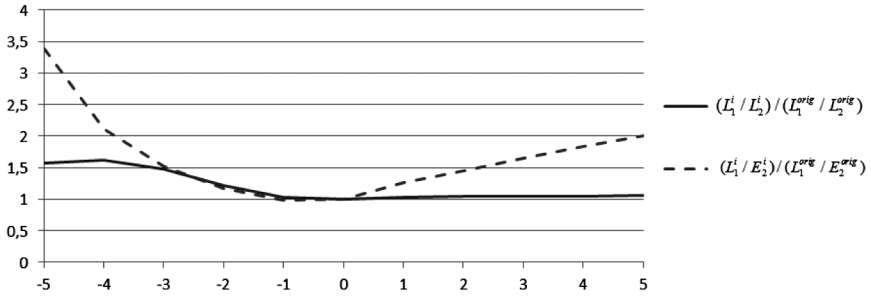


Рис. 3. Сравнение значений функций на изображении lena

На Рис. 3 приводится сравнение функций (3) и (4) для изображения lena. Методика построения графика аналогична графику, представленному на Рис. 1, за исключением того, что тут берутся не десять, а пять размеров фильтров для размытия и для увеличения резкости изображения. Скорость роста функции (3) при размытии мала по сравнению с предложенной функцией (4). Это объясняется наличием шума на изображении.

Результаты эксперимента

В данном разделе проводится сравнение регуляризационных компонент (3) и (4) на примере решения задачи оптимизации. С помощью фильтра Гаусса 6×6 с $\delta = 2{,}5$ были размыты три изображения: lena, baboon и airplane. Для каждого изображения выполнено десять проходов фильтром. Далее для каждого из изображений решалась оптимизационная задача:

$$I = \arg\min_{I} \left\| I \otimes K - P \right\|_2^2 \tag{6}$$

с помощью метода градиентного спуска:

$$, I_{i+1} = I_i - \lambda K^T \otimes (K \otimes I_i - P) \tag{7}$$

После каждой итерации метода градиентного спуска выполнялось клиппирование значений пикселей изображения I в диапазон [0, 255].

В данном эксперименте ставилась задача продемонстрировать, что помимо возможности роста функции (3) по мере размытия изображения, данная функция также может показывать убывающий характер по мере увеличения резкости изображения. Последнее означает, что в случае если метод градиентного спуска будет в качестве критерия останова опираться на показания функции (3), то есть возможность при восстановлении изображения сделать слишком много итераций, вследствие чего, «попасть» в область изображений с завышенной резкостью, качество которых также неприемлемо.

В эксперименте рассматривалось два случая, когда критерием останова выступала функция (3) и когда (4). Стоит отметить, что коэффициент λ в экспериментах выбирался равным 1. При таком значении параметра градиентный спуск с большой вероятность не сходится, но зато обеспечивается малое число итераций.

Результаты восстановления изображений представлены в Таблице 2. В качестве оценки качества восстановления изображений использовался PSNR. Предложенный подход дал выигрыш на всех рассмотренных изображениях. Выигрыш объясняется

Таблица 2

Результаты восстановления изображений

Изображение	Выигрыш, PSNR, дБ
airplane	18,7
baboon	12
lena	21,63

тем, что градиентный спуск с критерием останова в качестве (3) продолжил выполнение после достижения изображения «хорошего качества» и выдал в качестве результата изображение с завышенной резкостью.

Заключение

В данной работе были проанализированы существующие методы регуляризации, были выделены их недостатки. Встречаются случаи, когда метод регуляризации (3) дает минимум как на размытых изображениях вследствие зашумленности изображения или сильных перепадов градиентов функции интенсивности, так и на изображениях с завышенной резкостью. Предложенный регуляризационный компонент (4) за счет использования метода предсказания MED лишен такого недостатка. Это позволило получить выигрыш в качестве восстановления на выборке изображений из «стандартного набора».

Литература

1. *Грузман И. С., Киричук В. С., Косых В. П., Перетягин Г. И., Спектор А. А*. Цифровая обработка изображений в информационных системах. Новосибисрк: Изд-во НГТУ, 2002. 352 с.
2. *Anat Levin, Yair Weiss, Fredo Durand, William T. Freeman*. Understanding and evaluating blind deconvolution algorithms, Computer Vision and Pattern Recognition, 2009 IEEE.
3. *Amir Beck, Marc Teboulle*. A Fast Iterative Shrinkage-Thresholding Algorithm for Linear Inverse Problems. Siam J. Imaging Sciences c_ 2009 Society for Industrial and Applied Mathematics. Vol. 2, No. 1, pp. 183–202, 2009.
4. *Tikhonov, A. N*. (1963). «О решении некорректно поставленных задач и методе регуляризации» [Solution of incorrectly formulated problems and the regularization method]. Doklady Akademii Nauk SSSR 151: 501–504. Translated in Soviet Mathematics 4: 1035–1038.
5. *Dilip Krishan, Terence Tay, Rob Fergus*. Blind Deconvolution Using a Normalized Sparsity Measure, 2011 IEEE.
6. Information technology — Lossless and near-lossless compression of continuous-tone still images — Baseline, ISO/IEC 14495–1, 1999.

НОВЫЙ ПАССИВНЫЙ МЕТОД ОЦЕНКИ КАЧЕСТВА ГОЛОСА В СЕТИ 3GPP LTE

А. И. Акмалходжаев
программист СПбГУАП
E-mail: akmal.ilh@gmail.com

Аннотация. В статье рассматривается пассивный способ оценки качества голоса в беспроводных сетях четвертого поколения на примере 3GPP LTE. Предложен метод, основанный на совместном использовании алгоритмов PESQ и E-модели, который позволяет достичь хорошей точности оценки качества принимаемой речи. Измерения показали, что корреляция с алгоритмом PESQ при использовании нового метода для вокодера AMR-NB достигает 98%.

Введение

Методы оценки качества голоса можно разделить на две группы: объективные и субъективные. Субъективный метод подразумевает оценку качества группой людей (слушателей), которым в лабораторных условиях проигрывают оригинальный и подвергшийся изменениям голосовые фрагменты, после чего человек делает вывод о качестве обработанного сигнала [2]. Среднее значение оценки группы слушателей является показателем качества измененного сигнала. Однако этот метод является дорогостоящим и затратным по времени.

Объективные методы это алгоритмы, которые пытаются рассчитать качество измененного голосового сигнала. Их можно разделить на два вида: активные и пассивные. Для активных алгоритмов обязательным является наличие оригинального голосового сигнала, в то время как для пассивных считается, что оригинальный сигнал отсутствует. Наиболее точным активным алгоритмом на данный момент является PESQ [4] (рекомендация ITU-T. 862). Однако использовать PESQ невозможно, когда речь идет об оценке качества голоса на приемной стороне. В данном случае используют пассивные подходы.

Наиболее распространенный на данный момент пассивный алгоритм для пакетных сетей называется E-моделью и описан в стандарте ITU-T G.107 [3]. Очевидно, что точность пассивных алгоритмов меньше чем активных. В данной статье предлагается метод пассивной оценки качества голоса в сетях четвертого поколения, который позволяет улучшить характеристики стандартной E-модели за счет использования алгоритма PESQ, что возможно в беспроводных пакетных сетях. Точность метода оценивается для сети 3GPP LTE и вокодера AMR-NB [7].

Оценка качества голоса в пакетных сетях

Алгоритм PESQ

PESQ является наиболее известным алгоритмом в настоящее время. При расчете качества он учитывает природу человеческого речеобразования и слуха, за счет чего достигается высокая степень точности. Также алгоритм выравнивает оба сигнала по времени. Таким образом, в итоговой оценке не учитываются ухудшения качества, связанные с временными задержками. Значения на выходе алгоритма варьируются от –0.5 до 4.5 баллов, но могут быть пересчитаны в общепринятую шкалу MOS (Mean Option Score) [2]. MOS принимает значения от 1 до 5, где каждое значение соответствует определенному качеству речи: 5 — прекрасное; 4 — хорошее; 3 — удовлетворительное; 2 — низкое; 1 — плохое. Величину MOS, рассчитанную с помощью алгоритма PESQ, обычно обозначают как MOS-LQO [5].

При проектировании пассивных алгоритмов часто PESQ считают эталонным алгоритмом [1]. Для оценки точности нового алгоритма используют коэффициент корреляции r и $RMSE$, которые вычисляются по следующим формулам [3]

$$r = \frac{\sum(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum(x_i - \overline{x})^2 (y_i - \overline{y})^2}}, \quad (1)$$

$$RMSE = \sqrt{\frac{\sum(x_i - y_i)^2}{n}}, \quad (2)$$

где n — число оценок, x_i — значения MOS-LQO, а y_i — значения MOS, полученные на выходе исследуемого алгоритма.

E-модель

В отличие от PESQ E-модели не нужно знать ни переданный, ни принятый сигналы, т. к. для оценки используются значения факторов, которые привели к ухудшению качества: величина задержки, число потерянных пакетов при передаче, ухудшения из-за использования кодера речи и др. [3]. Оценка, выставляемая методом, называется R-фактором и рассчитывается на основе следующей формулы:

$$R = \mathrm{R}_0 - \mathrm{I}_s - \mathrm{I}_d - \mathrm{I}_{\mathrm{e-eff}} + \mathrm{A}, \quad (3)$$

где: R_0 — исходное значение R-фактора; I_s — искажения вносимые шумами в канале и связанные с обработкой исходного сигнала; I_d — искажения, связанные с задержками в сети; $\mathrm{I}_{\mathrm{e\_eff}}$ — искажения обусловленные использованием вокодера и потерями пакетов в канале; A — коэффициент преимущества. Коэффициент описывает удобства от использования того или иного вида

связи. Например, для проводной связи он равен 0, в то время как для сотовых сетей считается равным 5. R-фактор принимает значения от 0 до 100 и может быть пересчитан в MOS [3]:

$$\mathrm{MOS} = \begin{cases} 1 & R < 0 \\ 1 + 0.035R + R(R-60)(100-R)\times 7\times 10^{-6}, & 100 < R < 0. \\ 4.5 & R > 100 \end{cases} \quad (4)$$

Часто рассматривают упрощенную E-модель, которая учитывает только значение $\mathrm{I}_{\mathrm{e-eff}}$, в то время как остальные факторы принимаются по умолчанию. В этом случае формула 3 упрощается до следующего вида

$$R = \mathrm{R}_0 - \mathrm{I}_{\mathrm{e-eff}}, \quad (5)$$

где $\mathrm{R}_0 = 93.2$. В данном случае значение R учитывает лишь влияние кодера и потери в канале, и такая оценка может сравниваться с рассчитанными значениями PESQ. Для такого сценария точность R-фактора зависит от $\mathrm{I}_{\mathrm{e-eff}}$. По стандарту [3], $\mathrm{I}_{\mathrm{e-eff}}$ вычисляется по следующей формуле

$$\mathrm{I}_{\mathrm{e-eff}} = \mathrm{I}_{\mathrm{e}} + (95 - \mathrm{I}_{\mathrm{e}})\frac{Ppl}{\frac{Ppl}{BurstR} + Bpl}, \quad (6)$$

где I_{e} — описывает искажения, вносимые вокодером; *Bpl* — описывает стойкость вокодера к потерям пакетов; *Ppl* — количество потерянных пакетов в процентах. *BurstR* учитывается, когда пакеты теряются блоками, однако в данной работе потери считаются случайными и *BurstR = 1*. Рассмотрим передачу голоса в сети 3GPP LTE.

Передача голоса в сети 3GPP LTE

Сеть 3GPP LTE является пакетной сетью. При передаче по беспроводному каналу голосовые данные передаются в виде пакетов. Если на физическом уровне проверка CRC для принятого пакета не корректна, то он выбрасывается. Для кодирования речи в стандарте предусмотрено использование кодеков AMR-NB и AMR-WB. В данной статье рассматривается AMR-NB. AMR-NB является узкополосным кодеком и предусматривает несколько скоростей кодирования голоса. Основным в стандарте является режим сжатия 12.2 Кбит/с. Все дальнейшие выкладки приведены для этого режима, однако могут быть легко распространены на другие. Для кодирования речи алгоритм AMR разбивает входной сигнал на сегменты (фреймы) по 20 мс каждый и производит их сжатие. Каждый фрейм пересылается как отдельный пакет. Следовательно, можно сделать вывод, что на качество декодированной речи в 3GPP LTE влияют только потери пакетов и режим сжатия голоса. Если считать, что задержки в сети отсутствуют, то для оценки качества голоса на приемнике можно использовать упрощенную E-модель.

При сжатии кодек AMR с помощью алгоритма VAD оценивает содержит ли фрейм голосовые данные или нет [9]. Как показывает опыт, в ходе разговора каждый участник активен в среднем 50% времени. После периодов активности абонента следует период молчания. AMR учитывает эту специфику и отключает передачу в моменты молчания. Такая техника носит название DTX, при использовании которой сжатый поток представляет собой последовательность голосовых пакетов, за которым следует пакет VAD, после чего передача прекращается до момента новой активности абонента. Так как использование DTX является обязательной в LTE, данная техника учитывалась при проведении моделирования. Также AMR предусматривает алгоритм восстановления потерянного фрейма на основе предыдущих успешно принятых пакетов [8]. Для этого на физическом уровне не декодированный пакет маркируется как потерянный и информация о нем передается на декодер AMR, при этом тип пакета можно считать известным.

Гибридный пассивный метод оценки качества голоса для сети 3GPP LTE

Модель сети 3GPP LTE

Модель передачи данных от одного абонента к другому в сети LTE показана на рисунке 1. Как видно из рисунка потери пакетов происходят при передаче от первого абонентского устройства (АУ1) к базовой станции (БС1) и при передаче от базовой станции (БС2) к абонентскому устройству (АУ2). Пакеты, потерянные между АУ1 и БС1, маркируются и информация о них отправляется на БС2. Т. е. БС2 знает какие пакеты были потеряны на данном участке и передает их АУ2, которое декодирует принятый поток. В модели считается, что множества потерянных пакетов между АУ1 и БС1, и между БС2 и АУ2 не пересекаются. Обозначим количество ошибок в процентах от общего числа переданных пакетов на участке АУ1-БС1 как Ppl_1, а на участке БС2- АУ2 как Ppl_2.

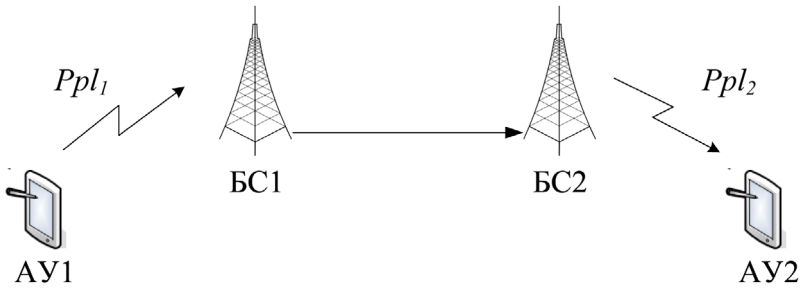


Рис. 1. Схема сети 3GPP LTE

Введем следующие обозначения: S_{or} — исходный речевой сигнал; S_{syn} — речевой сигнал, полученный при декодировании AMR потока без ошибок; S_{err1} — речевой сигнал, синтезированный из AMR потока, в котором учтены потерянные пакеты между АУ1 и БС1; S_{err2} — синтезированный речевой сигнал, в котором учтены потери пакетов между БС2 и АУ2; S_{dg} — результирующий речевой сигнал, для которого должна быть получена оценка качества и в котором учтены все ошибки. *Ppl* для S_{dg} рассчитывается как $Ppl_1 + Ppl_2$. Так как в LTE используется алгоритм HARQ и производится сигнализация о том, был ли пакет потерян или принят успешно, то справедливы следующие положения: S_{err1} может быть синтезирован на БС1 и БС2, так как известны позиции потерянных пакетов; S_{dg} известна на АУ2 и БС2; S_{or} известен лишь АУ1. Идеальная оценка качества голоса получается при анализе S_{or} и S_{dg} алгоритмом PESQ. Сравнение S_{or} и S_{syn} показывает, на сколько снизилось качество речи из-за использования кодека. При сравнении сигналов S_{err1} и S_{err2} с MOS-LQO будут показывать влияние соответствующих потерянных пакетов между соответствующими БС и АУ.

Описание улучшенного метода оценки качества голоса для сети 3GPP LTE

Качество принимаемой речи может быть оценено как на БС2, так и на АУ2. Так как в сети все решения о выборе скорости кода канала и выделении частотно-временного ресурса выполняет базовая станция оценка качества на БС2 выглядит предпочтительнее. Имея такую информацию БС2 может более гибко контролировать параметры передачи. Как было отмечено БС2 имеет информацию о S_{err1} и S_{dg}, в то время как S_{err1} на АУ2 не известно. Таким образом, БС2 имеет больше информации для выполнения более точной оценки, что является еще одним плюсом при выполнении измерения на БС2. Поэтому в работе предлагается оценивать качество речи именно на базовой станции.

Отметим, что при использовании E-модели считается, что каждый потерянный пакет одинаково влияет на ухудшение качества декодированной речи, но это не так. В ходе работы было замечено, что MOS-LQO сильно зависит от того, какой пакет был потерян. Т.е. не все ошибки имеют одинаковое влияние на качество сигнала. Если есть возможность пересчитать *Ppl* в зависимости от фактического воздействия ошибок на качество принимаемой речи, то точность E-модели может быть улучшена. Покажем, что на БС2 такая возможность есть.

Заметим, что разница между S_{err1} и S_{dg} обусловлена ошибками между БС2 и АУ2. Поэтому сделано предположение, что при сравнении этих сигналов алгоритмом PESQ полученное значение MOS-LQO будет отображать ухудшение качества, которое вызывают ошибки Ppl_2. Чтобы подтвердить этот факт, было проведено моделирование, в ходе которого в качестве исходного

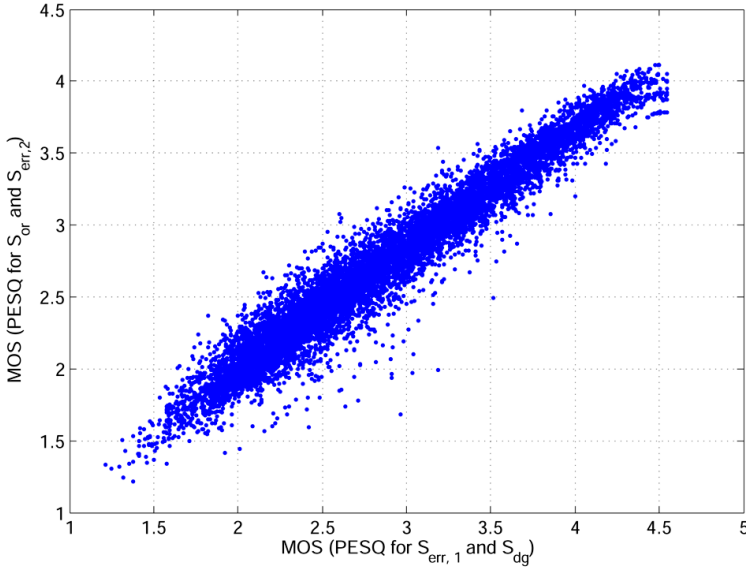


Рис. 2. Сравнение MOS-LQO для сигналов (S_{or}, S_{err2}) и (S_{err1}, S_{dg})

сигнала использовались 8 секундные речевые сегменты на американском английском с сайта ITU-T [6]. 28 доступных сигналов были разбиты на две группы: 16 сигналов для проведения измерений параметров предложенного алгоритма, 12 для проверки точности предложенного алгоритма. Для 16 сигналов моделировались независимые ошибки между БС1 и АУ1, и БС2 и АУ2. При моделировании были рассмотрены значения *Ppl* от 1% до 20% с шагом 1%. Для каждого значения *Ppl* генерировалось 30 различных шаблонов потерянных пакетов, где далее случайным образом выбиралась половина ошибок и относилась к Ppl_1, а оставшаяся часть к Ppl_2. Для полученных в результате моделирования пар сигналов (S_{err1}, S_{dg}), и (S_{or}, S_{err2}) были рассчитаны значения MOS-LQO. Оценка корреляции для двух множеств показала значение 98%, что видно на рисунке 2. Это говорит о том, что сделанное предположение верно и сравнение пары (S_{err1}, S_{dg}) отражает ухудшение качества речи, вызванное ошибками Ppl_2. Т.е. можно измерить фактическое влияние этих ошибок на качество голоса. Тогда новый алгоритм может быть сформулирован следующим образом.

На БС2 с помощью алгоритма PESQ рассчитывается новое значение Ppl_2, которое отображает фактическое воздействие ошибок на качество речи. Обозначим его как $Ppl_{2,new}$. Для этого вычислим $MOS\text{-}LQO_{syn}$ для S_{err1} и S_{dg}. Полученное число не описывает ухудшения качества из-за использования кодека, т.к. сравниваются два синтезированных сигнала и это должно быть учтено. Влияния кодека можно учесть в среднем, для чего необходимо пересчитать

MOS-LQO$_{syn}$ в MOS-LQO$_{or}$. Для того чтобы найти правило пересчета с помощью моделирования были рассчитаны значения MOS-LQO для пар сигналов (S_{or}, S_{dg}) и (S_{syn}, S_{dg}). MOS-LQO для первой пары отображают ухудшение качества из-за использования кодека, в то время как MOS-LQO для второй пары этой информации не содержат. Корреляция полученных значений MOS-LQO составила 98%, что говорит о том, что зависимость между MOS-LQO$_{syn}$ и MOS-LQO$_{or}$ линейная и может быть представлена следующей формулой

$$MOS - LQO_{\text{or}} = A \cdot MOS - LQO_{\text{syn}} + B, \tag{7}$$

где A и B некоторые коэффициенты. Моделирование с использованием 16 описанных выше речевых сигналов показало, что оптимальным с точки зрения минимума среднеквадратической ошибки является выбор $A = 0.8770$ и $B = 0.1738$. Далее MOS-LQO$_{or}$ пересчитывается в значение R-фактора по следующей формуле [1]:

$$\text{R} = 3.026MOS^3 - 25.314MOS^2 + 87.060MOS - 57.336, \tag{8}$$

после чего легко получить новое значение Ppl_2 из выражений 5 и 6 как

$$Ppl_{2,new} = \frac{R_0 - R - I_e}{95 - R_0 + R} Bpl, \tag{9}$$

где для режима AMR-NB 12.2 Кбит/с $Bpl = 10$, $I_e = 5$ [3].

Имея оцененное значение $Ppl_{2,\,new}$ и значение Ppl_1, выполняется оценка качества по E-модели для нового значения

$$Ppl_{new} = Ppl_{2,\,new} + Ppl_1.$$

Результаты моделирования

Для проверки точности нового подхода проведено моделирования для описанных выше 12 речевых сигналов. Поскольку точность предложенного алгоритма зависит от значений Ppl_2 и Ppl_1, отдельно были рассмотрены следующие случаи:

$$\frac{Ppl_2}{Ppl} = 0.25, \quad \frac{Ppl_2}{Ppl} = 0.5, \quad \frac{Ppl_2}{Ppl} = 0.75.$$

Моделирование проводилось для значений Ppl от 0% до 20% с шагом 1%. В каждом случае для каждого исследуемого речевого сигнала генерировалось 30 различных шаблонов потерянных пакетов, для которых оценивалось MOS по алгоритму PESQ и E-модели. В зависимости от значения $\frac{Ppl_2}{Ppl}$ случайным образом некоторые потерянные пакеты относились к Ppl_1, другие к Ppl_2.

Полученные значения корреляции и RMSE для стандартной E-модели составили $r = 0.92$ и $RMSE = 0.35$. Для предложенного алгоритма результаты моделирования представлены в таблице 1. Как видно за счет использования дополнительной информации на БС2, удается улучшить значение корреляции и значительно исправить значение RMSE. Если $Ppl_2 = Ppl$, то значение корреляции может достигать 99%.

Таблица 1

Результаты моделирования для предложенного метода оценки качества голоса

$Ppl_2 = Ppl$	0.25	0.5	0.75
r (%)	94	96	98
RMSE	0.28	0.22	0.18

Заключение

В работе предложен новый метод пассивной оценки качества голоса для беспроводных сетей четвертого поколения. Предлагается оценивать качество голоса на базовой станции, что позволяет использовать алгоритм PESQ для улучшения точности оценки. Таким образом, новый метод представляет собой гибридную схему, основанную на E-модели и алгоритме PESQ. Результаты моделирования для сети 3GPP LTE и вокодера AMR-NB показали значительное улучшение качества оценки по сравнению с обычной E-моделью. Плюсом данного метода является то, что в случае необходимости могут быть учтены и другие факторы, влияющие на качество голоса и предусмотренные E-моделью.

Литература

1. *L. Sun, E. Ifeachor*. Voice quality prediction models and their application in VoIP networks // IEEE Transactions on Multimedia. Vol. 8. No. 4. 2008. Pp. 809-820.
2. ITU-T, Recommendation P. 800; Methods for subjective determination of transmission quality. ITU-T Std. June 1996.
3. ITU-T, Recommendation G. 107; The E-model: a computational model for use in transmission planning. ITU-T Std. December 2011.
4. ITU-T Recommendation P.862; Perceptual evaluation of speech quality (PESQ): An objective method for end-to-end speech quality assessment of narrow-band telephone networks and speech codecs. ITU-T Std. February 2001.
5. ITU-T, Recommendation P.862.1; Mapping function for transforming P.862 raw result scores to MOS-LQO, International Telecommunication Union-Telecommunication Standardization Sector (ITU-T). ITU-T Std. March 2005.
6. ITU-T, Recommendation P.834; Methodology for the derivation of equipment impairment factors from instrumental models. ITU-T Std. March 2005.

7. 3GPP TS 26.101: “ Mandatory Speech Codec speech processing functions; Adaptive Multi-Rate (AMR) speech codec; Transcoding functions (Release 11)”, 3GPP Std.
8. 3GPP TS 26.091: “Mandatory Speech Codec speech processing functions; Adaptive Multi-Rate (AMR) speech codec; Error concealment of lost frames (Release 11)”, 3GPP Std.
9. 3GPP TS 26.094: “Mandatory speech codec speech processing functions; Adaptive Multi-Rate (AMR) speech codec; Voice Activity Detector (VAD) (Release 11)”, 3GPP Std.

АЛГОРИТМ ПЛАНИРОВАНИЯ РЕСУРСОВ НА БАЗОВОЙ СТАНЦИИ С УЧЕТОМ ТРЕБОВАНИЙ К КАЧЕСТВУ ОБСЛУЖИВАНИЯ ПОЛЬЗОВАТЕЛЕЙ[1]

А. В. Борисовская

аспирант кафедры инфокоммуникационных систем СПбГУАП

E-mail: solovyeva@vu.spb.ru

И. А. Пастушок

студент кафедры безопасности информационных систем СПбГУАП

E-mail: i.pastushok@vu.spb.ru

Аннотация. Статья посвящена современным системам передачи видеоинформации, построенным на основе стандарта LTE. Основное внимание уделяется анализу работы алгоритмов распределения ресурсов между пользователями с различными требованиями к качеству обслуживания. Алгоритмы планирования ресурсов не стандартизованы, поэтому в данной работе предложены критерии эффективности их работы. И представлена методика численного расчета верхних оценок для этих критериев. Также предложен и реализован вариант алгоритма распределения ресурсов на базовой станции, учитывающий требования QoS, такие как максимальное время обслуживания, гарантированная и максимальная скорость передачи данных. Показатели эффективности этого планировщика близки к граничным значениям.

Введение

В настоящее время беспроводные мультимедиа технологии приобретают все большую популярность. Самой распространенной из них является передача интерактивного и потокового видео по сети (видеоконференции, видео блоги, обмен видео файлами и т.п.) [1]. Беспроводные сети могут содержать достаточно большое количество пользователей, которым необходимо обеспечивать требуемое качество обслуживания (Quality of Service или QoS). В стандарте LTE закреплена возможность обеспечения требуемого уровня к качеству обслуживания, путем внесения дополнительных изменений в алгоритм распределения ресурсов на базовой станции [2]. Следовательно, производительность системы передачи видеоинформации зависит от эффективности работы алгоритма планирования ресурсов, реализованного на базовой станции. Данные алгоритмы не стандартизованы, и являются коммерческой тайной операторов

[1] Работа выполнена по плану работ, представленному в заявке № 14-11-00644 на конкурс грантов Российского научного фонда «Проведение фундаментальных научных исследований и поисковых научных исследований отдельными научными группами».

мобильной связи и производителей базовых станций, поэтому задача оценки их эффективности является актуальной. Разработчики современных систем передачи видео данных используют эвристические алгоритмы планирования ресурсов, для проверки работы которых, они используют системы имитационного моделирования (СИМ). Самой последней разработкой в данной области является система имитационного моделирования NS-3, которая соответствует последнему стандарту связи 3GPP LTE. Однако в СИМ NS-3 реализованы планировщики, которые учитывают лишь некоторые требования QoS. Таким образом, целью данной работы является разработка эффективного алгоритма планирования распределения ресурсов на базовой станции, учитывающая стандартизованные требования к качеству обслуживания пользователей.

Структура систем передачи видеоинформации

Современные системы передачи видеоинформации состоят из трех основных компонентов (Рисунок 1):

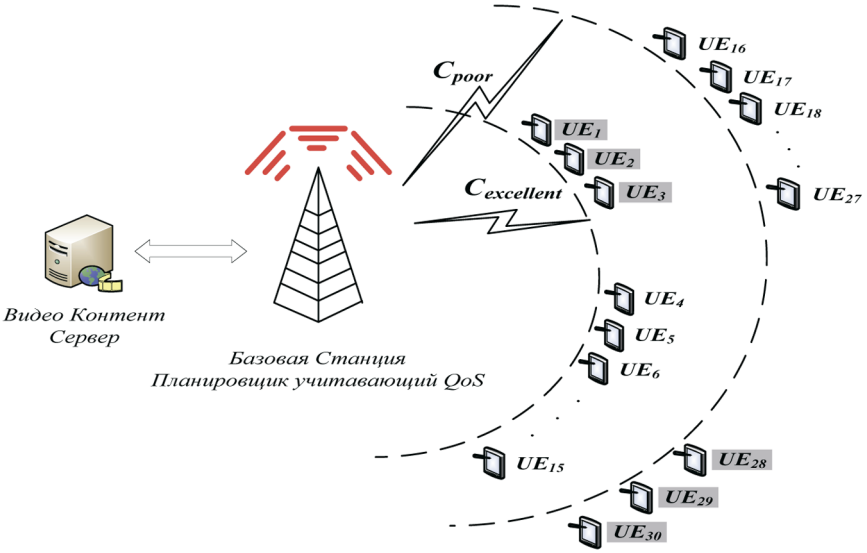


Рис. 1. Структура системы передачи видеоинформации

- Видео контент сервера являются хранилищами видео контента в различных качествах;
- Базовая станция — устройство, организовывающее беспроводное подключение между пользователем и видео сервером. Основным компонентом, данного устройства, является планировщик распределения ресурсов беспроводного канала. Планировщик распределяет ресурсы канала ис-

ходя из эвристики, реализованной в нем, входного потока и требований к качеству обслуживания абонентов [3];

- Адаптивный видео клиент — программное обеспечение, установленное на пользовательском устройстве, которое, исходя из оценки скорости передачи данных по каналу, запрашивает видео в определенном качестве с видеосервера.

Такие системы очень чувствительны к производительности сетей передачи информации. Наибольший вклад в производительность системы в целом вносит алгоритм планирования распределения ресурсов, установленный на базовой станции.

Пусть в системе находится N пользователей, каждый пользователь может развить скорость передачи данных по беспроводному каналу равную $C_i^{\max}$, если ему будут отданы все ресурсы канала. Данная величина зависит от радио условий, в которых находится пользователь. Реальная скорость передачи данных пользователя зависит как от радио условий, так и от общего количества абонентов в системе. Так как скорость передачи данных может изменяться во времени, то введем в рассмотрение среднюю скорость передачи данных пользователя S_i. По определению средняя скорость передачи данных — предел отношения количества переданных данных ко времени их передачи, устремлённому к бесконечности:

$$S_i = \lim_{t\to\infty} \frac{D_i(t)}{t}. \quad (1)$$

Критерии эффективности алгоритма планирования

Для оценки эффективности работы алгоритма распределения ресурсов введем следующие критерии:

- Скорость передачи данных привилегированной группы пользователей;
- Коэффициент использования канала, который может быть рассчитан следующим образом:

$$\eta = \sum_{i=1}^{N} \frac{S_i}{C_i^{\max}}. \quad (2)$$

Первый критерий показывает, насколько эффективно выполняется политика оператора в системе, а второй — насколько эффективно алгоритм планирования распределяет ресурсы беспроводного канала передачи информации.

Расчет скорости передачи данных по HTTP протоколу

Между клиентом и видео сервером установлено соединение вида HTTP. Поток данного вида представляет собой последовательность из запросов пользователей и ответов сервера (Рисунок 2).

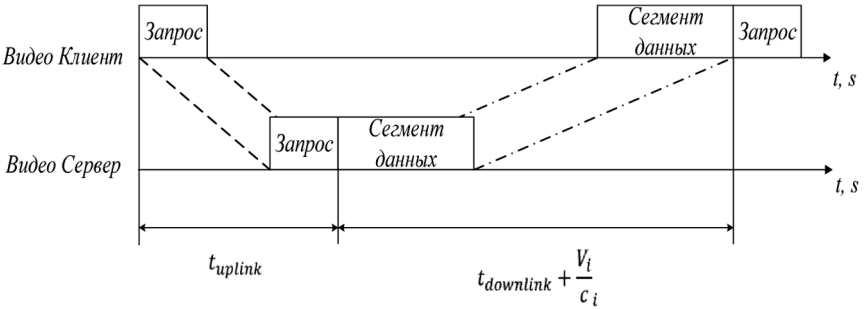


Рис. 2. Временная диаграмма передачи данных по HTTP протоколу

Покажем, как может быть вычислена средняя скорость передачи данных, согласно выражению (1). Так как информация разбита на сегменты, то количество переданных данных:

$$D_i(t) = n(t)V_i, \tag{3}$$

где $n(t)$ — количество скаченных сегментов за время t, V_i — объем одного сегмента пользователя i. Время передачи данных:

$$t = \sum_{s_{num}=1}^{n(t)} \left(rtt_{(s_{num},\, i)} + \frac{V_i}{c_i} \right) = \sum_{s_{num}}^{n(t)} rtt_{(s_{num},\, i)} + n(t)\frac{V_i}{c_i}, \tag{4}$$

где c_i — скорость передачи данных по беспроводному каналу пользователя i, и $rtt_{(s_{num},\, i)}$ — значение задержки в канале во время скачивания сегмента s_{num} пользователем i.

Первое слагаемое из (4) умножим и разделим на $n(t)$, тогда выражение (4) примет следующий вид:

$$t = n(t)\frac{\sum_{s_{num}}^{n(t)} rtt_{(s_{num},\, i)}}{n(t)} + n(t)\frac{V_i}{c_i} = n(t)\left(\overline{rtt_i} + \frac{V_i}{c_i} \right). \tag{5}$$

Подставим полученные результаты (3) и (5) в выражение (1), тогда итоговое значение скорости передачи данных пользователя:

$$S_i = \frac{V_i}{\overline{rtt_i} + \dfrac{V_i}{c_i}}. \tag{6}$$

Наибольшее влияние на полученное выражение (6) вносит значение скорости передачи данных по беспроводному каналу пользователя (c_i). Данная величина определяется алгоритмом планирования, установленным на базовой станции.

Стратегия планирования Round-Robin

Основной эвристикой распределения ресурсов беспроводного канала в частотной области является стратегия Round-Robin [4]. Данная стратегия ставит себе задачей обеспечение равного доступа к ресурсам канала между активными пользователями, путем циклического распределения частотно-временных единиц ресурсов беспроводного канала (Рисунок 3).

Рис. 3. Стратегия планирования Round-Robin

Следовательно, скорость передачи данных, при использовании стратегии Round-Robin за одну секунду, можно представить следующим выражением:

$$c_i = \frac{1000 \cdot N_{RBG}}{N} \cdot mcs_i, \tag{7}$$

где N_{RBG} — количество единиц частотно-временных ресурсов (Групп Ресурсных Блоков) канала в одном периоде планирования (подкадр), mcs_i — количество бит, передаваемое в одной группе ресурсных блоков i-го пользователя, данная величина зависит от радио условий, в которых находится пользователь, N — число ак тивных пользователей в системе.

Стратегия планирования Round-Robin с учетом требований QoS

В стандарте связи LTE закреплены следующие основные параметры требований к качеству обслуживания [2]:

- $S^i_{\max}$ — максимальная скорость передачи данных;
- $S^i_{\min}$ — гарантированная скорость передачи данных;
- τ_i — максимальное время обслуживания потока ($\tau_i < 1$ с).

Для обеспечения требований к качеству обслуживания абонентов все время планирования разбивается на интервалы равные τ_i. В начале каждого интервала вычисляется объём данных, который пользователь может передать в нем:

$$q_i(t) = \tau_i \cdot c_i(t),$$

где $c_i(t)$ — оценка скорости передачи данных в момент времени t. Далее общий объем данных, доступный для передачи пользователя в планируемом интервале вычисляется как:

$$Q_i(t) = \min\{Q_i(t) + q_i(t),\ c_i(t) \cdot 1c\}.$$

Ключевым моментом предложенного алгоритма является получение верхней оценки скорости передачи данных в момент времени t. Для этого был предложен следующий алгоритм.

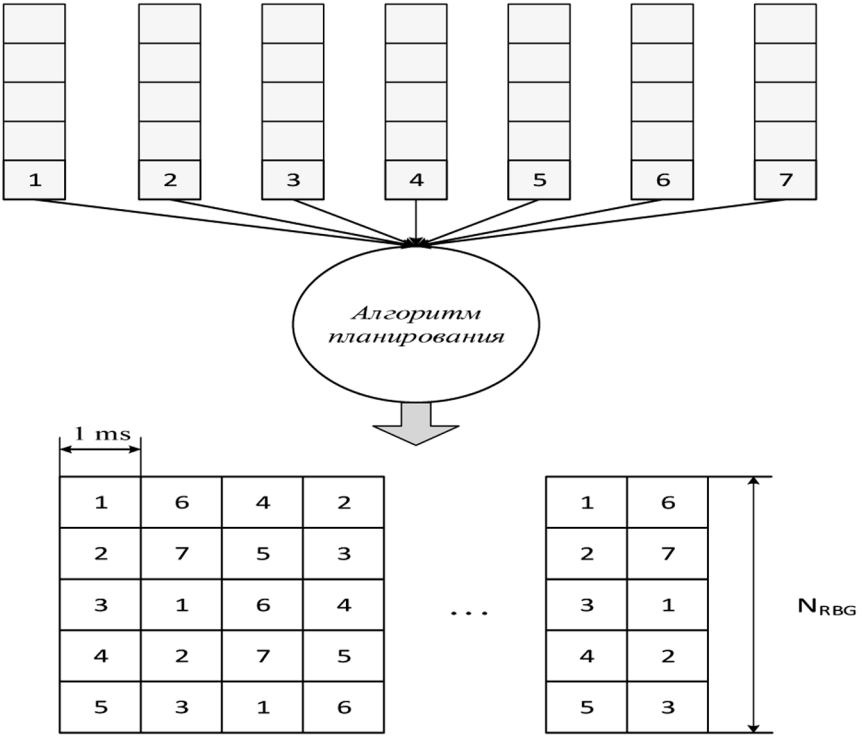


Рис. 4. Стратегия планирования Round-Robin с учетом QoS

На вход алгоритма подаются значения:

- $mcs = \{mcs_1, mcs_2, ..., mcs_N\}$ — информация о качестве канала пользователей;
- $d = \{d_1, d_2, ..., d_N\}$ — ограничения скоростей пользователей, выставленные оператором мобильной связи;
- $g = \{g_1, g_2, ..., g_N\}$ гарантированная скорость передачи;
- N_{RBG} — количество Групп Ресурсных Блоков в одном подкадре;

Выход:

- $c = \{c_1, c_2, ..., c_N\}$ — скорость передачи данных пользователей беспроводному по каналу.

Действие алгоритма выполняется в пять шагов.

На первом шаге алгоритма вычисляется скорость передачи данных по беспроводному каналу без учета ограничений, выставленных на стороне оператора, согласно выражению (7):

for все пользователи

$$c_i = \frac{1000 \cdot N_{RBG}}{N} \cdot mcs_i;$$

for end;

На втором шаге необходимо перераспределить ресурсы беспроводного канала, с учетом выставленных ограничений оператора. Для этого определяются пользователи, скорость передачи данных у которых превышает ограничение установленное оператором. На основе выставленных ограничений перераспределяются частотно-временные ресурсы канала, с учетом того, что пользователь с ограничением не участвует в распределении ресурсов после того как реализовал установленную скорость передачи данных.

$while\left(\exists d_i < c_i\right)$

$for\ all\ i : (d_i < c_i)\ \&\ (i \notin M_{limitedUsers})$

$$c_i = \frac{d_i}{mcs_i} \cdot mcs_i;$$

$$N_{reservedRBG} += \frac{d_i}{mcs_i};$$

$$M_{limitedUsers} = M_{limitedUsers} \bigcup \{i\};$$

$for\ end$;

$for\ all\ i \notin N_{limitedUsers}$

$$c_i = \frac{1000 \cdot N_{RBG} - N_{reservedRBG}}{N - |M_{limitedUsers}|} \cdot mcs_i;$$

$for\ end$;

$while\ end$;

На третьем шаге, после того как было установлена максимальная скорость передачи для каждого пользователя, необходимо найти тех абонентов, у которых не выполняются требования QoS по параметру гарантированной скорости передачи данных. Так же необходимо определить требуемый ресурс для того чтобы данные требования выполнялись:

for all users

if $(c_i < g_i)$

$$N_{needlyRBG} += \frac{g_i - c_i}{mcs_i};$$

$$M_{needlyUsers} = M_{needlyUsers} \bigcup \{i\};$$

if end;

for ens;

На четвертом шаге, определив необходимый ресурс для выполнения требований к качеству обслуживая пользователей, происходит поиск доступных ресурсов в системе:

$$
\begin{aligned}
&\textit{while } ((N_{aveableRBG} \neq N_{needlyRBG}) \,||\, (N_{discartedUsers} \neq N - \left|M_{needlyUsers}\right|)) \\
&\quad \textit{if } (i \notin M_{limitedUsers}) \\
&\qquad \textit{if } \left(c_i - mcs_i > g_i\right) \\
&\qquad\quad c_i = c_i - mcs_i; \\
&\qquad\quad N_{aveableRBG} += 1; \\
&\qquad \textit{else} \\
&\qquad\quad N_{discartedUsers} += 1; \\
&\qquad \textit{if end}; \\
&\quad i += 1; \\
&\quad \textit{if } (i == N) \\
&\qquad i = 0; \\
&\quad \textit{if end}; \\
&\textit{while end};
\end{aligned}
$$

На пятом шаге найденный ресурс циклически распределяется между пользователями, у которых не выполняются требования QoS по значению гарантированный скорости передачи данных.

На основании полученной оценки скорости передачи данных по беспроводному каналу можно оценить скорость передачи данных по протоколу HTTP, на основе выражения (6). И на основе выражения (2) можно рассчитать теоретическое значение коэффициента использования канала:

for все пользователи

$$S_i = \frac{V_i}{\overline{rtt_i} + \frac{V_i}{c_i}};$$

for end;

Параметры и результаты моделирования

Для сравнения аналитических результатов с моделированием в СИМ NS-3 был использован типовой сценарий (Рисунок 1) для систем передачи видеоинформации (Таблица 1). В данном сценарии предполагалось, что действие политики оператора распространяется только на непривилегированную

группу пользователей. Под привилегированными будем понимать пользователей с высокими требованиями к качеству обслуживания.

Таблица

Параметры моделирования

Параметр моделирования	Значение
Количество пользователей	30
Набор битрейтов на сервере	{0.6, 1, 2.5} Mbps
Процент привилегированных пользователей в системе	20%
Максимальная скорость передачи в хороших радио условиях ($C_{excllent}$)	55 Mbps
Максимальная скорость передачи в плохих радио условиях (C_{poor})	25 Mbps
Процент пользователей в хороших радио условиях	50%
Среднее значение задержки в канале	50 мс
Количество Групп Ресурсных Блоков в одном подкадре	25

Из результатов моделирования (Рисунок 5, Рисунок 6) следует, что реализованный планировщик Round-Robin, учитывающий требования QoS, удовлетворяет введенным критериям эффективности.

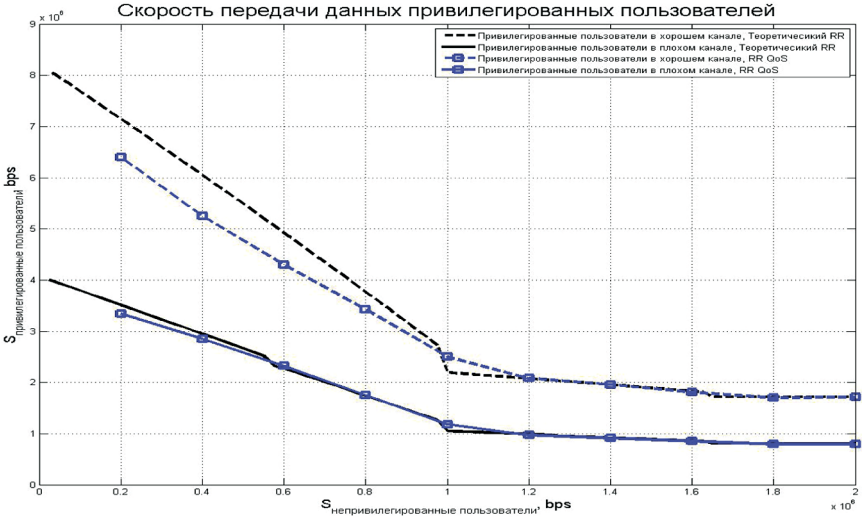


Рис. 5. Скорость передачи данных привилегированной группы пользователей

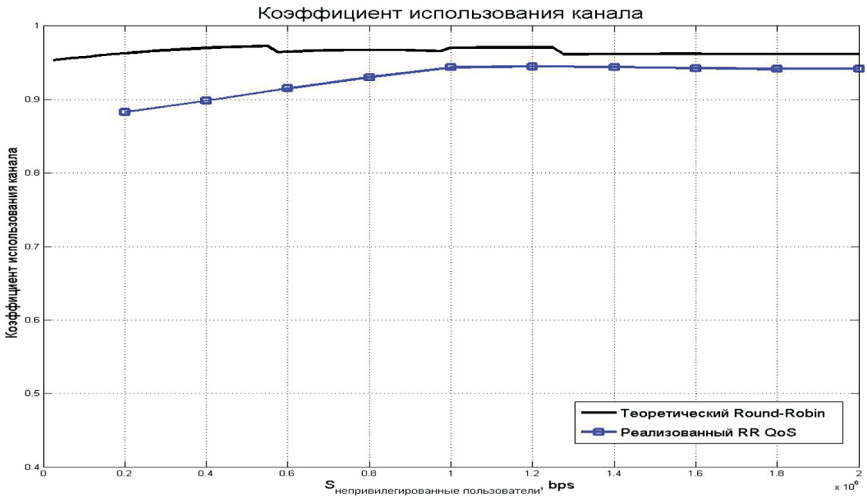


Рис. 6. Коэффициент использования канала

Заключение

В результате данной работы предложена и реализована стратегия планирования Round-Robin, удовлетворяющая критериям эффективности работы системы и учитывающая требования к качеству обслуживания пользователей (QoS). Эффективность алгоритма планирования продемонстрирована на примере системы передачи видеоинформации в СИМ NS-3.

Литература

1. *O. Oyman* and *S. Singh*. «Quality of Experience for HTTP Adaptive Streaming Services», IEEE Communication Magazine, vol. 4, pp. 20–27, April 2012.
2. 3GPP TS 23.203 V11.6.0 3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; Policy and charging control architecture (Release 11).
3. *Chris Johnson*. Long Tern Evolution IN BULLETS 1st Edition. 2010. 279 p.
4. LTE — The UMTS Long Term Evolution: From Theory to Practice / Eds. Stefania Sesia, Issam Toufik, Matthew Baker. John Wiley & Sons, Ltd., 2009. 611 p.

КРИПТОГРАФИЧЕСКИЕ СИСТЕМЫ ОСНОВАННЫЕ НА СПАРИВАНИИ

Е. С. Востокова

Санкт-Петербургский государственный Университет,
Математико-механический факультет

E-mail: lizk.vostokova@gmail.com

В 1975 году появилась криптография с открытым ключом, в результате чего все предложенные в дальнейшем асимметричные системы шифрования были основаы на некоторой вычислительно «трудной» задаче. Появление каждой такой системы приводило к пристальному изучению задачи, которая обеспечивала её безопасность. Так, после появления RSA началось изучение задачи факторизации чисел, а после появления протоколов Диффи—Хэллмана и системы Эль-Гамаля в центре внимания оказалась задача дискретного логарифмирования. В результате появились алгоритмы, такие как ρ- и χ-методов Полларда, общий метод решета числового поля и др., которые углубили понимание сущности этих задач и позволили решать их значительно быстрее, чем изначально предполагали авторы перечисленных криптосистем. Однако ни про задачу дискретного логарифмирования, ни про задачу факторизации не доказано, что они принадлежат классу NP-полных задач. Кроме того существуют алгоритмы решения этих задач для квантовых компьютеров, выполнимые за полиномиальное время. Совокупность вышеозначенных соображений приводит к необходимости продолжать поиск новых трудноразрешимых задач, на основе которых можно строить системы шифрования.

В данной работе предлагается новый подход к построению асимметричных систем шифрования на основе билинейного спаривания и описывается подход, позволяющий использовать его для построения криптосистем. Изложена общая схема таких систем, а также, несколько примеров, её реализующих.

Распараллеливание в OPEN MP и сплайновые аппроксимации

Бурова
Ирина Герасимовна

д.ф.-м.н.
профессор кафедры параллельных алгоритмов СПбГУ

О ПОСТРОЕНИИ ПОЛИНОМИАЛЬНЫХ И ТРИГОНОМЕТРИЧЕСКИХ АППРОКСИМАЦИЙ ТРЕТЬЕГО ПОРЯДКА

И. Г. Бурова, Н. С. Домнин

Пусть n — натуральное число, a, b — вещественные числа, $\{x_j\}$, $j = 0, 1, \ldots, n+1$ — равномерная сетка узлов с шагом h, $x_j = a + jh$, $b = x_{n+1}$.

Пусть известны значения функции $f \in C^3[a,b]$, $f(x_j - h/2)$, $j = 1, \ldots, n$.

Полиномиальные приближения B-сплайнами третьей степени задаем формулами

$$\tilde{u}(x) = C_j w_3(x_j + th) + C_{j+1} w_2(x_j + th) + C_{j+2} w_1(x_j + th)),$$

$$x \in [x_j, x_{j+1}], t \in [0,1],$$

здесь базисные полиномиальные функции $w_i(x_j + th)$, $i = 1, 2, 3$, определяем соотношениями [1]

$$w_1(x_j + th) = t^2/2,\ w_3(x_j + th) = (t-1)^2/2, w_2(x_j + th) = 1/2 + t - t^2.$$

Коэффициенты C_j находим решая систему уравнений $MC = F$, где

$$M = \begin{pmatrix} 3/4 & 1/8 & 0 & \ldots & 0 & 0 \\ 1/8 & 3/4 & 1/8 & \ldots & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & 3/4 & 1/8 \\ 0 & 0 & 0 & \ldots & 1/8 & 3/4 \end{pmatrix},\ f = \begin{pmatrix} f(x_1 - h/2) - f(x_0 - h/2)/8 \\ f(x_2 - h/2) \\ \ldots \\ f(x_{n-1} - h/2) \\ f(x_n - h/2) - f(x_{n+1} - h/2)/8 \end{pmatrix},$$

2. Тригонометрические приближения B-сплайнами третьей степени задаем формулами

$$\tilde{u}(x) = C_j w_3(x_j + th) + C_{j+1} w_2(x_j + th) + C_{j+2} w_1(x_j + th)),\ x \in [x_j, x_{j+1}], t \in [0, 1],$$

здесь тригонометрические базисные сплайны определяем соотношениями [2]

$$w_1(x_j + th) = \frac{(-1 + \cos(th))}{(2\cos(h) - 2)},\ w_2(x_j + th) = \frac{(2\cos(h) - \cos(th - h) - \cos(th))}{(2\cos(h) - 2)},$$

$$w_3(x_j + th) = \frac{(2\sin(h) - \sin(th) + \sin(-2h + th))}{(-2\sin(2h) + 4\sin(h))},$$

а коэффициенты C_j находим как решение системы уравнений $MC = F$, где

$$M = \begin{pmatrix} A_2 & A_1 & 0 & \dots & 0 & 0 \\ A_3 & A_2 & A_1 & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & A_2 & A_1 \\ 0 & 0 & 0 & \dots & A_3 & A_2 \end{pmatrix}, \quad F = \begin{pmatrix} f(x_1 - h/2) - A_3 f(x_0 - h/2) \\ f(x_2 - h/2) \\ \dots \\ f(x_{n-1} - h/2) \\ f(x_n - h/2) - A_1 f(x_{n+1} - h/2) \end{pmatrix},$$

$$A_1 = \frac{(-1 + \cos(h/2))}{(2\cos(h) - 2)}, \quad A_2 = \frac{(2\cos(h) - \cos(-h/2) - \cos(h/2))}{(2\cos(h) - 2)},$$

$$A_3 = \frac{(2\sin(h) - \sin(h/2) + \sin(-3h/2))}{(-2\sin(2h) + 4\sin(h))},$$

Системы уравнений с трехдиагональными матрицами решались методом встречной прогонки с распараллеливанием на два процессора.

Результаты аппроксимации некоторых функций полиномиальными и тригонометрическими сплайнами на промежутке [–1, 1] приведены в таблицах 1, 2 и представлены на рис. 1, 2.

Таблица 1

Фактические погрешности приближения полиномиальными сплайнами

f	$h = 0.1$	$h = 0.01$	$h = 0.001$	$h = 0.0001$
$1/(1+25x^2)$	$1.384 \cdot 10^{-2}$	$4.727 \cdot 10^{-6}$	$4.669 \cdot 10^{-9}$	$1.869 \cdot 10^{-11}$
$\sin(x)$	$8.046 \cdot 10^{-5}$	$7.546 \cdot 10^{-7}$	$7.482 \cdot 10^{-9}$	$7.476 \cdot 10^{-11}$
$\sin(3x)$	$2.526 \cdot 10^{-4}$	$7.702 \cdot 10^{-7}$	$1.093 \cdot 10^{-9}$	$1.125 \cdot 10^{-10}$
x^3	$6.130 \cdot 10^{-4}$	$5.410 \cdot 10^{-6}$	$5.338 \cdot 10^{-8}$	$5.331 \cdot 10^{-10}$

Таблица 2

Фактические погрешности приближения тригонометрическими сплайнами

f	$h = 0.1$	$h = 0.01$	$h = 0.001$	$h = 0.0001$
$1/(1+25x^2)$	$1.380 \cdot 10^{-2}$	$4.706 \cdot 10^{-6}$	$4.648 \cdot 10^{-9}$	$1.869 \cdot 10^{-11}$
$\sin(x)$	$8.123 \cdot 10^{-5}$	$7.546 \cdot 10^{-7}$	$7.482 \cdot 10^{-9}$	$7.476 \cdot 10^{-11}$
$\sin(3x)$	$2.519 \cdot 10^{-4}$	$7.703 \cdot 10^{-7}$	$1.093 \cdot 10^{-9}$	$1.125 \cdot 10^{-10}$
x^3	$6.185 \cdot 10^{-4}$	$5.411 \cdot 10^{-6}$	$5.338 \cdot 10^{-8}$	$5.331 \cdot 10^{-10}$

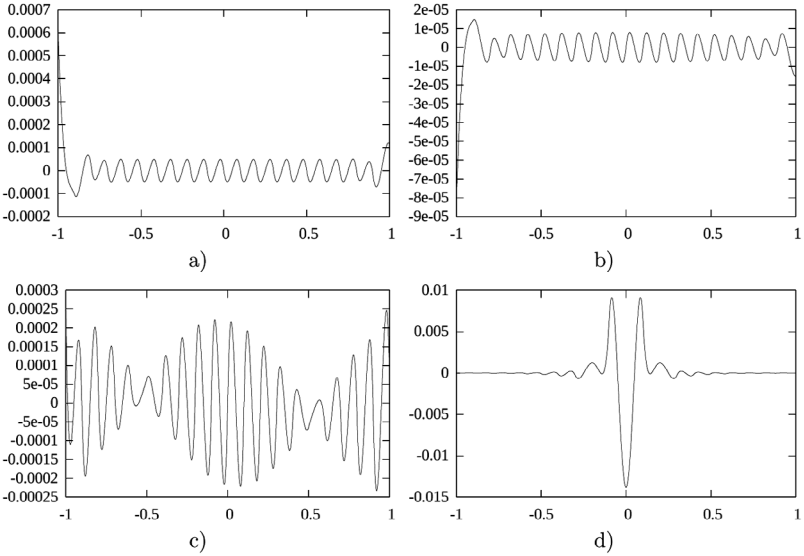


Рис. 1. Фактические погрешности аппроксимации полиномиальными сплайнами при $h=0.1$: a) $f(x)=x^3$, b) $f(x)=\sin(x)$, c) $f(x)=\sin(3x)$, d) $f(x)=\dfrac{1}{1+25x^2}$.

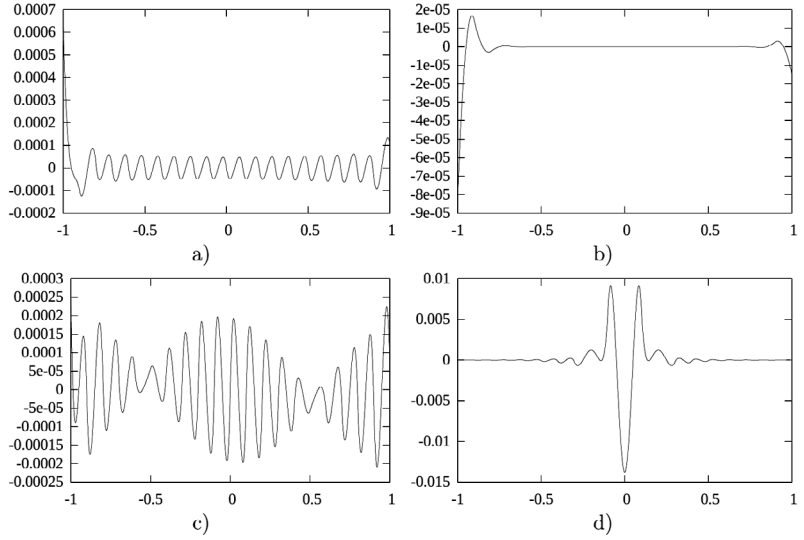


Рис. 2. Фактические погрешности аппроксимации тригонометрическими сплайнами при $h=0.1$: a) $f(x)=x^3$, b) $f(x)=\sin(x)$, c) $f(x)=\sin(3x)$, d) $f(x)=\dfrac{1}{1+25x^2}$.

Литература

1. *Завьялов Ю. С., Квасов Б. И., Мирошниченко В. Л.* Методы сплайн-функций. М., 1980. 352 с.
2. *Бурова И. Г., Евдокимова Т. О.* О гладких тригонометрических сплайнах второго порядка // Вестн. С.-Петерб. ун-та. Сер. 1. 2004. Вып. 3 (17). С. 11–16.

ОБ АППРОКСИМАЦИИ РАЗРЫВНЫМИ ИНТЕГРО-ДИФФЕРЕНЦИАЛЬНЫМИ СПЛАЙНАМИ ТРЕТЬЕГО ПОРЯДКА

И. Г. Бурова, Т. О. Евдокимова

Аннотация. Рассматривается построение разрывных полиномиальных и тригонометрических интегро-дифференциальных сплайнов, а также построение на их основе непрерывных аппроксимаций.

Интегро-дифференциальные полиномиальные сплайны представлены в работе [1], неполиномиальные интегро-дифференциальных сплайны изучались в [2], непрерывные и непрерывно дифференцируемые полиномиальные и тригонометриеские базисные сплайны, использующие значения производных, и основанные на них приближения рассмотрены в работах [3, 4].

В данной работе рассматриваются построение разрывных полиномиальных и тригонометрических интегро-дифференциальных базисных сплайнов, построение на их основе непрерывных аппроксимаций; приведены результаты численных экспериментов и оценки погрешностей.

1. Построение базисных сплайнов

Рассмотрим промежуток $[a, b]$, где a и b — вещественные числа. Возьмём натуральное число $n \geq 2$ и построим равномерную сетку узлов $\{x_j\}$ с шагом $h = \dfrac{(b-a)}{n}$:

$$X : a = x_0 < \ldots < x_{j-1} < x_j < x_{j+1} < \ldots < x_n = b.$$

Пусть $u \in C^3[a, b]$, и заданы значения $\int_{x_k}^{x_{k+1}} u(t)\,dt$, $k = 0, \ldots, n-1$. Рассмотрим на каждом (x_k, x_{k+1}) приближение для $u(x)$ в виде

$$\tilde{u}_k(x) = \left(\int_{x_{k-1}}^{x_k} u(t)\,dt\right)\omega_k^{<-1>}(x) + \left(\int_{x_k}^{x_{k+1}} u(t)\,dt\right)\omega_k^{<0>}(x) + \left(\int_{x_{k+1}}^{x_{k+2}} u(t)\,dt\right)\omega_k^{<1>}(x), \qquad (1)$$

где $\omega_k^{<-1>}(x)$, $\omega_k^{<0>}(x)$, $\omega_k^{<1>}(x)$ определяем из условий

$$\tilde{u}_k(x) = u(x) \text{ при } u(x) = \varphi_i(x),\ i = 1, 2, 3.$$

Предполагаем, что $\varphi_i(x)$, $i = 1, 2, 3$ — чебышевская система на $[x_0, x_n]$, $\varphi_i \in C^3[x_0, x_n]$.

На промежутке (x_k, x_{k+1}) базисные сплайны $\omega_k^{<s>}(x)$, $s = -1, 0, 1$, находим из системы уравнений

$$\left(\int_{x_{k-1}}^{x_k} \varphi_i(t)\,dt\right)\omega_k^{<-1>}(x) + \left(\int_{x_k}^{x_{k+1}} \varphi_i(t)\,dt\right)\omega_k^{<0>}(x) +$$

$$+\left(\int_{x_k}^{x_{k+1}} \varphi_i(t)\,dt\right)\omega_k^{<1>}(x) = \varphi_i(x), \quad i = 1, 2, 3. \tag{2}$$

А. Рассмотрим вначале случай $\varphi_i = x^{i-1}$, $i = 1, 2, 3$. В этом случае система уравнений для нахождения базисных сплайнов будет иметь вид:

$$h\omega_k^{<-1>}(x) + h\omega_k^{<0>}(x) + h\omega_k^{<1>}(x) = 1,$$

$$-\frac{1}{2}h^2\omega_k^{<-1>}(x) + \frac{1}{2}h^2\omega_k^{<0>}(x) + \frac{3}{2}h^2\omega_k^{<1>}(x) = x,$$

$$\frac{1}{3}h^3\omega_k^{<-1>}(x) + \frac{1}{3}h^3\omega_k^{<0>}(x) + \frac{7}{3}h^3\omega_k^{<1>}(x) = x^2.$$

Отсюда, переходя к $x = x_k + th$, $t \in (0,1)$, получаем

$$\omega_k^{<-1>}(x_k + th) = \frac{1}{6h}(2 - 6t + 3t^2), \tag{3}$$

$$\omega_k^{<0>}(x_k + th) = -\frac{1}{6h}(-6t + 6t^2 - 5), \tag{4}$$

$$\omega_k^{<1>}(x_k + th) = \frac{1}{2h}t^2 - \frac{1}{6h}. \tag{5}$$

Пусть $\|f\| = \|f\|_{X(a,b)} = \max\limits_i \sup\limits_{x\in(x_i, x_{i+1})} |f(x)|$, $\|f\|_{(x_i, x_{i+1})} = \sup\limits_{x\in(x_i, x_{i+1})} |f(x)|$.

Теорема 1. *Пусть функция* $u \in C^3[a, b]$. *Для аппроксимации функции* $u(x)$, $x \in (x_k, x_{k+1})$ *сплайнами (1), (3)–(5) справедлива оценка*

$$\left|\tilde{u}_k^{pol}(x) - u(x)\right| \le K_k h^3 \|u'''\|_{X(x_{k-1}, x_{k+2})}, \quad x \in (x_k, x_{k+1}),$$

$K_k = 0{,}44$.

Введем функцию $\widetilde{U}^{pol}(x)$, $x \in (a, b)$, связанную с $\tilde{u}_k^{pol}(x)$ соотношением $\widetilde{U}^{pol}(x) = \tilde{u}_k^{pol}(x)$, $x \in (x_k, x_{k+1})$.

Следствие. *Для погрешности аппроксимации полиномиальными сплайнами (1), (3)–(5) выполняется соотношение*

$$\left\| \widetilde{U}^{pol} - u \right\|_{(a,\,b)} \le Kh^3 \,\| u''' \|_{(a,b)},\ K = 0,44.$$

В. Рассмотрим случай $\varphi_1 = 1,\ \varphi_2 = \sin(x),\ \varphi_2 = \cos(x).$ Пользуясь системой (2) и переходя к $x = x_k + th,\ t \in (0,1),$ получаем

$$\widetilde{\omega}_k^{<-1>}(x_k + th) = \frac{\sin(h) - h\cos(th - h)}{2h\sin(h)(1-\cos(h))} = \frac{2 + 3t^2 - 6t}{6h} + O(h), \tag{6}$$

$$\widetilde{\omega}_k^{<0>}(x_k + th) = \frac{h\cos(th - h) - \sin(2h) + h\cos(th)}{2h\sin(h)(1-\cos(h))} = \frac{5 + 6t - 6t^2}{6h} + O(h), \tag{7}$$

$$\widetilde{\omega}_k^{<1>}(x_k + th) = \frac{\sin(h) - h\cos(th)}{2h\sin(h)(1-\cos(h))} = \frac{3t^2 - 1}{6h} + O(h). \tag{8}$$

Будем рассматривать на каждом промежутке $(x_k,\, x_{k+1})$ приближение для $u(x)$ в виде

$$\tilde{u}_k^{trig}(x) = \left(\int_{x_{k-1}}^{x_k} u(t)\,dt \right) \widetilde{\omega}_k^{<-1>}(x) + \left(\int_{x_k}^{x_{k+1}} u(t)\,dt \right) \widetilde{\omega}_k^{<0>}(x) +$$

$$+ \left(\int_{x_{k+1}}^{x_{k+2}} u(t)\,dt \right) \widetilde{\omega}_k^{<1>}(x), \tag{9}$$

где $\widetilde{\omega}_k^{<-1>}(x)$, $\widetilde{\omega}_k^{<0>}(x)$, $\widetilde{\omega}_k^{<1>}(x)$ определяются соотношениями (6)–(8).

Теорема 2. *Для погрешности аппроксимации тригонометрическими сплайнами (9), (6)–(8) выполняется соотношение*

$$\left| \tilde{u}_k^{trig}(x) - u(x) \right| \le K_k h^3 \,\| u' + u''' \|_{(x_k, x_{k+1})},$$

где $x \in (x_k,\, x_{k+1})$, $K_k = 1/8$.

Введем функцию $\widetilde{U}^{trig}(x)$, $x \in (a,\, b),$ связанную с тригонометрическим сплайном $\tilde{u}_k^{trig}(x)$ соотношением $\widetilde{U}^{trig}(x) = \tilde{u}_k^{trig}(x)$, $x \in (x_k,\, x_{k+1}).$

Следствие. *Для погрешности аппроксимации тригонометрическими сплайнами (9), (6)–(8) выполняется соотношение*

$$\left\| \widetilde{U}^{trig} - u \right\|_{(a,b)} \le Kh^3 \,\| u''' \|_{(a,b)},\quad K = 1/8.$$

2. Построение непрерывных аппроксимаций. Полиномиальный случай

Пусть C_k, $k=1, \ldots, n-1$ — вещественные числа, $h=\text{const}$. На (x_k, x_{k+1}) строим приближение для $u(x)$ в виде

$$\tilde{\tilde{u}}_k^{pol}(x) = \left(\int_{x_{k-1}}^{x_k} u(t)\,dt\right)\omega_k^{<-1>}(x) + \left(\int_{x_k}^{x_{k+1}} u(t)\,dt\right)\omega_k^{<0>}(x) + C_{k+1}\,\omega_k^{<1>}(x), \quad (10)$$

а на (x_{k-1}, x_k) строим приближение для $u(x)$ в виде

$$\tilde{\tilde{u}}_{k-1}^{pol}(x) = \left(\int_{x_{k-2}}^{x_{k-1}} u(t)\,dt\right)\omega_{k-1}^{<-1>}(x) + \left(\int_{x_{k-1}}^{x_k} u(t)\,dt\right)\omega_{k-1}^{<0>}(x) + C_k\,\omega_{k-1}^{<1>}(x).$$

Из условия $\tilde{\tilde{u}}_{k-1}^{pol}(x_k-) = \tilde{\tilde{u}}_k^{pol}(x_k+)$ получаем

$$C_{k+1} + 2C_k = f_k, \quad k=1, \ldots, n-1,$$

$$f_k = \left(\int_{x_{k-2}}^{x_{k-1}} u(t)\,dt\right) - 3\left(\int_{x_{k-1}}^{x_k} u(t)\,dt\right) + 5\left(\int_{x_k}^{x_{k+1}} u(t)\,dt\right);$$

$$2C_n = f_n,$$

$$f_n = \left(\int_{x_{n-2}}^{x_{n-1}} u(t)\,dt\right) - 3\left(\int_{x_{n-1}}^{x_n} u(t)\,dt\right) + 5\left(\int_{x_n}^{x_{n+1}} u(t)\,dt\right) - \left(\int_{x_{n+1}}^{x_{n+2}} u(t)\,dt\right).$$

Теперь нетрудно получить

$$C_n = f_n/2, \quad C_{n-i-1} = -(C_{n-i} - f_{n-i-1})/2, \quad i=1, \ldots, n-1.$$

3. Построение непрерывных аппроксимаций. Тригонометрический случай

Построение осуществляется аналогичным полиномиальному случаю образом с использованием тригонометрических сплайнов $\tilde{\omega}_k^{<s>}(x)$, $s=-1, 0, 1$. На (x_k, x_{k+1}) приближение для $u(x)$ строится в виде

$$\tilde{\tilde{u}}_k^{trig}(x) = \left(\int_{x_{k-1}}^{x_k} u(t)\,dt\right)\tilde{\omega}_k^{<-1>}(x) + \left(\int_{x_k}^{x_{k+1}} u(t)\,dt\right)\tilde{\omega}_k^{<0>}(x) + C_{k+1}\tilde{\omega}_k^{<1>}(x). \quad (11)$$

Таким образом, здесь из условия $\tilde{\tilde{u}}_{k-1}^{trig}(x_k-) = \tilde{\tilde{u}}_k^{trig}(x_k+)$ получаем

$$d_{k+1}C_{k+1} + d_kC_k = 6hf_k, \quad k=1, \ldots, n-1,$$

$$d_k = \frac{6h(\sin h - h\cos h)}{2h\sin h(1-\cos h)},\quad d_{k+1} = -\frac{6h(\sin h - h)}{2h\sin h(1-\cos h)},$$

$$f_k = A(h)\left(\int_{x_{k-2}}^{x_{k-1}} u(t)\,dt\right) + B(h)\left(\int_{x_{k-1}}^{x_k} u(t)\,dt\right) + C(h)\left(\int_{x_k}^{x_{k+1}} u(t)\,dt\right);$$

$$d_n C_n = 6hf_n,\quad d_n = \frac{6h(\sin h - h\cos h)}{2h\sin h(1-\cos h)},$$

$$f_n = A(h)\left(\int_{x_{k-2}}^{x_{k-1}} u(t)\,dt\right) + B(h)\left(\int_{x_{k-1}}^{x_k} u(t)\,dt\right) +$$

$$+C(h)\left(\int_{x_k}^{x_{k+1}} u(t)\,dt\right) - A(h)\left(\int_{x_{k+1}}^{x_{k+2}} u(t)\,dt\right),$$

а

$$A(h) = \frac{\sin(h) - h}{2h\sin h(\cos h - 1)},\quad B(h) = \frac{2h\cos h - \sin h - \sin 2h + h}{2h\sin h(\cos h - 1)},$$

$$C(h) = \frac{\sin 2h - h\cos h - h}{2h\sin h(\cos h - 1)}.$$

Лемма. *1) В полиномиальном случае*

$$\left|C_k - \int_{x_k}^{x_{k+1}} u(t)\,dt\right| \le K_1 h^4 \,\|u'''\|_{(x_{k-1},\, x_{k+2})},\quad K_1 = 3/2.$$

2) В тригонометрическом случае

$$\left|C_k - \int_{x_k}^{x_{k+1}} u(t)\,dt\right| \le K_2 h^4 \,\|u' + u'''\|_{(x_{k-1},\, x_{k+2})},\quad K_2 = 9/4.$$

Теорема 3. *Пусть функция* $u \in C^3[a, b]$. *1) Для аппроксимации функции* $u(x),\ x \in (x_k, x_{k+1})$ *сплайнами (10), (3)–(5) справедлива оценка*

$$\left|\overset{\approx}{u}{}_k^{pol}(x) - u(x)\right| \le Kh^3 \,\|u'''\|_{X(x_{k-1},\, x_{k+2})},\quad K = 1{,}94.$$

2) Для аппроксимации функции $u(x),\ x \in (x_k, x_{k+1})$ тригонометрическими сплайнами (11), (6)–(8) справедлива оценка

$$\left|\overset{\approx}{u}{}_k^{trig}(x) - u(x)\right| \le Kh^3 \,\|u' + u'''\|_{X(x_{k-1},\, x_{k+2})},\quad K = 2{,}375.$$

Литература

1. *Киреев В. И., Пантелеев А. В*. Численные методы в примерах и задачах. М., 2008. 480 с.
2. *Бурова И. Г*. Аппроксимация вещественными и комплексными минимальными сплайнами. СПб.: Изд-во СПбГУ, 2013. 142 с.
3. *Бурова И. Г., Евдокимова Т. О*. О гладких тригонометрических сплайнах второго порядка // Вестн. С.-Петерб. ун-та. Сер. 1. 2004. Вып. 3 (17). С. 11–16.
4. *Бурова И. Г., Евдокимова Т. О*. О гладких тригонометрических сплайнах третьего порядка // Вестн. С.-Петерб. ун-та. Сер. 1. 2004. Вып. 4. С. 12–21.

О СЖАТИИ И ВОССТАНОВЛЕНИИ ИЗОБРАЖЕНИЯ С ПОМОЩЬЮ ПАРАМЕТРИЧЕСКИ ЗАДАННЫХ СПЛАЙНОВ

О. В. Безрукавая

E-mail: olga.bezrukavaya@gmail.com

Аннотация. Рассматривается построение изображения с помощью параметрически заданных кусочно-линейных и квадратичных сплайнов. Приведен пример работы программы на основе рукописных букв алфавита. Рассматривается пример построения изображения с помощью кусочно-линейных сплайнов в реальном времени на языке C++.

1. Постановка задачи

Проблемы сжатия, восстановления, хранения, передачи и кодировки текстовой информации имеет большое практическое значение. В данной работе предлагается рассмотреть один из вариантов сжатия и восстановления изображения с помощью обработки его параметрически заданными сплайнами. Будем рассматривать изображения букв русского алфавита и арабские цифры. Для построения будут использоваться кусочно-линейные и квадратичные сплайны.

Для построения изображения используем кусочно-линейные и квадратичные сплайны, заданные параметрически.

Буква, изображение которой хотим построить, задается минимальным набором опорных точек. Опорными точками называем такие точки, которые используются для восстановления исходного изображения с заданной точностью. Так как задание изображения замкнутой кривой на плоскости предоставляет некоторые трудности, то будем использовать параметрическое представление функции.

2. Построение минимальных параметрически заданных сплайнов

Имеем функции $x = x(t)$ и $y = y(t)$. Пусть $x, y \in C_2[a, b]$, $\{tj\}$ — упорядоченная сетка узлов, $x(tj)$ — значение функции x в узле tj, $y(tj)$ — значение функции y в узле tj. Построим приближение для наших функций в виде

$$X(t) = x(t_j)\, w_j(t) + x(t_j + 1)\, w_j + 1\,(t), \quad t \in [t_j, t_j + _i],$$
$$Y(t) = y(t_j)\, w_j(\mathrm{t}) + y(t_j + 1)\, w_j + 1\,(t), \quad t \in [t_j, t_j + _i],$$

где

$$w_j(t) = \frac{t - t_{j+1}}{t_j - t_{j+1}}, \quad t \in [t_j, t_{j+1}],$$

$$w_{j+1}(t) = \frac{t - t_j}{t_{j+1} - t_j}, \quad t \in [t_j, t_{j+1}].$$

Теперь, пользуясь этими формулами, можно построить наше изображение с помощью кусочно-линейного сплайна, заданного параметрически.

Имеем функции $x = x(t)$ и $y = y(t)$. Пусть $x, y \in C_3[a, b]$, $x(t_j)$ — значение функции x в узле t_j, $y(t_j)$ — значение функции y в узле t_j.

Тогда используем на промежутке $[t_j, t_{j+1}), j = 1, ..., n - 1$, следующую формулу:

$$X(t) = ACx(t_j) - ABx(t_{j+1}) + BCx(t_{j+2}),$$
$$Y(t) = ACy(t_j) - ABy(t_{j+1}) + BCy(t_{j+2}),$$

где

$$A = \frac{t - t_{j+2}}{t_j - t_{j+1}}, \quad B = \frac{t - t_j}{t_{j+1} - t_{j+2}}, \quad C = \frac{t - t_{j+1}}{t_j - t_{j+2}}.$$

Используя приведенные выше формулы, можем построить изображение с помощью квадратичного сплайна, заданного параметрически.

3. Построение изображения буквы алфавита с помощью параметрически заданных сплайнов

Рассмотрим в качестве нашего изображения прописную букву «Б». Построим ее с помощью кусочно-линейного и квадратичного сплайна. Для начала возьмем минимальное число точек для линейного сплайна, по которым может быть восстановлено изображение, и построим эту букву с помощью кусочно-линейного и квадратичного сплайна.

На рис. 1 изображена буква «Б», построенная с помощью линейных, на рис. 2 — с помощью квадратичных сплайнов, а также изображены опорные точки, по которым строилась буква «Б».

а) *б)*

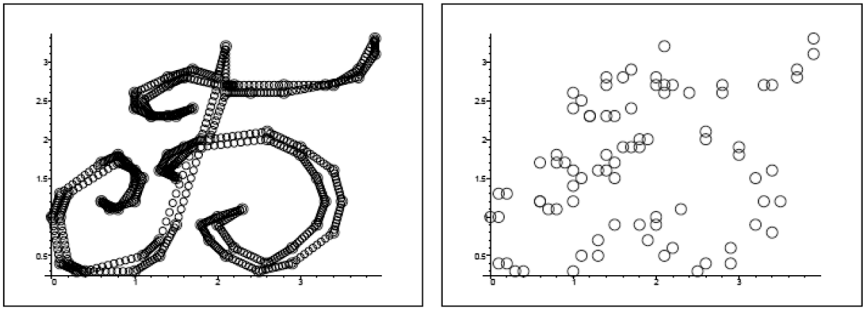


Рис. 1. Изображение: *а*) — буквы «Б» линейным сплайном; *б*) — опорных точек буквы «Б»

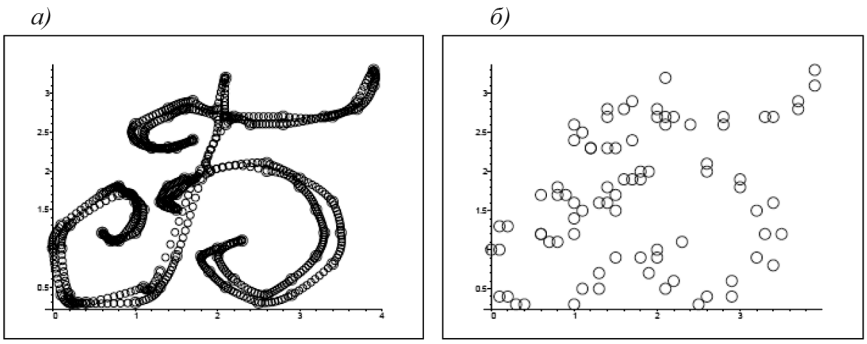


Рис. 2. Изображение: *а*) — буквы «Б» квадратичным сплайном; *б*) — опорных точек буквы «Б»

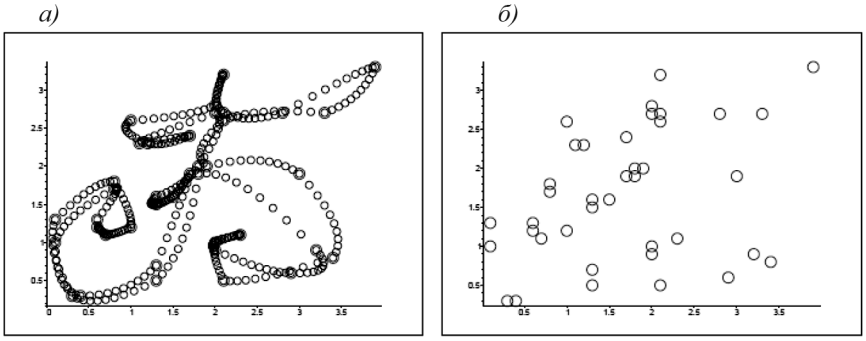


Рис. 3. Изображение: *а*) — буквы «Б»; *б*) — опорных точек буквы «Б»

Минимальное количество точек для линейного сплайна оказалось равным 83. Теперь будем уменьшать количество точек таким образом, чтобы при построении изображения с помощью квадратичного сплайна по этим точкам наша буква все еще была узнаваема.

На рис. 3 изображена буква «Б», построенная с помощью квадратичного сплайна, а также изображены опорные точки, по которым строилась буква «Б».

Итак, можем сделать вывод, что минимальное количество точек для прописной буквы «Б», построенной с помощью квадратичного сплайна, равно 41.

4. Построение изображения в реальном масштабе времени с помощью кусочно-линейного сплайна

Задача программы состоит в том, чтобы с помощью линейного сплайна по опорным точкам, заданным с курсора мыши, построить простое изображение и координаты данных точек записать в файл.

Данная программа написана на языке C++, и представляет собой окно, в котором будет производиться само построение изображения, справа от окна располагаются кнопка «Отменить все» и два поля, где будут записаны координаты точек последнего клика мыши.

Кратко о работе программы:

1. С помощью курсора ставим точки. Далее эти точки соединяются линией. Красным цветом обозначаются точки, которые построены с помощью нажатия на кнопку мыши, а черным те, что появляются при построении их с помощью кусочно-линейного сплайна по формулам, приведенным в пункте 2.
2. При нажатии на курсор мыши координаты данной точки показываются в полях справа и записываются во внешний файл.
3. Очистка экрана происходит следующим образом либо при нажатии на кнопку «Отменить все», либо при закрытии окна программы.

Литература

1. *Бурова И. Г., Демьянович Ю. К*. Теория минимальных сплайнов. СПб.: Изд-во СПбГУ, 1999. 357 с.
2. *Бурова И. Г*. Применение математического пакета Maple 7 в курсе методов вычисления. 2004.

Методы хранения, поиска и анализа информации

Новиков
Борис Асенович

д.ф.-м.н.
профессор кафедры информатики СПбГУ

SUPPORTING ADDITIONAL TREE DATA STRUCTURES IN GIST

P. V. Fedotovsky
SPbSU Student
E-mail: pavel.v.fedotovsky@gmail.com

K. E. Cherednik
SPbSU Student
E-mail: kirill.cherednik@math.spbu.ru

G. A. Chernishev
SPbSU Assistant Professor
E-mail: chernishev@gmail.com

Abstract. In this paper we deal with the Generalized Search Tree (GiST), an index data structure supporting an extensible set of queries and data types. We show how the strictness of its interface impedes its usage with some of the recently developed tree structures. We also introduce a modification to GiST that solves aforementioned problem for some of these structures without interfering with GiST concurrency control mechanism. Next, some guidelines for adjusting existing GiST-based trees are presented. Finally we illustrate our technique via a Revised R*-tree (RR*-tree) implementation.

Introduction

An index is an important part of a modern database management system. It could greatly speed-up query evaluation. The focus of this paper is multidimensional indexes — those that operate with multidimensional data.

There are many approaches to multidimensional indexing. Approaches may be roughly classified to tree-based and hash-based indexes. One of the most popular among tree-based structures is R-tree and its variants [1].

R-tree is a height-balanced tree similar to B-tree. It defines a hierarchy of hyper-rectangles, which partition the data space. The search is a tree traversal starting from the root node and proceeding to leaf nodes. During the traversal only necessary (the ones, which may contain requested data) hyper-rectangles are examined. Despite being a relatively old data structure — this tree was proposed by Antonin Guttman in 1984 [3], it is still being actively developed. Many industrial-level DBMS implement R-tree as an access method. One can name Oracle, PostgreSQL, SQLite and several others.

One of the characteristics crucial to R-tree performance is minimization of both coverage and overlap area for its nodes. Several variants of R-tree were pro-

posed to address this issue. For example, R*-tree demonstrates particularly good behavior in this regard. It uses a combination of a revised node split algorithm and the concept of forced reinsertion at node overflow. This is based on the fact that R-tree structure is highly susceptible to the order in which entries are inserted.

GiST is a data structure and API that can be used to build a variety of height-balanced search trees. It also provides a concurrency control mechanism. This is a widespread approach that is used, for example, in PostgreSQL. It can be used to easily implement a range of well-known search trees, including B+-tree, R-tree and many others. Nevertheless it has its own limitations. For example it cannot be readily used to implement R*-tree (because of its reliance on forced reinsertion technique) [2].

Recent research activity brought up more R-tree based structures. In this paper we are interested in RR*-tree, because it is one of the top performing variants according to reference [4]. Moreover, this tree poses several common (to other tree variants) characteristics, which prevent its integration into GiST. It is a redesign of R*-tree that takes into account reinsertion concept drawbacks and overcomes it by reengineering subtree choice and node split algorithms. This behavior leads to the question whether it is possible to implement RR*-tree using GiST or not. The goal of this paper is to investigate this opportunity. To the best of our knowledge this is the first attempt to embed RR*-tree into GiST.

GiST: Basic Notions

GiST is a popular data structure for implementing search trees. Essentially it is a balanced tree of variable fan-out between kM and M, $(2/M) \leq k \leq M$, with the exception of the root node, which may have fan-out between 2 and M [2]. The constant k is termed the minimum fill factor of the tree [2]. Here k and M are defined by a developer. An example is presented on the figure 1.

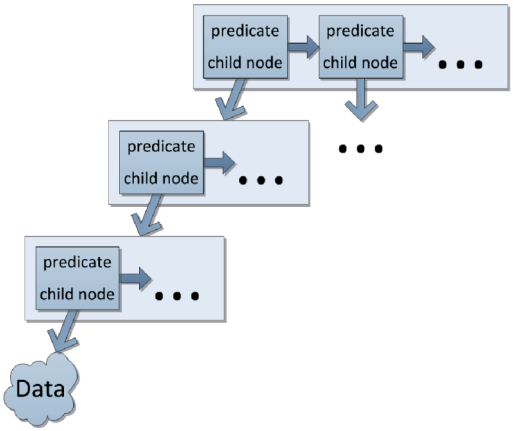


Fig. 1. A GiST structure: an overview

In order to use it, developer has to implement `Predicate` structure and several methods. The `Predicate` structure represents a predicate, which is used for the tree construction. This predicate describes data spatially contained in a given entry. It may be a multidimensional rectangle or a multidimensional sphere.

GiST supports insert, search, update and delete queries. Their implementation is predefined and uses six user-implemented methods [2]:

- `Consistent(E, q)`: given an entry `E = (p, ptr)`, and a query predicate `q`, returns false if $p \wedge q$ can be guaranteed unsatisfiable, and true otherwise;
- `Union(P)`: given a set `P` of entries, returns some predicate `r` that holds for all tuples stored below these entries;
- `Compress(E)`: given an entry `E = (p, ptr)` returns an entry `(p', ptr)` where `p'` is a compressed representation of `p`;
- `Decompress(E)`: given a compressed representation `E = (p', ptr)`, where `p' = Compress(p)`, returns an entry `(r, ptr)` such that $p \rightarrow r$;
- `Penalty(E1, E2)`: given two entries $E_1 = (p_1, ptr_1), E_2 = (p_2, ptr_2)$, returns a domain-specific penalty for inserting E_2 into the subtree rooted at E_1;
- `PickSplit(P)`: given a set `P` of `M+1` entries, splits `P` into two sets of entries P_1, P_2, each of size at least `kM`.

This means that for a tree structure to be implemented using GiST, it must implement these six methods. Examples of such tree structures are: B+-tree, R-tree.

GiST Drawbacks

GiST requirements for tree structure are rather strict. There are several promising data structures that almost satisfy GiST requirements, but due to their specificity, some of the operations cannot be expressed via GiST methods.

Let's consider RR*-tree — one of the latest variants of R-tree. The main differences are `ChooseSubtree` and `Split` methods. Its `ChooseSubtree` method cannot be expressed in terms of `Penalty` method because it should be able to access information about all the processed entries. It uses adaptive strategy (volume-based or perimeter-based), which depends on the state of all entries that are contained in a node.

GiST Modification

To solve the aforementioned problems we suggest the following GiST contract modifications:

- Replace `Penalty()` method, with:
  `Entry FindOptimalEntry(Entry[] entries, entry_to_insert);`
- Add new methods:
  `UpdateNodeInfo(Entry[] entries, NodeInfo info);`

- Change `PickSplit` method signature from:
  `PickSplit(Entry[] entries)`
  to:
  `PickSplit(Entry[] entries, NodeInfo info).`

Let's take a closer look at the proposed modifications. `FindOptimalEntry` method should be invoked during new entry insertion. GiST searches for optimal node to store the entry in order to minimize further search-related costs. `FindOptimalEntry` method should find entry optimal for expansion among those that are stored in a node. This modification provides means for straighforward implementation of the subtree selection algorithm. For backward compatibility it may be implemented as a method that minimizes `Penalty`.

`UpdateNodeInfo` method is used to store additional information about node's entries that can be later used in `PickSplit` algorithm, giving it more flexibility. User has to define `NodeInfo` structure (in the same manner the one has to do for the `Predicate` structure).

To support these GiST changes the following modifications have to be done:

- `ChooseSubtree` should invoke user-implemented `FindOptimalEntry` method instead of searching for optimal node for insertion by means of penalty minimization
- Split should be modified as follows:
  - `PickSplit` should be invoked with additional argument NodeInfo, that is determined by node to be split (for example as its field)
  - `UpdateNodeInfo` should be invoked for both result sets and corresponding nodes

This GiST extension doesn't interfere with concurrency control mechanism described in paper [5] as proposed modifications only affect parts of code, that operate with data that was already locked.

Example: Revised R*-tree

RR*-tree differs from R-tree in two main aspects: Split algorithm and `ChooseSubtree` algorithm (in GiST notation). Moreover Split algorithm requires information about node's original center. So modified GiST method will look like this:

- `FindOptimalEntry` — "CSRevised" algorithm from RR*-tree paper [4];
- `PickSplit` — "Split" algorithm from RR*-tree paper [4].

Only small modifications of these methods are required to adapt them to GiST interface.

Also structure `NodeInfo` may be defined as follows:

```
struct NodeInfo {
    Predicate bounding_box_original_center;
};
```

And `UpdateNodeInfo` method will store minimal bounding box's center in `bounding_box_original_center`.

This algorithm for RR*-tree was implemented in our multidimensional transactional index prototype. We plan its experimental evaluation as a basis for future work.

Conclusion

The paper describes modification to popular data structure GiST that is used to implement tree-like structures. One of the advantages of implementing new structures using GiST interface is the opportunity to employ GiST concurrency control mechanism. However a lot of structures could not be implemented this way. A RR*-tree may serve as an example. Proposed modification solves this problem for a range of structures. For example, Hilbert R-tree [6] — another popular R-tree variant can also be implemented in GiST using this technique.

References

1. *A. N. Papadopoulos, A. Corral, A. Nanopoulos,* and *Y. Theodoridis*. R-Tree (and Family). In L. LIU and M. T. OZSU, editors, Encyclopedia of Database Systems, pages 2453–2459. Springer US, 2009. 10.1007/978-0-387-39940-9 300.
2. *Joseph M. Hellerstein, Jeffrey F. Naughton,* and *Avi Pfeffer*. 1995. Generalized Search Trees for Database Systems. In Proceedings of the 21th International Conference on Very Large Data Bases (VLDB'95), Umeshwar Dayal, Peter M. D. Gray, and Shojiro Nishio (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 562–573.
3. *Antonin Guttman*. 1984. R-trees: a dynamic index structure for spatial searching. In Proceedings of the 1984 ACM SIGMOD international conference on Management of data (SIGMOD'84). ACM, New York, NY, USA, 47–57. DOI=10.1145/602259.602266 http://doi.acm.org/10.1145/602259.602266
4. *Norbert Beckmann* and *Bernhard Seeger*. 2009. A revised r*-tree in comparison with related index structures. InProceedings of the 2009 ACM SIGMOD International Conference on Management of data (SIGMOD'09), Carsten Binnig and Benoit Dageville (Eds.). ACM, New York, NY, USA, 799–812. DOI=10.1145/1559845.1559929 http://doi.acm.org/10.1145/1559845.1559929
5. *Marcel Kornacker, C. Mohan,* and *Joseph M. Hellerstein*. 1997. Concurrency and recovery in generalized search trees. In Proceedings of the 1997 ACM SIGMOD international conference on Management of data (SIGMOD'97), Joan M. Peckman, Sudha Ram, and Michael Franklin (Eds.). ACM, New York, NY, USA, 62–72.
6. *Ibrahim Kamel* and *Christos Faloutsos*. 1994. Hilbert R-tree: An Improved R-tree using Fractals. In Proceedings of the 20th International Conference on Very Large Data Bases (VLDB'94), Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 500–509.

CREATING SENTIMENT DICTIONARIES AND ANALYSIS OF GOODS REVIEWS IN RUSSIAN

Alina Dubatovka

Student of Dept. of Analytical Information Systems,
Saint-Petersburg State University

E-mail: alina.dubatovka@gmail.com,

Boris Novikov

Professor of Dept. of Analytical Information Systems,
Saint-Petersburg State University

E-mail: b.novikov@spbu.ru

Abstract. Nowadays a lot of people use different online stores to choose and buy products. These services often can provide data for opinion extraction and expose some structure and rating that simplify this analysis. In this paper, the method for creating sentiment dictionaries and analysis of goods reviews using information and structure of reviews from online stores such as Yandex.Market is described.

Introduction

It is not a secret that a great part of human activity is based on opinion of society or just some particular group of people. Opinions influence on decisions of both individuals and entire organizations and companies. At the same time, the development of Web2.0 provides much more opportunities for users' opinion surveys, since users share their experiences, views and opinions with their friends and others.

More and more people are using the internet as a "deliberative body", browsing reviews about various products and services. It raises such problems as searching for goods and services with certain positive characteristics or creation a kind of "adviser" based on analysis of customers' reviews. Of course, companies are also interested in the analysis of users' comments about their products. Furthermore, the sentiment analysis finds its application in contextual advertising, allowing to advertise item where it is praised.

There are so many online stores that contain reviews of various products as well as customer reviews, which are of interest to manufacturers, such as Yandex. Market or Amazon. Moreover, these sites tend to contain a lot of information that are easy to analyze because of the particular structure. Thus many sites along with review on particular product contain a user rating that readily allows approximately calculate product rating, but, unfortunately, does not allow to understand anything about the reasons for one or another opinion, because a review usually isn't

an emotionally homogeneous and contains both positive and negative statements about different aspects and characteristics of the product. However, it provides additional useful information for the opinion mining about the objects and aspects, for example, sections “advantages” and “disadvantages” that have definite polarity.

In this regard, the opinion mining has become important for researchers (because of a huge number of the natural language processing subtasks), but also for different companies.

Related work

The description of related work is split into two parts: discussion of different sentiment analysis approaches and examination of aspect extraction methods.

Sentiment analysis

The sentiment extraction problem can be considered as the classification problem: it is necessary to include a document either in positive or in negative class. In [6] the algorithm consists of two steps of classification: the first one finds neutral phrases and the second one determines polarity of non-neutral phrases. In addition, specialized resources often contain product ratings that allow to use this data as a training sets for supervised learning algorithms.

Lexicon-based approach determines polarity of the text, using polarity of its words [1]. These methods use sentiment dictionaries, containing polarity of each word expressed numerically. Usually this value is equal to 1, if the word is positive, and -1 in the opposite case.

In this work, we use lexicon-based algorithm described in [5]. Both supervised learning and the lexicon-based algorithms are effective to sentiment analysis. A considerable drawback of supervised learning is the need for training data belonging to the respective domain where the analyzer will be used. At the same time the lexicon-based methods need precompiled sentiment dictionary, which should also be adequate for a given object domain, and therefore can not be universal for different goods.

Aspect extraction

The most commonly used approach to the problem of aspect extraction is the methods of unsupervised learning and statistical methods. The advantage of these methods is the fact that they do not require training data, which is not available for public use.

Problem of aspect extraction can be solved by bootstrapping [8, 9]: having some initial set of aspects it is possible to iteratively extract similar in some metric words or phrases, thus expanding the original set.

Statistical methods consider the problem of aspects extraction as a problem of extracting terms that are the most commonly used in the document. One of them is

C-value method is described in [2], which combines statistical and linguistic approaches to term recognition. A frequency based method also can be find in [3, 4].

However, for aspects extraction both unsupervised learning and statistical methods are equally effective and do not require any preprocessed data so can be applied directly to the existing reviews.

Methodology

The key idea of the method is to use sections "advantages" and "disadvantages" of reviews on Yandex.market as the most emotionally rich texts for producing sentiment dictionaries for particular object or object domain. Obtained dictionaries then are used to determine the polarity of comments, reviews and surveys simultaneously complementing dictionary. Moreover, usually these sections of reviews contain the most part of evaluations of product characteristics, so they can be used as data for preliminary extracting the aspects that will be elaborated during the following processing of reviews.

Therefore, the method includes the following stages:

- definition polarity of reviews about the product and its aspects;
- creating sentiment dictionary for this object domain based on sections "advantages" and "disadvantages" of reviews on Yandex.Market;
- extension of sentiment dictionary during reviews processing;
- extraction objects and aspects of the goods based on sections "advantages" and "disadvantages" of reviews on Yandex.Market;
- elaboration extracted aspects in the analysis of comments.

To implement the algorithm for determining the contextual polarity, we applied a lexicon-based method described in [5]. This method determines the polarity of the author's opinion about some aspect based on the polarity of opinions about each of the occurrences of this aspect in the review.

Creating sentiment dictionary

To compile a sentiment dictionary the method described in [5] was slightly modified. All occurrences of adjectives and nouns are not yet included in the dictionary were considered as candidates for inclusion instead of object aspects in original algorithm. For each such candidate the sentiment score and the number of occurrences of the word in the document were calculated. Thereafter, common words with pronounced polarity (absolute values of the sentiment score and the frequency should exceed a certain thresholds) are added to the dictionary.

The initial dictionary was obtained as a partial translation of dictionary described in [6] into Russian. It consists of five hundred words and is subsequently expanded and used for the opinion mining algorithm.

Extension of the dictionary

For improvements and extensions of the dictionary, the algorithm determining candidates for inclusion in the dictionary was applied not only to “advantages” and “disadvantages”, but also to the comments during their processing.

So besides determining the sentiment of aspects the algorithm also continues to gather statistics of the words (namely, the sentiment score and the number of occurrences). If one of the candidates exceeds the thresholds of the score and frequency, it should be included in the dictionary. Thus during the processing of comments it is also possible to improve the dictionary in addition to the opinion extraction.

Experiments

As long as the method for aspect extraction has not been developed yet, it is only possible to evaluate the efficiency of phrase-level and document-level parts of proposed method. There are two ways for it:

1. Assuming all sentences from section “advantages” are positive and all phrases from section “disadvantages” are negative, the testing of phrase-level sentiment analysis can be realized;
2. Using sections “rating” and “comment” of product reviews and considering the comments with 4 or 5 rating as positive whereas comments with 1 or 2 rating as negative, the data for document-level evaluation is obtained.

As data for training, analysis and testing, various reviews Yandex.Market including sections on “advantages”, “disadvantages”, “comment” and “rating” are used. During a processing, each review is split into sentences using the nltk-library [http://www.nltk.org/], after that using the mystem [https://company.yandex.ru/technologies/mystem/] POS-tagging of each phrase is performed: extracted tokens are normalized and part of speech for each token is defined. Next, all of prepositions, conjunctions and numerals are thrown out from sentences, because having no polarity they only “clog up” the distance function. After all, the algorithm described above is applied to the resulting text.

For creating sentiment dictionary 1682 notes from section “advantages” and 1752 comments from “disadvantages” were processed. In result, the seed dictionary, including 561 words, was expanded to 1287 words. This new dictionary was used for further sentiment analysis.

In the first experiment 3246 texts from section “advantages” consisting of 8504 sentences was used to test positive phrase-level sentiment analysis. Also negative phrase-level sentiment analysis was evaluated with help of 2997 notes from section “disadvantages” containing 6469 phrases.

The results of these tests are presented in the following table:

	Recall	Precision	F-measure
Positive	0.56	0.62	0.59
Negative	0.17	0.51	0.26

In addition, it was discovered that proposed algorithm marks a lot of sentences (about 30% in both positive and negative tests) like neutral and it could be suggested that the threshold in this algorithm should be decreased. Another possible reason for such results is that in fact there are a lot of neutral or even positive sentences in the section "disadvantages". For example, "there are no disadvantages" or just "no". At the same time, "advantages" can contain mixed or negative phrases like "The only not quite pleasant feature of the processor is heating". Therefore, it is necessary to clean data for future experiments to achieve more precise evaluation. One of the ideas is to take "advantages" with 4 or 5 ratings and "disadvantages" with 1 or 2 ratings. Some other causes and ways for improvements are described in [7].

Conclusion and future work

In this paper, the opinion mining technique for product reviews is presented. Proposed method uses existing lexicon-based algorithm for finding sentiment of opinion about aspect and creates needed sentiment dictionary from initial set of sentiment words, using structure of reviews on online shops and processing of sections "advantages" and "disadvantages" to extract the most emotional words. To find polarity of phrase the basic algorithm is applied with assumption that every noun in the phrase is an aspect, so sentiment of phrase can be calculated as the average of polarity of its nouns. And the sentiment of review in whole is obtained as an average of all phrases sentiments.

In the future work the experiments to evaluate and tune proposed method will be provided to find parameters and thresholds that improve its accuracy. The method for aspect extraction using the sections "advantages" and "disadvantages" and refinement of a list of extracted aspects during processing of comments will be elaborated. So it will be possible to realize the algorithm of automatic determination of aspects and creating sentiment dictionary for them and following processing of goods reviews from given object domain.

References

1. *Xiaowen Ding, Bing Liu,* and *Lei Zhang*. Entity discovery and assignment for opinion mining applications. In John F. Elder IV, Franoise Fogelman-Souli, Peter A. Flach, and Mohammed Javeed Zaki, editors, KDD, pages 1125–1134. ACM, 2009.
2. *K. Frantzi, S. Ananiadou,* and *H. Mima*. Automatic Recognition of Multi-Word Terms: the Cvalue/NC-value Method. International Journal on Digital Libraries, 2000(3): 115–130, 2000.

3. *Minqing Hu* and *Bing Liu*. Mining and summarizing customer reviews. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, KDD'04, pages 168–177, New York, NY, USA, 2004. ACM.
4. Lun-Wei Ku, Yu-Ting Liang, and Hsin-Hsi Chen. Opinion extraction, summarization and tracking in news and blog corpora. In AAAI Symposium on Computational Approaches to Analysing Weblogs (AAAI-CAAW), pages 100–107, 2006.
5. *Nicolas Nicolov, Franco Salvetti,* and *Steliana Ivanova*. Sentiment analysis: Does coreference matter. In In AISB 2008 Convention Communication, Interaction and Social Intelligence, 2008.
6. *Theresa Wilson, Janyce Wiebe,* and *Paul Hoffmann*. Recognizing contextual polarity in phraselevel sentiment analysis. In Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP), pages 347–354, 2005.
7. *Florian Wogenstein, Johannes Drescher, Dirk Reinel, Sven Rill,* and *Jrg Scheidt*. Evaluation of an algorithm for aspect-based opinion mining using a lexicon-based approach. In Erik Cambria, Bing Liu, Yongzheng Zhang, and Yunqing Xia, editors, WISDOM, page 5. ACM, 2013.
8. *David Yarowsky*. Unsupervised word sense disambiguation rivaling supervised methods. In ACL-95, pages 189–196, Cambridge, MA, 1995. ACL.
9. *Jingbo Zhu, Huizhen Wang, Muhua Zhu, Benjamin K. Tsou,* and *Matthew Y. Ma*. Aspect-based opinion polling from customer reviews. T. Affective Computing, 2(1):37–49, 2011.

TEXT DETECTION IN NATURAL SCENES WITH MULTILINGUAL TEXT

Mikhail Zarechensky
E-mail: zarechenskij@gmail.com

Scientific supervisor:
Ph. D. ***Natalia Vassilieva***
Department of Analytical Information Systems,
St. Petersburg State University

Abstract. Detecting text in natural scenes is an important prerequisite for further text recognition and other image analysis tasks. Most of text detection methods for scene images usually use a priori knowledge of language to detect text. As a rule such algorithms are evaluated on datasets which contain scenes only with text in English. This paper discusses known text detection algorithms and investigates them for invariance to the language.

1. Introduction

Recent advances in digital technology allow to take pictures from a large number of mobile devices. As a result, the number of photos taken by users is increasing every day. At the same time, we often have no annotations for images except those made by the device. Text in images provides important information about semantics of the image. Annotated images can be used in various applications, such as content-based image retrieval, automatic navigation, automatic translation. It is often the case that a language of a text in an image is not known in advance, or a single image contains text areas with text in different languages. How to effectively detect and recognize text in scene images is an actual research question. Text detection is an important prerequisite for further text recognition. In this paper we explore the problem of text detection.

In this paper, we discuss several known text detection algorithms and investigate them for invariance to a language. Quality of the text detection algorithm greatly depends on the shooting conditions and noise on the image, but in this paper we focus on the problem of language invariance of the algorithms in good conditions. First, we distinguish the main common steps of these algorithms. Second, we provide a theoretical estimation of language invariance for every step of the algorithms. Third, we perform experiments with two algorithms on different datasets to confirm theoretical result.

2. Related work

In order to recognize text in an image, it first has to be robustly detected. Unlike text detection for document images, text detection for scenes is still a challenging task due to the large variety of text appearance in images. Text in scenes can have different variations of the font style, size, distortion; it can have different contrast due to different lighting conditions. The whole image can also vary greatly. We should take into account low resolution, low contrast, heterogeneous background. Such variety gives rise to various approaches to text detection.

Existing methods for scene text detection can be broadly categorized into three groups: texture-based methods, region-based methods and hybrid methods.

The basic difference between these techniques of text detection methods is described in [12]. In this paper we consider only connected components based methods. According to the results of the competition at the ICDAR 2013, this approach proved itself to be more effective comparing to others. We picked methods proposed by Yin et al.[11], Gomez et al. [5] and Chen et al. [2] for further consideration. These algorithms have good results on the ICDAR datasets and use different approaches to detect text.

3. Overview of text detection algorithms

The algorithm proposed by Yin et al. [11] was presented at ICDAR 2013 and got the first place in "Multi-script Robust Reading Competition in ICDAR 2013" [6]. It uses an approach based on the MSER algorithm to find character glyphs.

A disadvantage of the MSER [7] is that it detects a lot of false positives — regions that do not contain characters. To solve this problem, the algorithm by Yin et al. [11] additionally performs parent-children elimination for the MSER tree. It improves accuracy of finding character regions. The main idea is to eliminate regions with very small or very big aspect ratio.

The next step of the algorithm is to group characters in order to construct text candidates. Character candidates are clustered into text candidates by the single-link clustering algorithm. The parameters depend on the following features: spatial distance, a differences between width and height, top and bottom alignments, color difference, stroke width difference.

At the final step text candidates are labeled by a classifier as text or non-text areas. The following features are used to train the classifier: smoothness, the average stroke width, stroke width variation, height, width, and aspect ratio.

As described in [12] for algorithms presented in [2], [5], text features that are using to filter out non-characters are similar to features discussed above.

4. Analysis of the main steps of the algorithms

In this section we discuss the main steps of the methods and provide a theoretical estimation of their language invariance.

Character candidates extraction

As presented above, for the region decomposition it is common to use the MSER algorithm. The MSER algorithm depends only on the intensity of the image. Since the text in the image tends to have equal intensity, at least in each symbol, the result of the algorithm is independent of language.

We can conclude that the MSER algorithm is equally applicable for region decomposition as for images, containing only one language and for images with multilingual text. As the Canny edge detector not depend on a language, it follows that the modified MSER, proposed by Chen et al. is also invariance to a language

Filtering of regions

Let us review every feature used for region filtering.

- **Aspect ratio**

Most letters of English language have aspect ratio being close to 1, so this feature might be useful to filter out false character candidates. To cope with elongated letters such as “i” or “l”, a threshold should be small enough. On the one hand, this feature can be used for many languages because even if a letter has a very small aspect ratio and is filtered out, the absence of this letter will not affect the grouping of whole word at the grouping stage.

On the other hand, when an entire word is not split into the characters it might cause difficulties in text detection. There are languages in which every word is continuously connected. For instance, Hindi, in which all words are linked by continuous line. In this case, rational use of this feature is difficult, because words might be very long.

Thus this feature has limitations and may not be used for all languages.

- **Region height**

Irrespective of language, height of the characters in one word are always about the same. Therefore, this type of filter is invariant to the language.

- **Number of holes**

Number of holes in the English characters and in the hieroglyphs might be different. Therefore, this feature requires an additional configuration for different languages.

- **Stroke width**

This feature is very important as it is shown in the work Epshtein et al.[4]. However, the proposed implementation has a limitation for the elements that have non-parallel edges. This feature of the implementation is essential for such languages as Arabic. Also the style of writing in Arabic language tends to have more variation in the stroke width, thus to achieve maximum efficiency, this feature must be configured for different languages separately.

5. Empirical analysis

To confirm the theoretical estimations provided in the previous section we will perform a series of experiments for the two methods described in the section 2.

Description of the experiments

For the experiments was selected the algorithm proposed by Yin et al.[11].

For evaluation we used a similar approach and the same quality measures as in the evaluation scheme of ICDAR 2013 competition. The following quality measures are used: precision, recall and f-measure.

Description of test data

In the first group of tests we run the selected algorithm on the following datasets: MSRA-TD500, ICDAR 2011, ICDAR 2013. ICDAR 2011 dataset contains images with text in English only. MSRA-TD500 dataset contains images with text in English and Chinese. And ICDAR 2013 dataset contains images with multilingual text including Indo-Aryan languages and Chinese writing.

Also we created a new synthetic dataset which contains images with multilingual text. Dataset split by languages with prevailing text features described above: stroke width, alignment, aspect ratio, etc. Each group corresponding to the one language contains 100 images. Specific image contains only one word and simple background to focus only on the text features.

Example from synthetic dataset
with text in Urdu

Analysis of the experimental results

The results of every test are presented in the following table.

Dataset	Recall	Precision	F-measure
ICDAR 2011	0.68	0.86	0.76
ICDAR 2013	0.42	0.64	0.51
MSRA-TD500	0.21	0.52	0.30

The results of the algorithm on the synthetic dataset are presented in the following table.

Features	Language	Recall	Precise	F-measure
	Chinese	0.64	0.72	0.68
	English	0.57	0.73	0.64
Stroke width	Sinhala	0.48	0.69	0.57
Alignment	Maldivian	0.33	0.69	0.44
Alignment, Stroke width	Urdu	0.23	0.63	0.33
Aspect ratio	Bengali	0.10	0.82	0.18
Aspect ratio	Marathi	0.06	0.91	0.11

Based on the experimental results one may see that the difference between the result on the ICDAR 2011 dataset which contains only images with English text, and all others, is quite big.

From the experimental results on the synthetic dataset we can conclude that quality of some text detection algorithms could strongly depend on such text features as aspect ratio, alignment and stroke width.

6. Conclusion

In this work the following results were obtained.

1. The most efficient text detection algorithms are discussed.
2. The main common steps of text detection algorithms are identified.
3. Every step of text detection algorithms is analyzed analytically for invariance to a language.
4. Evaluated a series of experiments.

During the work it was obtained that the existing set of features may strongly depend on a language. By changing settings of the rules that are used in the algorithms you can improve the text detection results on some pre-defined languages.

As a possible continuation to this work it is planned to implement a complete algorithm that solves the problem of text detection irrespective of a language. The analysis presented in this paper helps to identify problem pieces of the existing algorithms. The created dataset and the experimental results will allow to evaluate better the result of this new algorithm.

References

1. *J. Canny*. A computational approach to edge detection. IEEE Trans. Pattern Analysis and Machine Intelligence, 8:679–698, 1986.
2. *H. Chen, S. S. Tsai, G. Schroth, David M. Chen, R. Grzeszczuk,* and *B. Girod*. Robust text detection in natural images with edge- enhanced maximally stable extremal regions. IEEE International Conference on Image Processing, Sep 2011.

3. *A. Desolneux, L. Moisan,* and *J.-M. Morel*. A grouping principle and four applications. IEEE Trans. PAMI, 2003.
4. *B. Epshtein, E. Ofek,* and *Y. Wexler*. Detecting text in natural scenes with stroke width transform. In CVPR, 2010.
5. *L. Gomez* and *D. Karatzas*. Multi-script text extraction from natural scenes. ICDAR, 2013.
6. *D. Kumar, M. N. Anil Prasad,* and *A. G. Ramakrishnan*. Multi-script robust reading competition in icdar 2013. In ACM Proc. International Workshop on Multilingual OCR, (MOCR 2013), 2013.
7. *J. Matas, O. Chum, M. Urban,* and *T. Pajdl*. Robust wide baseline stereo from maximally stable extremal regions. In British Machine Vision Conference, volume 1, pages 384–393, 2002.
8. *L. Neumann* and *J. Matas*. Real-time scene text localization and recognition. IEEE Conference on Computer Vision and Pattern Recognition, 2012.
9. *N. Otsu*. A threshold selection method from gray-level histograms. IEEE Transactions on Systems, Man and Cybernetics, 9(1):62–66, 1979.
10. *I. Zeki Yalniz, Douglas Gray,* and *R. Manhmatha*. Adaptive exploration of text regions in natural scene images. ICDAR, 2013.
11. *X.-C. Yin, X. Yin,* and *K. Huang*. Robust text detection in natural scene images. CoRR, abs/1301.2628, 2013.
12. *M. Zarechensky*. Text detection in natural scenes with multilingual text. SYRCoDIS, 2014.

УПРАВЛЕНИЕ ДАННЫМИ ОБ ОТХОДАХ ПОТРЕБЛЕНИЯ НА ПРИМЕРЕ ГОРОДА ПЕТРОЗАВОДСКА

О. Ю. Янюк

студентка кафедры Информатики СПбГУ

E-mail: olya.ianiuk@gmail.com

Введение

На интеллектуальный анализ данных (АД) о городах, позволяющий неоспоримо лучше понять процессы сложных городских систем, возлагают большие надежды, связанные с повышением качества жизни населения. Однако, особенности данных подобной природы обуславливают определённые трудности в управлении ими, и, как следствие, подходы АД успешно применяются лишь ограниченным кругом специалистов [1, 2].

Разработка и/или совершенствование, а также применение инструментов управления городскими данными в настоящих прикладных задачах ведёт не только к повышению эффективности конкретных урбанистических решений, но и к более широкому принятию многообещающих методов исследования городского пространства.

Общие задачи и решение

Основными задачами представляемого проекта являются:

1) повышение темпов внедрения раздельного сбора отходов потребления (ОП) в городе Петрозаводске;
2) поддержка спроса на вторсырьё и сортировку твёрдых бытовых отходов (ТБО);
3) вовлечение большего количества жителей в процесс сортировки.

Общее предлагаемое решение заключается в разработке и создании информационных сред для:

1) анализа компонентов системы управления ОП; 2) управления логистикой вывоза ТБО;
3) повышения общей информированности и вовлечения жителей.

Краткая характеристка проблемы

Перед Петрозаводском стоит проблема невозможности дальнейшего эксплуатирования городской свалки. В то же время, в городе существуют

относительно благоприятные условия для внедрения раздельного сбора ТБО, в сочетании с переработкой вторсырья — подхода являющегося наиболее эффективным в решении возникшей проблемы [3, 4].

Во-первых, это неудовлетворённость спроса на вторсырьё со стороны переработчиков[1]. Во-вторых, существование запроса на сортировку со стороны населения [5]. И, наконец, определённая активность администрации [7, 8].

Тем не менее, судя по некоторым опубликованным результатам [9] и общим рекомендациям специалистов [10], внедрение раздельного сбора ТБО проходит недостаточно эффективно: разобщённость и неравномерность организации является и неудобством для жителей и управляющей компании, и значительной издержкой частного бизнеса, занимающегося переработкой.

Аналитическая система

О функциональности системы

На сегодняшний день для Петрозаводска, как впрочем и для большинства городов в мире, характерен недостаток автоматически собирающихся данных о состоянии городского пространства. Более того, как упоминалось выше, аналитический аппарат управления данными о городах несовершенен [1, 2]. В этих условиях, одной из самых сложных задач исследования системы управления ОП в Петрозаводске является процесс feature engineering, начиная с задачи определения признаков (features) и источников, из которых они могут быть извлечены, заканчивая интеграцией данных и получением чистой статистики. Поэтому, для обеспечения указанного процесса, должна быть развёрнута гибкая рабочая среда, позволяющая эффективно удовлетворять разнообразные аналитические потребности.

В Таблице 1 приведены существующие источники данных, интеграция которых, быть может в любых комбинациях, потенциально имеет значение для исследования компонентов системы ОП. С учётом требований, обозначенных выше, максимальную свободу аналитикам могло бы обеспечить использование отдельных модулей существующих инструментов работы с данными, в большей степени подходящих на тех или иных этапах, в сочетании с самостоятельно созданными вспомогательными скриптами. Примером некоторого предела данного подхода служит разработка специфичного программного комплекса под конкретные нужды исследования [11, 12].

---

[1] Компания «ЭкоЛинт», производящая тротуарную плитку из пластика, бумаги и строительного мусора, готова не только бесплатно устанавливать специализированные контейнеры, но и безвозмездно вывозить ТБО [6].

Таблица 1

Релевантные источники данных

Название источника	Потенциальное применение
Карты OpenStreetMap	Мониторинг привязанных объектов (специализированных контейнеров)
Сервис Yandex.Пробки	Расчёт времени прибытия (к специализированному контейнеру); управление логистикой
Микроблог Twitter, Сервисы Instagram, 4square	Сообщения с геолокационной привязкой+семантический анализ — мнение о городе, городских событиях и пространствах и т. п., в том числе контекст ТБО; изображения мест в городе с той же целью
Социальные сети (Vkontakte, Facebook и т.п.)	Распространение информации о внедрении сортировке ТБО

С другой стороны, академические наработки, представленные в [13] эффективно управляют «big data» — к которым, в частности, относятся данные о городах — в силу особенностей разрабатываемого языка запросов. Во-первых, аналитические сценарии прозрачно и гибко формулируются на языке благодаря его декларативности. Во-вторых, представление операций запросов на промежуточном языке, в терминах обобщённых алгебраических операций, предоставляет широкие оптимизационные возможности во время выполнения запросов интегрирующих разнородные источники. Однако, несмотря на важные достоинства, инструменты [13] требуют доработки, в соответствии с подходом к решению прикладных проблем, что и является одной из подзадач текущего исследовательского проекта.

О структуре языка запросов

В основе концепции представленной в [13] лежит понятие *q*-set, которое представляет собой модель <*q*, *B*, *S*>, где *q* — запрос, *B* — базовое множество оцениваемых на предмет «схожести» объектов, *S* — множество оценок элементов из *B*, соответствующих запросу *q*. Сходство — мера близости в определённом пространстве, вычисление которой зависит от алгебраических операций использующихся в запросе. Таким образом, результирующий *q*-set содержит и запрос, и результат его исполнения, выраженный в оценках. Так как для алгебраических операции входными и выходными параметрами являются *q*-sets — запросы имеет вложенную структуру.

Пример запроса

Для того, чтобы проиллюстрировать работу информационной системы, строящейся для анализа компонентов системы управления ОП, обратимся к стандартной потребности исследователей — использованию данных о физических объектах городского пространства. Приведём пример первоначальной обработки карт OpenStreetMap (OSM), выполняемой в системе.

В XML формате карт OSM (.osm) существует три типа примитивов: nodes, ways, relations. Пусть объекты в базовых множествах B первоначальных q-sets совпадают с одним из примитивов. Запрос q для первоначального q-set соответствует процессу извлечения данных из источника — в нашем случае сервера openstreetmap.org — поэтому, будем реплицировать одну и ту же географическую область (г. Петрозаводск), как локальную копию данных карты. Затем, извлекать необходимые данные из копии и материализовать результат, присваивая объектам оценки, совпадающие с порядковым номером исполненного запроса на извлечение или, что то же, порядковым номером локальной копии.

В карты OSM постоянно вносятся изменения, таким образом, при использовании данных на локальной машине, появляется задача поддержки информации в актуальном состоянии. Стандартный способ решения — обновление с помощью логов изменений (changesets), ведущихся на стороне сервера. Однако, при таком подходе необходимо дополнительно заботиться о сохранении предыдущих состояний карты, которые бывают необходимы в аналитических целях, например, для выявления сути внесённых изменений или характеристик osm-сообщества, а также в практических — автоматического определения дубликатов с заданной точностью. Для того, чтобы одновременно и обновлять данные, и собирать необходимую статистику изменений, в используемом языке запросов достаточно выбрать лишь подходящие операции.

Пусть объект *кажется существует*, если он извлекается из более чем одной среди трёх последних локальных копий. Поскольку содержимое копий соотносится с первоначальными q-sets, применим к последним простую операцию нормирования (norm), переводящую значение оценок в интервал [0,1], а затем операцию superunion, для вычисления *доверия к существованию объекта* на карте:

```
filter(scores> 0.9,
    superunion(norm(3_primary_qsets)))
```

Формальное определение superuninon, а также других спецификаций операций объединения для алгебры языка запросов приведено в [14]; оценка, которая соответствует выбранной операции учитывает эффект «хора» — объекты аргументов с близкими оценками имеют большее влияние на результи-

рующую оценку, а также эффект «крика» — объект с сильно отличающейся оценкой имеет большее влияние на результирующую оценку. Первое свойство естественно для определения *существования* объекта, данного выше, второе свойство вытекает из выражения *большего доверия* объектам последних копий.

Операция фильтрации (filter) пересчитывает оценки в q-set, в соответствии с заданным значением, которое, в приведённом запросе, выбрано эмпирически.

Таблица 2

Изменение оценок объектов в результирующем *q*-set

ID	#151	#152	#153 = #545	#546	Changes in scores
280679. . .		2 => 1	3 => 2=> 1	...	NA=>0.9=>0.96=>0.972
2808005201				2	NA=>0.9=>...

В Таблице 2 показано изменение оценок в результирующем q-set, соответствующее исполнениям запроса в течение 60 часов. В результате данного эксперимента, было обработано 1400 локальных копий, содержащих около 182211 примитивов nodes в каждой; зафиксировано добавление описаний точек с идентификаторами 2806791140, 2806793108, 2806793109 в копии #152 и 2808005201 в копии #546, что полностью совпало с логами изменений.

О проектировании

Можно выделить следующие архитектурные особенности строящейся аналитической системы:

1) иерархическая структура запросов ослабляет необходимость логического разделения бизнес-логики и данных;
2) необязательна формализация ETL процессов, так как источники данных определяются при формулировке аналитической потребности;
3) секционирование, резервное копирование или удаление данных также может осуществляться при помощи декларативных запросов.

Заключение

В докладе рассмотрен пример работы информационной системы, строящейся в рамках исследовательского проекта для решения урбанистических задач. Предложенный подход позволяет определить уникальность города подмножеством источников данных, использующихся для анализа, а значит его применение универсально, в силу незначительности природы источников для декларативного языка запросов в основе системы.

Литература

1. *M. Batty*. Big data; big issues. Geographical Magazine of the Royal Geographical Society, 75, 2013.
2. *M. Batty*. Big data, smart cities and city planning. Dialogues in Human Geography, 3(3):274–279, 2013.
3. *И. Ступин*. От свалок к хвостохранилищам. Эксперт, 038:88–91, 2012.
4. *M. Rodionov, T. Nakata*. Design of an optimal waste utilization system: a Case Study in St. Petersburg, Russia. Sustainability, 3(9):1486–1509, 2011.
5. Vkontakte. Переработка: Шаг второй. [WWW], 2013. http://vk.com/pererabotka_ptz, (visited 2014-03-31).
6. *В. Бирюкова*. Карельские волонтёры провели экологическую акцию в Международный День Земли. [WWW], 2013. http://bit.ly/1pzfhnh, (visited 2013-05-03).
7. Northern Dimension Environmental Partnership. Petrozavodsk solid waste management. [WWW], 2009. http://bit.ly/1cr8E1B, (visited 2014-01-20).
8. *Ю. Савицкая*. ОТХОДЫ: повышение информированности в сфере сортировки и переработки. Отчёт, Администрация Города Петрозаводска, 2011–2012.
9. *Norden*. W. A. S. T. E. Waste Awareness: Sorting, Treatment, Education. [WWW], 2011. http://bit.ly/ZZE7Bs, (visited 2013-05-03).
10. *И. Бабанин*. Организация селективного сбора: рекомендации. Твёрдые бытовые отходы, 09:10–17, 2009.
11. *J. S. Evans-Cowley, G. Griffin*. Micro-participation: The role of microblogging in planning. Available at SSRN, February 2011.
12. *C. Roth, SM. Kang, M. Batty, M. Barthlemy*. Structure of urban movements: Polycentric activity and entangled hierarchical flows. PLoS ONE, 6(1):e15923, 2011.
13. *B. Novikov, N. Vassilieva, A. Yarygina*. Querying big data. In CompSysTech, 1–10, 2012.
14. *A. Yarygina, B. Novikov, N. Vassilieva*. Processing complex similarity queries: A systematic approach. In ADBIS (2), 212–221, 2011.

Кибернетика и робототехника

Фрадков
Александр Львович

д.т.н.
профессор кафедры теоретической кибернетики СПбГУ
заведующий лабораторией
«Управление сложными системами» ИПМаш РАН

Лучин
Роман Михайлович

ст.преподаватель кафедры теоретической кибернетики СПбГУ
научный сотрудник лаборатории теоретической кибернетики

УПРАВЛЕНИЕ ПРОМЫШЛЕННЫМ ГИДРАВЛИЧЕСКИМ ПРИВОДОМ НА ПРИМЕРЕ РОБОТОТЕХНИЧЕСКОГО КРАНА-МАНИПУЛЯТОРА, ПРИМЕНЯЕМОГО В ЛЕСОЗАГОТОВИТЕЛЬНОЙ ПРОМЫШЛЕННОСТИ

А. А. Лосенков

магистрант, кафедры Систем управления и информатики Университета ИТМО

E-mail: alosenkov@gmail.com

С. В. Арановский

к.т.н., с.н.с. кафедры Систем управления и информатики

E-mail: s.aranovskiy@gmail.com

Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Аннотация. Предложена модель гидравлической системы, учитывающая такие особенности промышленных гидроприводов, как нелинейность золотникового гидрораспределителя и наличие компенсатора давления. Проанализировано положение динамического равновесия модели. Предложено использование обращения статической нелинейности при построении закона управления и обосновано введение прямой связи по скорости. Представлены результаты экспериментальных исследований на примере робототехнического крана-манипулятора.

Введение

Гидравлические машины, системы гидропривода и устройства на их основе широко применяются во многих отраслях промышленности. Среди причин широкого распространения гидропривода в перечисленных областях можно отметить его высокую удельную мощность. В настоящей работе рассматривается гидравлический привод робототехнического крана-манипулятора [1], применяемый в тяжелых гидрофицированных машинах лесозаготовительной промышленности. Экспериментальные исследования проводились на лабораторном прототипе такого крана[1].

Модели гидравлических систем осложнены нелинейностями уравнений потоков жидкости и динамики давлений. По этой причине большинство тяжелых промышленных гидрофицированных машин являются неавтоматизированными. В настоящее время автоматизация таких машин представляет

[1] Авторы благодарят департамент прикладной физики и электроники Университета Умео, г. Умео, Швеция, за предоставленное для экспериментов оборудование.

большой интерес для промышленности: использование систем автоматического управления позволило бы увеличить эффективность работы машины, при этом снизив затраты топлива и нагрузку на оператора.

В большинстве известных статей по управлению гидравлическим приводом используются линейные модели гидрораспределителей и не учитывается наличие дополнительных элементов, присущих промышленным гидросистемам. Для точного управления промышленными гидравлическими системами требуются более сложные, нелинейные модели, а так же учет таких устройств, как компенсатор давления. Кроме того, в большинстве работ полагается, что измерению доступен широкий спектр сигналов, включая измерения давления, положения штока гидроцилиндра, его скорость и даже ускорение, что далеко не всегда возможно на практике.

Модель гидропривода

Схема гидравлического привода, состоящего из гидроцилиндра и золотникового гидрораспределителя, представлена на рис. 1.

Гидроцилиндр состоит из двух камер, A и B. Давления в камерах обозначены как P_a и P_b, а площади гидроцилиндра со стороны соответствующих камер как A_a, A_b. Положение штока гидроцилиндра обозначено как $0 \le x \le x_{\max}$; q_a, q_b — потоки рабочей жидкости между гидрораспределителем и соответствующими камерами. Гидрораспределитель имеет четыре гидролинии: A, B, S и T. К гидролинии S подведен гидравлический насос, которому соответствует давление P_S, гидролиния T подключена к резервуару, которому соответствует давление P_T. Гидролинии A и B гидрораспределителя соединены с соответствующими камерами гидроцилиндра. Управление золотником в гидрораспределителе производится при помощи электромагнита изменением входного сигнала управления u.

Уравнения динамики давлений имеют вид (на основе [2; 3]):

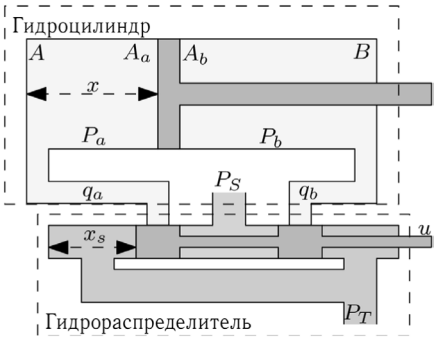


Рис. 1. Схема рассматриваемого гидравлического привода

$$\begin{aligned} \dot{P}_a &= \frac{\beta}{V_{a0} + x A_a}\left(q_a - \dot{x} A_a\right), \\ \dot{P}_b &= \frac{\beta}{V_{b0} - x A_b}\left(q_b + \dot{x} A_b\right), \end{aligned} \qquad (1)$$

где $V_{a0} > 0$ и $V_{b0} > A_b \cdot x_{\max}$ объемы камер A и B соответственно при нулевом смещении поршня $x = 0$, β — модуль объемного сжатия рабочей жидкости, q_a и q_b — потоки жидкостей к камерам A и B соответственно, которые определяются на основе [3–5]:

$$q_a = \begin{cases} c_a S(u)\sqrt{|P_S - P_a|}\,\mathrm{sign}(P_S - P_a) & \text{при } u \ge 0, \\ c_a S(u)\sqrt{|P_a - P_T|}\,\mathrm{sign}(P_a - P_T) & \text{при } u < 0, \end{cases} \quad (2)$$

$$q_b = \begin{cases} -c_b S(u)\sqrt{|P_b - P_T|}\,\mathrm{sign}(P_b - P_T) & \text{при } u \ge 0, \\ -c_b S(u)\sqrt{|P_S - P_b|}\,\mathrm{sign}(P_S - P_b) & \text{при } u < 0, \end{cases} \quad (3)$$

где $c_a > 0$ и $c_b > 0$ — постоянные коэффициенты, зависящие от свойств рабочей жидкости, геометрии золотника и иных физических параметров, $S(u) \in [-1; 1]$ — нормализованная статическая нелинейная функция, описывающая открытие рабочего окна и включающая в себя нелинейности типа мертвая зона и насыщение, подробно рассмотренная в [6–8]. Для работы с нелинейной функцией $S(u)$ введем статическую инверсию нелинейности, т.е. такую функцию $\psi(v)$ что $S(\psi(v)) = v$. Для рассматриваемого крана функция $S(u)$ имеет вид, представленный на рис. 2.

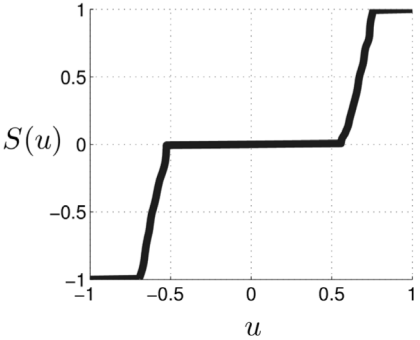


Рис. 2. Вид статической нелинейной функции $S(u)$

Компенсатор давления — это вспомогательное устройство, входящее в состав гидрораспределителя. Как правило, вместе с компенсатором давления в систему входит контур чувствительности к нагрузке. Гидросхемы золотникового гидрораспределителя с компенсатором давления и без него представлены на рис. 3.

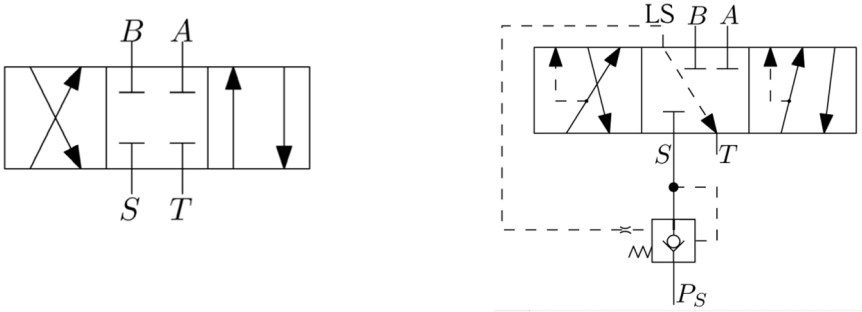


Рис. 3. Гидросхемы гидрораспределителей:
(а) — без компенсатора давления; (б) — с компенсатором давления

Гидролиния *LS* передает сигнал с нагруженного порта (того порта, который подключен к гидролинии насоса *S*) в компенсатор давления. В зависимости от мгновенного значения сигнала нагрузки, компенсатор давления поддерживает падение давления на гидрораспределителе на постоянном уровне ΔP_{PC}, устанавливаемом производителем. С учетом наличия компенсатора давления, уравнения потоков жидкости (2) и (3) принимают вид:

$$q_a = \begin{cases} c_a S(u)\sqrt{\Delta P_{PC}} & \text{при } u \geq 0, \\ c_a S(u)\sqrt{|P_a - P_T|}\,\mathrm{sign}(P_a - P_T) & \text{при } u < 0, \end{cases} \quad (4)$$

$$q_b = \begin{cases} -c_b S(u)\sqrt{|P_b - P_T|}\,\mathrm{sign}(P_b - P_T) & \text{при } u \geq 0, \\ -c_b S(u)\sqrt{\Delta P_{PC}} & \text{при } u < 0. \end{cases} \quad (5)$$

Гидравлическая сила, порождаемая гидроцилиндром, описывается выражением:

$$F_h = P_a A_a - P_b A_b \quad (6)$$

Уравнение движения штока гидроцилиндра имеет вид:

$$m\ddot{x} = F_h - F_{ex}, \quad (7)$$

где m совокупная масса штока и приложенной нагрузки, F_{ex} совокупная внешняя сила, включающая в себя силы трения, гравитации и нагрузки.

Пусть система находится под действием постоянной внешней силы $F_{ex} = \mathrm{const}$ и постоянного сигнала управления $u_0 = \mathrm{const} \neq 0$. Анализ динамического равновесия полученной модели (в настоящем тезисе не приводится) рассматриваемой системы, при действии на нее постоянных внешних сил $F_{ex} = \mathrm{const}$ и постоянного сигнала управления $u_0 = \mathrm{const} \neq 0$, показывает, что равновесная скорость $\dot{x}$ пропорциональна условной площади открытия рабочего окна гидрораспределителя $S(u_0)$ и не зависит от внешней силы F_{ex}. Такое поведение системы отличается от поведения, свойственного гидросистемам с золотниковыми гидрораспределителями без компенсатора давления, где $\dot{x}$ является функцией и от площади открытия $S(u_0)$, и от внешней силы F_{ex}.

Экспериментальные исследования

В ходе экспериментальных исследований рассматривалась задача управления положением гидравлического привода с использованием предложенной в настоящей работе модели. Сформулируем задачу управления как обеспечение слежения угла θ за задающим сигналом θ*, используя только лишь измерения угла θ. Для достижения цели управления используется преобразование заданной угловой координаты θ* в линейную x^*:

$$x^* = f_x(\theta^*), \quad \dot{x}^* = \frac{df_x(\theta)}{d\theta}\dot{\theta}^*, \tag{8}$$

и преобразование измеренного углового положения θ в выдвижение штока гидроцилиндра $x = f_x(\theta)$. Таким образом, рассматриваются два сигнала ошибки:

$$e_x = x^* - x, \quad e_\theta = \theta^* - \theta. \tag{9}$$

Ошибка e_x используется в обратной связи при управлении, а ошибка e_θ для оценки качества работы регулятора. Закон управления имеет следующий вид:

$$u = \begin{cases} 0.95\text{sign}\left(e_x\right) & \text{если} \quad |e_x| > 0.2 \\ \theta \qquad 0 & \text{если} \quad |e_x| < 0.004 \quad \dot{x}^* = \\ \psi\left(v + \dot{x}^*\right) & \text{иначе.} \end{cases} \tag{10}$$

где входящий в третье условие дополнительный управляющий сигнал v формируется следующим образом:

$$v = k_p e_x + k_s \,\text{sign}\,(e_x) \tag{11}$$

Закон управления (11) является релейно-пропорциональным, и известно, что его прямое применение при наличии у объекта мертвой зоны может привести к нежелательным осцилляциям сигнала управления, которые негативно сказываются на качестве работы системы. Для борьбы с этим негативным явлением воспользуемся методиками, описанными в [9; 10]:

- Фильтрацией релейного члена:

$$v(t) = k_p e_x(t) + k_f F\,[\text{sign}\,(e_x(t))], \tag{12}$$

где сигнал $F[y]$ представляет собой сигнал y, пропущенный через устойчивый линейный фильтр $F(s) = 1/(\tau s + 1)$.

- Интегрированием функции $\text{sign}\,(\cdot)$:

$$v(t) = k_p e_x(t) + k_{si} \int_0^t \text{sign}\,(e_x(t))\, dt. \tag{13}$$

Так же рассмотрим пропорционально-интегральный регулятор:

$$v(t) = k_p e_x(t) + k_i \int_0^t e_x(t)\, dt. \tag{14}$$

Экспериментальные исследования проводились на звене лабораторного прототипа крана, применяемого в лесозаготовительной промышленности (см. рис. 4).

Рис. 4. Вид экспериментальной установки

В ходе эксперимента измерялся угол θ, который конструктивно ограничен в пределах $\theta \in [-150°; -20°]$. Результаты экспериментальных исследований представлены на рис. 5.

Как видно из рис. 5, все регуляторы показали схожие результаты: ошибка слежения e_θ не превышает двух градусов, что является очень хорошим точностным показателем для такого класса систем, а среднее значение ошибки находится в районе нуля.

Заключение

Рассмотрена промышленная гидравлическая система на примере робототехнического крана-манипулятора, применяемого в лесозаготовительной промышленности. Подобная система отличается от большинства известных моделей учетом нелинейной зависимости площади открытия рабочего окна гидрораспределителя от величины смещения его золотника, а так же наличием компенсатора давления в гидравлической системе. Предложена модель системы, включающая в себя компенсатор давления. На основе полученной модели и анализа динамического равновесия сделан вывод о возможности введения статической нелинейной кривой и прямой связи по скорости. Проведены экспериментальные исследования в реальном времени на лабора-

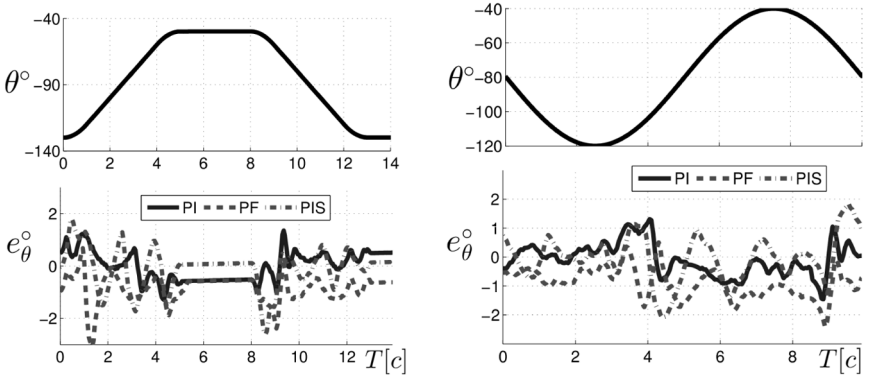


Рис. 5. Результаты экспериментальных исследований при слежении за (а) — S-кривой, (б) — синусоидой с амплитудой $A = 40°$ и частотой 0.1 Гц

торном прототипе робототехнического крана-манипулятора, применяемого в лесозаготовительной промышленности. В ходе экспериментов измерению было доступно лишь положение штока гидроцилиндра, рассмотрены три различных закона управления. Результаты экспериментов показывают, что использование обращения статической нелинейной характеристики гидрораспределителя и введение прямой связи по скорости позволяют обеспечить хорошее качество слежения.

Литература

1. *Ortiz Morales D.* и др. Increasing the Level of Automation in the Forestry Logging Process with Crane Trajectory Planning and Control // Journal of Field Robotics. 2014. Т. 31. № 3. С. 343–363.
2. *Sohl G. A., Bobrow J. E.* Experiments and simulations on the nonlinear control of a hydraulic servosystem // Control Systems Technology, IEEE Transactions on. 1999. Т. 7. № 2. С. 238–247.
3. *Боровин Г. К.* и др. Моделирование гидравлической системы экзоскелетона // Математическое моделирование. 2006. Т. 18. № 10. С. 39–54.
4. *Merritt H. E.* Hydraulic Control Systems. USA: Wiley, John & Sons, Incorporated, 1967. 360 С.
5. *Башта Т. М., Руднев С. С., Некрасов Б. Б.* Гидравлика, гидромашины и гидроприводы: учебник для вузов. 2-е изд., перераб. М.: Машиностроение, 1982. 424 с.
6. *Гамынин Н. С.* Гидравлический привод систем управления. М.: Машиностроение, 1972. 376 с.
7. *Арановский С. В. Фрейдович Л. Б., Никифорова Л. В., Лосенков А. А.* Моделирование и идентификация динамики золотникового гидрораспределителя. Часть I: моделирование // Известия ВУЗов. Приборостроение. 2013. Т. 56. № 4. С. 52–56.
8. *Арановский С. В. Фрейдович Л. Б., Никифорова Л. В., Лосенков А. А.* Моделирование и идентификация динамики золотникового гидрораспределителя. Часть II: Идентификация // Известия ВУЗов. Приборостроение. 2013. Т. 56. № 4. С. 57–60.
9. *Лосенков А. А., Арановский С. В.* Система управления гидроприводом с компенсацией статической нелинейности // Научно-технический вестник информационных технологий, механики и оптики. 2013. № 5 (87). С. 77–81.
10. *Utkin V., Guldner J., Shijun M.* Sliding mode control in electro-mechanical systems. : Taylor & Francis, CRC press, 1999. 325 с.
11. *Shtessel Y.* и др. Sliding mode control and observation. New York: Springer, 2014. 356 pp.

КОМПЕНСАЦИЯ НЕИЗВЕСТНОГО МУЛЬТИСИНУСОИДАЛЬНОГО ВОЗМУЩЕНИЯ ПРЯМЫМ АДАПТИВНЫМ МЕТОДОМ НА ОСНОВЕ ДЕКОМПОЗИЦИИ ВОЗМУЩЕНИЯ

А. А. Лосенков

магистрант, кафедры Систем управления и информатики Университета ИТМО

E-mail: alosenkov@gmail.com

С. В. Арановский

к.т.н., с.н.с. кафедры Систем управления и информатики

E-mail: s.aranovskiy@gmail.com

Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики

Аннотация. Рассмотрена задача адаптивной компенсации детерминированного возмущения, представляющего собой конечную сумму синусоидальных сигналов с неизвестными параметрами. Число образующих возмущение гармоник неизвестно, однако задается верхняя граница их количества. Объект устойчив, но может содержать неустойчивые нули. Модель объекта полагается известной. Представлен метод компенсации возмущения, не требующий явного обращения передаточной функции объекта и основанный на декомпозиции возмущения. Представлены результаты моделирования, иллюстрирующие работоспособность предложенного метода.

Введение

Одной из важнейших задач теории управления является компенсация возмущений, действующих на объект управления. Возмущения могут выражаться, к примеру, в виде шумов или вибраций. Существуют пассивные методы борьбы с подобными возмущениями, которые выражаются во внедрении некоторых конструктивных решений (разнообразные демпферы), которые способны автономно решать задачу компенсации в определенных пределах. Однако, как правило, пассивные устройства не обладают гибкостью в управлении, которая присуща активным методам компенсации возмущений. Среди последних можно выделить активные методы подавления шумов (ANC — active noise control, к примеру, [1]), а так же активные методы виброзащиты, которые могут проявляться, к примеру, в таких прикладных областях, как производство жестких дисков компьютеров [2; 3] и даже управление вертолетами [4]. Подробнее с разнообразием методов можно ознакомиться в обзорной статье [5].

Возмущения могут представлять собой неизмеряемые мультисинусоидальные сигналы с неизвестными параметрами. Если параметры объекта и входные сигналы системы известны, возмущение может быть оценено на основании измерений входа и выхода. В этом случае закон управления, обеспечивающий компенсацию, мог бы быть найден путём явного с обращения передаточной функции объекта управления, что привело бы к неустойчивости системы в случае наличия неустойчивых нулей. К тому же, как правило, явное обращение нереализуемо для технических объектов, так как степень полинома числителя передаточной функции превысила бы степень полинома знаменателя. Поэтому поиск решения задачи компенсации производится в семействе адаптивных законов управления. В работе предложен прямой адаптивный метод, обеспечивающий компенсацию мультисинусоидального возмущения с неизвестными параметрами (частотами, амплитудами и фазами).

Постановка задачи

Рассмотрим систему управления в дискретном времени $t = T \cdot k$, где T — период дискретизации, k — дискретное время, структурная схема которой представлена на рис. 1.

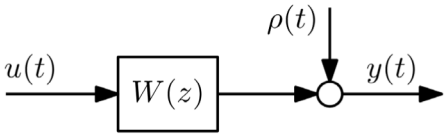


Рис. 1. Структурная схема рассматриваемой системы управления

где z^{-1} — элемент задержки такой, что $y(t)z^{-1} = y(t+1)$ Система является автономной. Сигналом управления в системе является сигнал $u(t)$, а регулируемой переменной

$$y(t) = W(z)\,u(t). \tag{1}$$

Объект управления $W(z) = B(z)/A(z)$ линейный, стационарный, устойчивый. Модель объекта задана в виде передаточной функции

$$\hat{W}(z) = \frac{\hat{B}(z)}{\hat{A}(z)}, \tag{2}$$

и полагается известной, однако может содержать неустойчивые нули. К объекту управления приложено внешнее детерминированное возмущение, представляющее собой конечную сумму N синусоидальных сигналов вида:

$$\rho(t) = \sum_{i=1}^{N} A_i \sin(\omega_i t + \varphi_i). \tag{3}$$

Параметры возмущения (3): амплитуды A_i, частоты ω_i, фазы φ_i — неизвестны. Точное число образующих возмущение синусоид N так же неизвестно, однако известен верхний предел их количества $N_{\max}: N \le N_{\max}$. Частоты возмущений ограничены $\omega_{\min} \le \omega_i \le \omega_{\max}$.

Целью управления является компенсация возмущения $\rho(t)$ или, иными словами, сведение выходного сигнала $y(t)$ в установившемся режиме к нулю

$$y(t) \to 0 \text{ при } t \to \infty. \tag{4}$$

Оценка возмущения (3) может быть получена как

$$\hat{\rho}(t) = y(t) - \hat{y}(t), \tag{5}$$

где $y(t)$ выход объекта, а $\hat{y}(t) = \hat{W}(z)\, u(t)$ представляет собой выход модели. Так как модель объекта считается известной, $\hat{W}(z) = W(z)$ и оценка возмущения (5) представляется в виде:

$$\hat{\rho}(t) = \rho(t) + \varepsilon_\rho(t), \tag{6}$$

где невязка $\varepsilon_\rho(t) \to 0$.

Закон управления, обеспечивающий выполнение цели управления (4) для известного объекта (2) и возмущения с неизвестными параметрами (3), в идеальном случае мог бы быть синтезирован в виде:

$$u(t) = -\hat{W}^{-1}(z)\, \hat{\rho}(t). \tag{7}$$

Однако, на практике закон управления (7) нереализуем из-за того, что его реализация требует явного обращения передаточной функции объекта, а в этом случае максимальная степень полинома числителя превысила бы максимальную степень полинома знаменателя. Тогда поиск решения задачи компенсации (4) произведем в семействе адаптивных законов управления.

Основной результат

Рассмотрим некий вектор $\overline{\mathbf{x}}(t)$, элементы которого представляют собой синусоидальные сигналы тех же частот, что и возмущение (3), но отличных амплитуд $B_{i,j}$ и $\psi_{i,j}$ вида:

$$\overline{x}_j(t) = \sum_{i=1}^{N} B_{i,j} \sin(\omega_i t + \psi_{i,j}), \quad j = 1, \ldots, 2N_{\max}. \tag{8}$$

Пусть существует вектор $\overline{\mathbf{k}}$ такой, что существует декомпозиция возмущения $\hat{\rho}(t)$ вида:

$$\overline{\mathbf{k}}^T \cdot \overline{\mathbf{x}}(t) = \sum_{j=1}^{2N_{\max}} \overline{k}_j\, \overline{x}_j(t) = \hat{\rho}(t). \tag{9}$$

Условия существования декомпозиции (9) представлены в [6; 7].

Получить вектор $\overline{\mathbf{x}}(t)$ можно путем фильтрации оценки возмущения $\hat{\rho}(t)$

$$\overline{x}_j(t) = F_j(z)\hat{\rho}(t), \quad j = 1, ..., 2N_{\max}, \tag{10}$$

где $F_j(z)$ — линейные устойчивые фильтры. Фильтры должны быть выбраны таким образом, чтобы декомпозиция (9) существовала. Выберем фильтр вида

$$\overline{x}_j(t) = \frac{F_0(z)}{z^{j-1}}\hat{\rho}(t), \quad j = 1, ..., 2N_{\max}, \tag{11}$$

где $F_0(z)$ — базовый фильтр. Структурная схема банка фильтров (11) представлена на рис. 2.

Для дискретных систем доказано (см. работы [6; 7]), что при построении фильтра в виде (11), декомпозиция всегда существует.

Пропустим сигналы (8) через модель объекта управления (2) и получим набор сигналов

$$x_j(t) = \hat{W}(z)\,\overline{x}(t). \tag{12}$$

Так как объект устойчивый, сигналы в наборе (12) будут иметь те же частоты, что и (8) с точностью до экспоненциально затухающих со временем членов. Тогда можно считать, что для сигналов $x_j(t)$ существует декомпозиция, аналогичная (9):

$$\mathbf{k}^T \cdot \mathbf{x}(t) = \sum_{j=1}^{2N_{\max}} k_j x_j(t) = \hat{\rho}(t). \tag{13}$$

Поиск вектора коэффициентов $\mathbf{k}$ будет производиться не аналитическими методами, а с использованием аппарата теории идентификации. Тогда вектор оценок $\hat{\mathbf{k}}$ может быть получен путём минимизации следующего критерия:

$$\hat{\mathbf{k}}(t) = \arg\min_{k} \sum_{t} \left(\hat{\rho}(t) - \sum_{j=1}^{2N_{\max}} x_j(t)\,\hat{k}_j \right)^2. \tag{14}$$

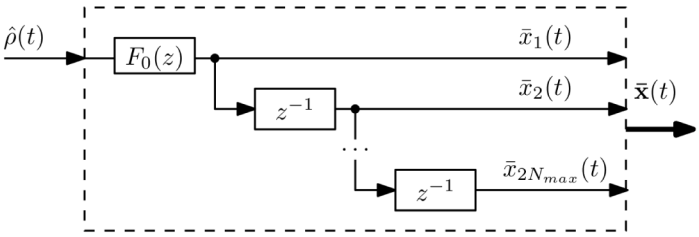


Рис. 2. Структурная схема банка фильтров (11)

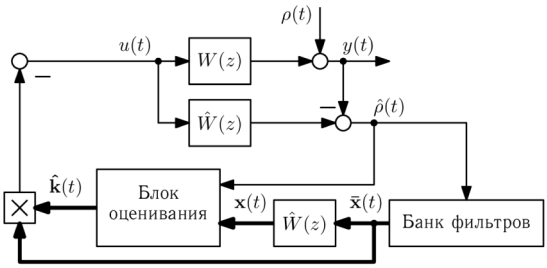


Рис. 3. Структурная схема замкнутой системы

При $\hat{\mathbf{k}}(t) = \mathbf{k}$ критерий (14) обнуляется, и закон управления, обеспечивающий решение задачи компенсации (4), может быть сформирован как линейная комбинация вида:

$$u(t) = -\hat{\mathbf{k}}^T(t) \cdot \overline{\mathbf{x}}(t) = -\sum_{j=1}^{2N_{\max}} \hat{k}_j(t)\,\overline{x}_j(t). \qquad (15)$$

Структурная схема полученной модели (1) — (3), (5), (11), (12), (14), (15) представлена на рис. 3.

Следует отметить, что регулируемая переменная $y(t)$ не участвует в алгоритме компенсации, и представленная на рис Рисунок 3 схема может быть приведена к эквивалентному разомкнутому контуру, структурная схема которого изображена на рис. Рисунок 4.

В таком виде проще анализировать систему: объект управления $W(z)$ является устойчивым, фильтры $F_j(z),\ \ j = 1, ..., 2N_{\max}$, так же устойчивы. Тогда требование к устойчивости системы можно свести к требованию устойчивости алгоритма адаптации: если оценки $\hat{\mathbf{k}}$ ограничены при $\varepsilon_\rho(t) \to 0$, то и все сигналы в контуре так же ограничены. Если же выполняется:

$$\hat{\mathbf{k}}(t) \to \mathbf{k},\ \text{то}\ y(t) \to 0, \qquad (16)$$

и, следовательно, задача компенсации (4) будет решена.

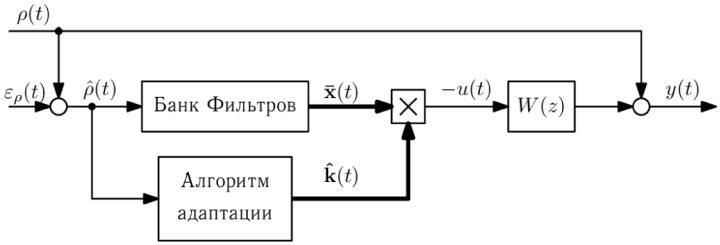


Рис. 4. Структурная схема эквивалентного разомкнутого контура

Численное моделирование

В качестве примера рассмотрим дискретную линейную стационарную устойчивую модель малогабаритного оптического телескопа (подобные объекты рассмотрены в [8–10]), заданную в виде передаточной функции восьмого порядка

$$W(z) = \frac{0.01446z^7 - 0.009906z^6 - \ldots - 0.009924z + 0.01444}{z^8 - 5.897z^7 + 16.22z^6 - \ldots + 12.05z^2 - 3.774z + 0.5577}. \qquad (17)$$

Пусть возмущение вида (3) имеет в своем составе не более трех $N_{\max} = 3$ синусоидальных сигналов из набора

$$\begin{aligned} \rho_1(t) &= 0.5\sin(52 \cdot 2\pi t + 30 \cdot \pi / 180), \\ \rho_2(t) &= 0.9\sin(60 \cdot 2\pi t + 60 \cdot \pi / 180), \\ \rho_3(t) &= 1.5\sin(70 \cdot 2\pi t + 90 \cdot \pi / 180), \\ \rho_4(t) &= 1.2\sin(64 \cdot 2\pi t + 40 \cdot \pi / 180). \end{aligned} \qquad (18)$$

В качестве базового фильтра выберем эллиптический полосовой фильтр 10 порядка с бесконечной импульсной характеристикой (БИХ) и полосой пропускания 50–75 Гц. Амплитудно-частотная характеристика фильтра представлена на рис. Рисунок 5.

Для оценки вектора параметров $\hat{\mathbf{k}}(t)$ при помощи выражения (14) воспользуемся рекуррентным методом наименьших квадратов со списыванием [11]:

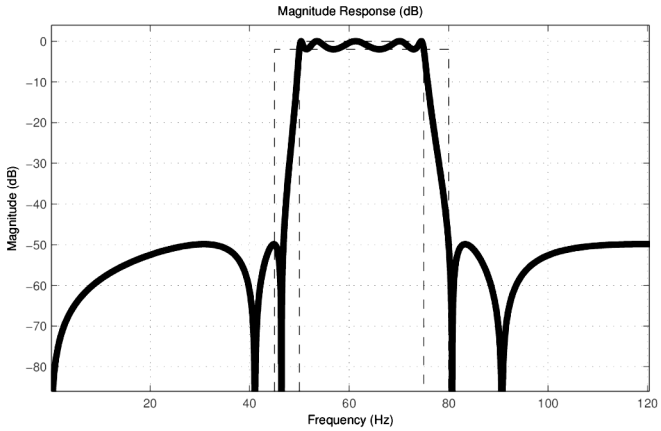


Рис. 5. Амплитудно-частотная характеристика базового фильтра

$$\hat{k}(t) = \hat{k}(t-1) + G(t)\,\varepsilon(t),$$
$$\varepsilon(t) = \hat{\rho} - x^T(t)\,\hat{k}(t-1),$$
$$G(t) = P(t)\,x(t) = \frac{P(t-1)\,x(t)}{\lambda + x^T(t)P(t-1)\,x(t)}, \qquad (19)$$
$$P(t) = \frac{1}{\lambda}\left(P(t-1) - G(t)\,x^T(t)\,P(t-1)\right),$$

где $x(t) = \{x_j\}$, $j = 1, ..., 2N_{\max}$, $0 \le \lambda \le 1$ — коэффициент списывания.

Результаты симуляции модели (1) — (3), (5), (11), (12), (14), (15) на примере (17)–(19) представлены на рисунках ниже.

Как видно из рис. 6, представленный алгоритм полностью компенсирует возмущения, состоящие не более, чем из $N_{\max} = 3$ гармоник, как и было заявлено. При большем же числе гармоник, $N \ge 4$ компенсации возмущения

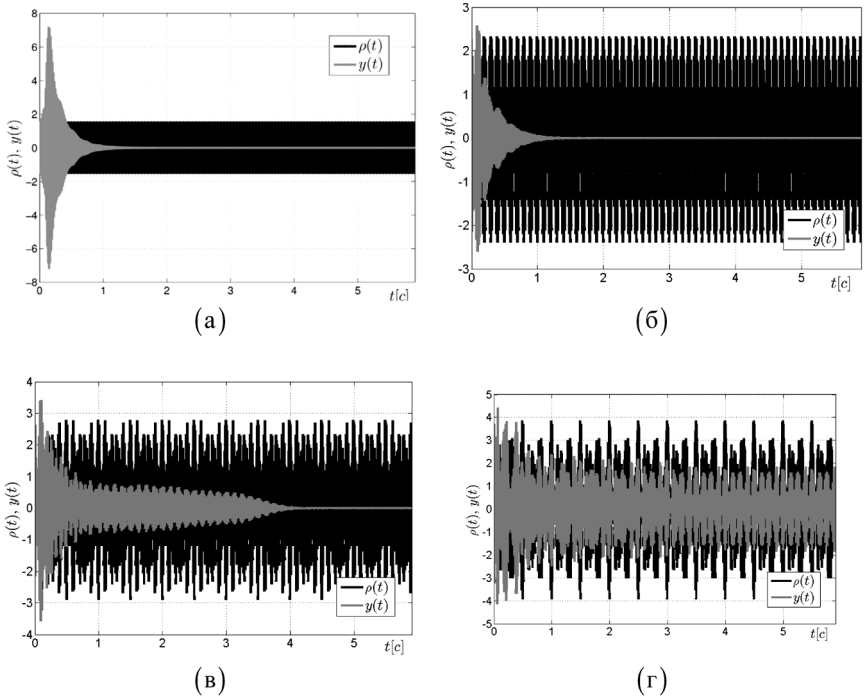


Рис. 6. Результаты моделирования при $2N_{\max} = 6$, $\lambda = 0.995$ и возмущениях (18):
(а) — при $N = 1$, $\rho(t) = \rho_3(t)$; (б) — при $N = 2$, $\rho(t) = \rho_2(t) + \rho_3(t)$; (в) — $N = 3$, $\rho(t) = \rho_1(t) + \rho_2(t) + \rho_3(t)$; (г) — $N = 4$, $\rho(t) = \rho_1(t) + \rho_2(t) + \rho_3(t) + \rho_4(t)$

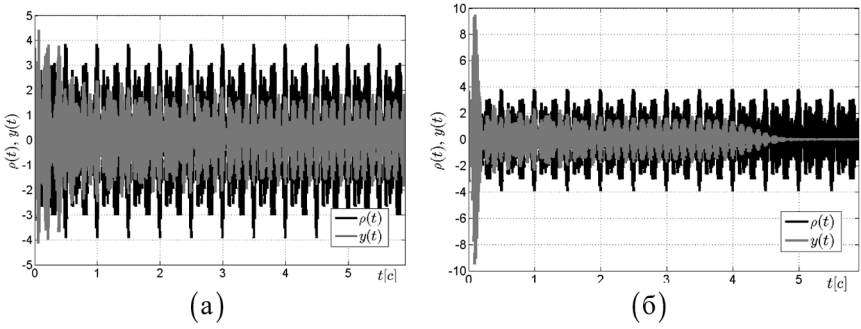


Рис. 7. Результаты моделирования при $\lambda = 0.995$ и возмущении $\rho(t) = \rho_1(t) + \rho_2(t) + \rho_3(t) + \rho_4(t)$ из набора (18): (а) — при $2N_{\max} = 6$; (б) — при $2N_{\max} = 10$

не происходит, однако можно добиться компенсации за счет увеличения $N_{\max}$ и, как следствие, порядка банка фильтров (11) и алгоритма адаптации (19).

Как видно из рис. 7, повышение порядка фильтра позволило скомпенсировать возмущение.

Заключение

Предложен прямой метод адаптивной компенсации внешнего детерминированного возмущения, основанный на декомпозиции возмущения. Предложенный метод обеспечивает устойчивость замкнутой системы, не предполагает промежуточной идентификации частот возмущающих гармонических сигналов и использует измерения только выходного сигнала. Приведены результаты моделирования компенсации возмущения, представляющего собой сумму нескольких синусоидальных сигналов с неизвестными параметрами.

Литература

1. *Kuo S. M., Morgan D. R.* Review of DSP Algorithms for Active Noise Control // Proceedings of the 2000 IEEE International Conference on Control Applications. Anchorage, Alaska, USA, 2000. С. 243–248.
2. *Sacks A., Bodson M., Khosla P.* Experimental results of adaptive periodic disturbance cancellation in a high performance magnetic disk drive // ASME Journal of Dynamic Systems Measurement and Control. 1996. No. 118. С. 416–424.
3. *Chen X., Tomizuka M.* A minimum parameter adaptive approach for rejecting multiple narrow-band disturbances with application to hard disk drives // IEEE Transactions on Control Systems Technology. 2012. No. 20. С. 408–415.
4. *Bittanti S., Moiraghi L.* Active control of vibrations in helicopters via pole assignment techniques // IEEE Transactions on Control Systems Technology. 1994. No. 2 (4). С. 343–350.

5. *Bodson M.* Rejection of periodic disturbances of unknown and time-varying frequency // International Jounal of Adaptive control and signal processing. 2005. T. 19. C. 67–99.
6. *Aranovskiy S.* Adaptive attenuation of disturbance formed as a sum of sinusoidal signals applied to a benchmark problem // European control conference. Switzerland, Zurich: July, 2013. C. 2879–2884.
7. *Aranovskiy S., Freidovich L.* Adaptive compensation of disturbances formed as sums of sinusoidal signals with application to an active vibration control benchmark // European Journal of Control. 2013. No. 19. C. 253–265.
8. *Васильев В. Н.* и др. Состояние и перспективы развития прецизионных электроприводов комплексов высокоточных наблюдений // Известия ВУЗов. Приборостроение. 2008. № 6. С. 5–11.
9. *Арановский С. В., Фуртат И. Б.* Робастное управление безредукторным прецизионным электроприводом оси оптического телескопа с компенсацией возмущений // Мехатроника, автоматизация, управление. 2011. № 9. С. 8–13.
10. *Арановский С. В., Бардов В. М.* Метод оптимальной идентификации параметров линейного динамического объекта в условиях возмущения // Проблемы управления. 2012. № 3. С. 35–40.
11. *Åström K., Wittenmark B.* Adaptive Control. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1994. Issue 2. 580 pp.

РАСПОЗНАВАНИЕ ПОЛА ЧЕЛОВЕКА ПО ФОТОГРАФИИ

А. С. Кавокин

студент 4 курса кафедры системного программирования СПбГУ

E-mail: alexanderkavokin@gmail.com

С. Ю. Сартасов

старший преподаватель кафедры системного программирования СПбГУ

E-mail: stanislav.sartasov@gmail.com

Аннотация. Распознавание пола человека является одной из задач современной кибернетики и машинного обучения. Пол можно определять по фотографии, видео или в режиме on-line при работе с камерой.

В данной работе автор предлагает алгоритм распознавания пола, используя фронтальную фотографию лица человека.

Введение

Получение информации о человеке по его фотографии является практически значимой задачей. Результаты распознавания личности могут быть использованы для улучшения безопасности, в финансовых операциях, в бизнесе. Например, распознавание пола может пригодиться в рекламе, когда необходимо оценить аудиторию, которой будет предъявляться продукт.

Обзор предыдущих результатов в данной области

Один из стандартных способов распознавания пола предполагает использование метод опорных векторов (Support Vector Machine). Основная идея метода — перевод исходных векторов, содержащих данные, в пространство более высокой размерности и поиск разделяющей гиперплоскости с максимальным зазором в этом пространстве.

Для каждого пикселя изображения считается оператор локальных бинарных шаблонов (Local Binary Pattern). Он представляет собой описание окрестности пикселя в двоичной форме. Оператор использует восемь пикселей окрестности, принимая центральный пиксель в качестве порога. Пиксели, которые имеют значения больше, чем центральный пиксель, или равные ему, принимают значение «1», те, которые меньше центрального пикселя, принимают значения «0». Таким образом, получается восьмиразрядный бинарный код, который описывает окрестности пикселя.

Существуют различные работы, основанные на данном методе. В работе Boosting Sex Identification Performance [1], проведенной под патронажем

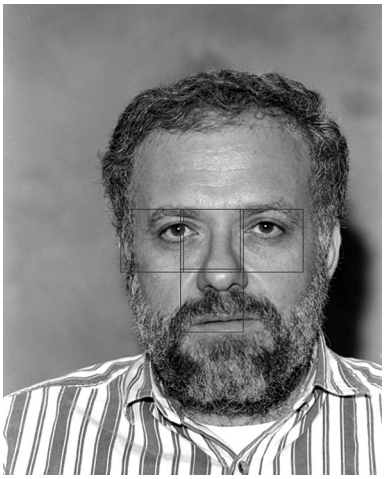


Рис. 1. Пример результата поиска ключевых областей

Google Research Inc. был достигнут результат в 93.1% точности распознавания. В 2008 году в работе Gender Recognition from body [2] , проведенной в Университете штата Иллинойс в Урбана-Шампейн была создана программа, определяющая возраст человека. При определении одним из факторов является распознавание пола. Неплохого результата с ошибкой всего в 3.4% добились авторы в работе Gender Classification with Support Vector Machine [3].

Постановка задачи

Целью работы является разработка алгоритма распознавания пола человека по фотографии на основе классификатора SVM и последующая его реализация. Основываясь на указанной выше цели, были поставлены следующие задачи:

1. получить открытую базу данных с лицами людей;
2. реализовать парсер для базы данных;
3. придумать алгоритм обучения классификатора SVM;
4. реализовать обучение классификатора SVM.

Алгоритм идентификации по опорным регионам

Было предложено анализировать не всю фотографию, а только некоторые ключевые области на лице. Для этого требовалось обучить один классификатор SVM распознаванию и поиску этих областей. Затем необходимо было обучить второй классификатор, который способен определять предполагаемый пол владельца каждой найденной ключевой области. Учитывая возможные погрешности, было решено использовать разные веса для ключевых областей. И в результате анализа этих областей определить пол человека.

Реализация

Выбор базы данных

В качестве базы данных была выбрана БД color FERET. Данная БД содержит 2,5 тысячи различных фотографий 856 людей, собранных в период между декабрем 1993 и августом 1996. К каждой фотографии прилагаются метаданные, содержащие такую информацию как расположение ключевых

точек, пол, возраст, внешний вид (в очках/без очков) и прочие. Дополнительным преимуществом данной БД является то, что она находится в свободном доступе.

Парсер базы данных

Для каждой фотографии в базе данных были предоставлены метаданные. Анализируя их, можно получить информацию о поле, нации, расположении глаз, носа и рта, возрасте, угле поворота лица и пр. В связи с выбранным алгоритмом было решено использовать расположение глаз, рта и носа, а также пол. Парсер должен был для каждой фотографии получать требуемые данные из соответствующих метаданных.

Обучение классификатора SVM

Для обучения первого классификатора было выбрано 100 фронтальных фотографий из базы данных color FERET. Каждое изображение было разделено на клетки, размером 80x80 пикселей, имеющие небольшой (5 пикселей) нахлест друг на друга. У каждого пикселя считалось значение ЛБШ. В дальнейшем использовались только равномерные ЛБШ (не имеющие подряд более трех нулей или единиц). После чего для всех клеток создавалась гистограмма на основе посчитанных значений ЛБШ. Поскольку было 4 типа ключевых областей, было выбрано 5 классов (еще 1 класс как не ключевая область). Для работы с SVM была использована библиотека libsvm для C#. В качестве данных для обучения были строки в формате:

<label> <index1>:<value1> <index2>:<value2> …,

где

<label> — натуральное число, характеризующее класс;
<index> — целое число, характеризующее столбец гистограммы;
<value> — целое число, характеризующее количество элементов в соответствующем столбце гистограммы.

Для обучения второго классификатора использовалось 8 классов областей: мужской нос, женский нос, и аналогично для рта и глаз. Данные для обучения создавались аналогичным образом, как и для первого классификатора.

Результаты

В конце работы были получены следующие результаты: на обучающем множестве точность распознавания ключевых областей составляла 95.3%, пол определялся с точностью 87.8%. Эксперименты с фотографиями, не входящими в закрытое тренировочное множество, дали более слабый результат:

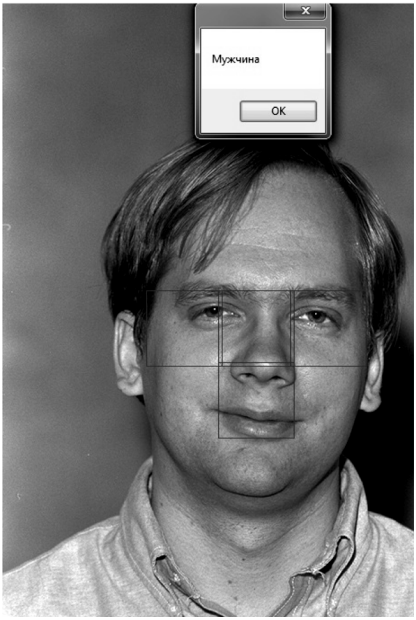


Рис. 2. Пример распознавания мужчины

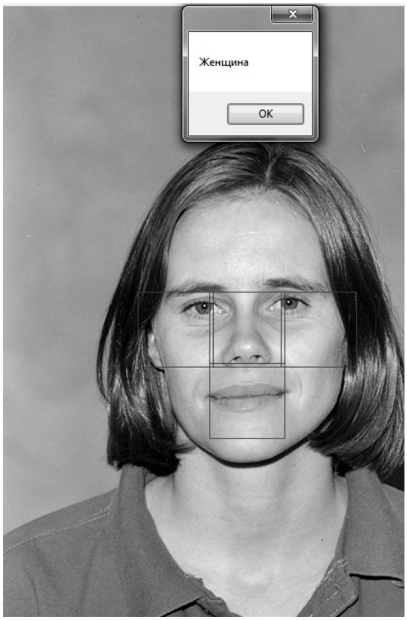


Рис. 3. Пример распознавания женщины

программа обнаружила 83.4% ключевых областей и корректно определила пол в 76.1% случаях.

Заключение

Полученные результаты не являются окончательными. При тестировании возникла гипотеза, что процент распознавания можно увеличить, изменив входные параметры для классификаторов SVM, а также сделать больше различных фотографий для обучения.

В дальнейшем возможно развитие данной работы. Например, распознавание пола по видео или распознавание эмоций.

Литература

1. *Shumeet Baluja and Henry A. Rowley*. Boosting Sex Identification Performance. http://www.cs.cmu.edu/afs/cs/usr/har/www/ijcv2007-sex.pdf
2. *Liangliang Cao, Mert Dikmen, Yun Fu* and *Thomas S. Huang*. Gender recognition from body. http://dl.acm.org/citation.cfm?id=1459470
3. *Baback Moghaddam* and *Ming-Hsuan Yang*. Gender Classification with Support Vector Machines. http://www.cs.utexas.edu/~grauman/courses/378/handouts/moghaddam2000.pdf

ЛОКАЛЬНАЯ ОЦЕНКА КАЧЕСТВА ОТПЕЧАТКОВ ПАЛЬЦЕВ

Т. В. Чугаева

студентка 461 группы кафедры системного программирования СПбГУ

E-mail: chugaevatv@gmail.com

С. Ю. Сартасов

ст. преп. кафедры системного программирования СПбГУ

E-mail: stanislav.sartasov@gmail.com

Аннотация. Идентификация личности с помощью отпечатков пальцев может быть осложнена низким качеством изображений. Для обеспечения более точных результатов и уменьшения количества ошибок ложного допуска и ложного отказа в доступе используют алгоритмы оценки качества.

В данной работе представлено получение локальной оценки качества отпечатков пальцев на основе улучшения отдельной части алгоритма NFIQ [1].

Введение

Отпечатки пальцев всё шире используются для идентификации личности в силу своей уникальности и неизменности. Существует множество алгоритмов распознавания отпечатков, однако все они бессильны перед низким качеством изображений. Причины возникновения такого рода изображений могут быть различны: слишком жирная или сухая кожа рук, крупные поры или случайное движение пальца при снятии отпечатка (см. Рис. 1).

При распознавании изображений низкого качества на основе локальных признаков [2] могут быть найдены ложные минуции (точки, в которых об-

Рис. 1. Примеры отпечатков пальцев

рываются или раздваиваются папиллярные линии). Для того чтобы заранее избежать такого рода ошибок, были придуманы алгоритмы определения качества. Например, NFIQ, который характеризует изображение значением от 1 (высокое качество) до 5 (низкое качество) [1]. Данный алгоритм применяется FBI и DHS.

Целью данной работы является получение локальной оценки качества отпечатков пальцев на основе разных выборок минуций согласно их качеству и сопоставление результатов между собой.

Получение карты качества отпечатка

В алгоритме NFIQ качество отпечатка определяется в зависимости от контрастности изображения, определённости направления папиллярных линий и областей высокой кривизны. Так для каждого отпечатка вычисляются:

1. Карта направлений.
2. Карта низкого контраста.
3. Карта неопределённого направления.
4. Карта высокой кривизны.

Далее полученные результаты объединяются в карту качества отпечатка.

Карта направлений

Цель создания данной карты — показать области с достаточным количеством рёбер и выявить их общее направление.

Сначала изображение делится на непересекающиеся квадратные блоки со стороной $M = 8$ (см. Рис. 2). Для определения общего направления блока требуется рассмотреть некоторую окрестность — окно со стороной $L = 24$ смещение блока относительно окна $N = 8$). Если размер изображения не кра-

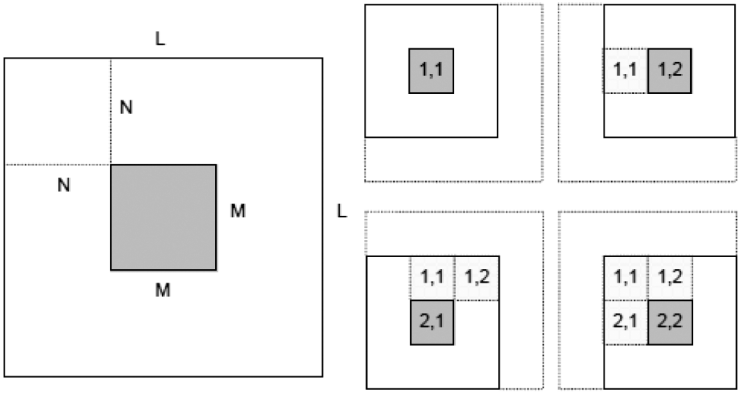


Рис. 2. Блоки изображения и перекрывающимися окнами

тен размеру блока, окно может зайти за границы изображения. В этом случае изображение дополняется средними значениями серого — 128.

Далее для каждого блока изображения мы инкрементально поворачиваем его окно в соответствии с каждым из 16 направлений (угол между направлениями равен 11.25°) и проводим DFT (дискретное преобразование Фурье) для каждого направления, т. е. поворота окна.

Карта низкого контраста

Следующим шагом выполняется поиск блоков низкого контраста. Они чаще всего находятся по краям изображения и являются фоном.

Для определения таких областей используется сравнение распределения интенсивности пикселей в окне блока. Соответственно в низкоконтрастных областях интенсивность будет мало меняться, поэтому и распределение будет небольшим. Однако для блоков с чётко выраженными рёбрами распределение интенсивности будет выше.

Карта неопределённого направления

Данная карта отмечает блоки без определённого направления. Изначально в карте направлений им не присваивается какое-то определённое значение, однако впоследствии оно может возникнуть под влиянием соседних блоков.

Карта высокой кривизны

Высокая кривизна наблюдается в областях, где находятся точки ядра и дельты (см. Рис. 3). Существует два способа нахождения таких блоков:

1. Измерять совокупное изменение направления ребер в соседних блоках.
2. Измерять наибольшее изменение направления между направлением в блоке и направлением каждого из соседей.

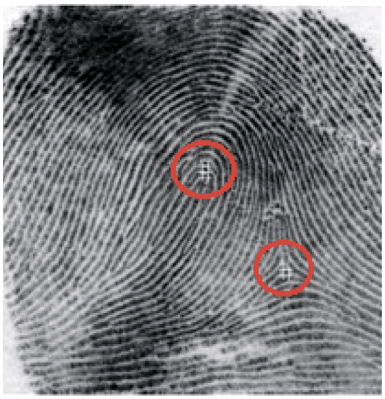


Рис. 3. Точки ядра (слева) и дельты (справа)

Карта качества

Объединяя все описанные выше результаты, вычисляется карта качества отпечатка. Каждому блоку изображения присваивается значение от 0 (плохое качество) до 4 (отличное качество). Тогда для нахождения минуций будем использовать только блоки качества не ниже 2.

Сопоставление отпечатков

Для проверки алгоритма был проведён ряд сравнений отпечатков пальцев с целью вычисления ошибок FAR и FRR, то есть коэффициентов ложного пропуска (False Acceptance Rate) и ложного отказа в доступе (False Rejection Rate). Следует учитывать, что данные показатели взаимосвязаны, и при снижении FAR, как правило, растёт FRR. Также вычислялся равный уровень ошибок EER (Equal Error Rate), при котором ошибки FAR и FRR равны. Как правило, чем ниже EER, тем выше точность сопоставлений отпечатков пальцев.

Была сопоставлено 800 отпечатков из открытой базы данных отпечатков пальцев [3].

Для каждого отпечатка выполнялись следующие шаги:

1. улучшение изображения отпечатка [4];
2. глобальная бинаризация;
3. утончение [5].

Далее осуществлялся поиск минуций по числу пикселей вокруг: если 1 или 3 и больше, то это минуция. Также воедино сливались минуции оказавшиеся в небольшом радиусе вокруг одной из них. Направление минуций определялось на основе поля ориентаций.

После вычисления минуций отпечатков был проведён ряд сравнений для вычисления значений FAR, FRR и EER при учёте всех минуций отпечатков, минуций качества не ниже 3 и минуций качества не ниже 2. Сравнение реализовано с помощью MCC SDK [6]. Полученные результаты представлены на Рисунке 4 и в Таблице 1. Видно, что наименьшее значение EER достигается при выборе минуций качества не ниже 2.

Таблица 1

Значение EER для разных выборок минуций

Качество минуций	Порог	EER
Все минуции	0,5826	0,32128
Минуции качества 3 и 4	0,49767	0,28742
Минуции качества 2, 3 и 4	0,54055	0,25355

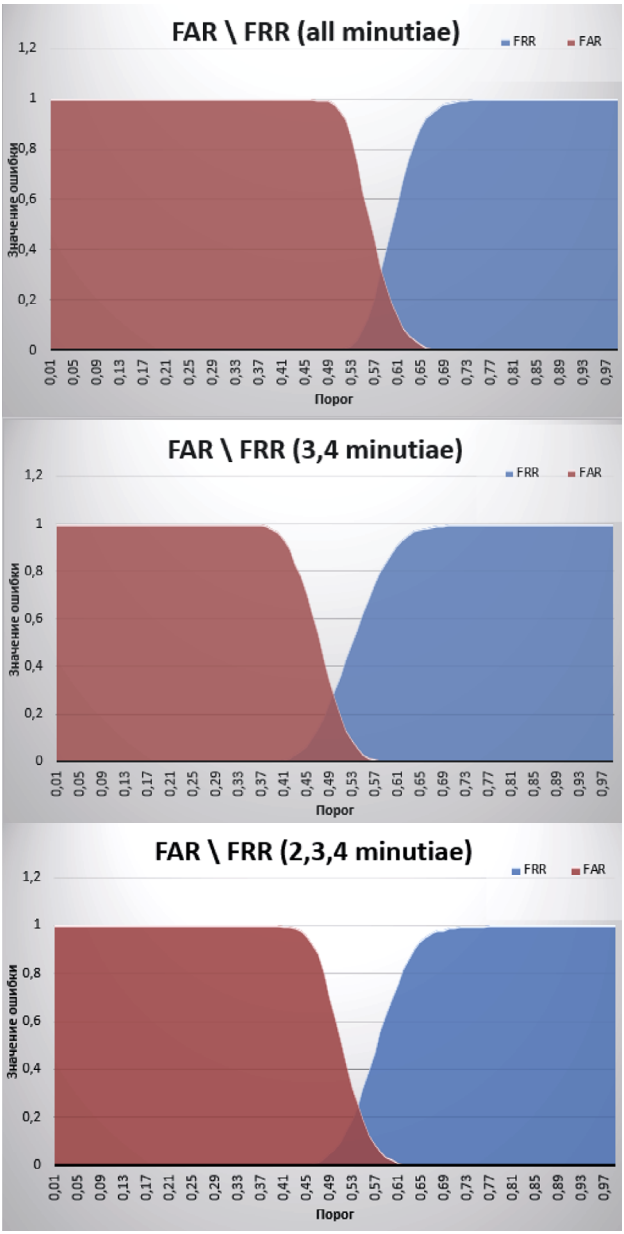


Рис. 4. Значения FAR и FRR для сопоставления отпечатков по всем минуциям, по минуциям качества не ниже 3 и по минуциям качества не ниже 2

Заключение

В работе был рассмотрен алгоритм получения локальной оценки качества отпечатков пальцев NFIQ и было предложено его улучшение. Также было проведено тестирование предложенной версии алгоритма на отпечатках открытой базы данных.

Литература

1. *Craig I. Watson, Michael D. Garris, Elham Tabassi, Charles L. Wilson, R. Michael McCabe, Stanley Janet, Kenneth Ko*. User»s Guide to NIST Biometric Image Software (NBIS) 2004.
2. *Maltoni D., Maio D., Jain A. K., Prabhakar S*. Handbook of Fingerprint Recognition, Second Edition. 2009.
3. *Maio D., Maltoni D., Cappelli R., Wayman J. L., Jain A. K*. FVC2000: Fingerprint verification competition. March 2002.
4. *Hartwig Fronthaler, Klaus Kollreider, and Josef Bigun*. Local Features for Enhancement and Minutiae Extraction in Fingerprints. March 2008. http://www2.hh.se/staff/josef/publ/publications/fronthaler08tip.pdf
5. *Louisa Lam, Seong-Whan Lee and Ching Y*. Suen Thinning Methodologies // A Comprehensive Survey. September 1992.
6. Biometric System Laboratory MCC Software Development Kit (SDK) Version 1.4. 2014.
7. *R. Cappelli, M. Ferrara* and *D. Maltoni*. Minutia Cylinder-Code: a new representation and matching technique for fingerprint recognition // IEEE Transactions on Pattern Analysis Machine Intelligence, vol. 32, no. 12, pp. 2128–2141, December 2010.
8. *R. Cappelli, M. Ferrara,* and *D. Maltoni*. Fingerprint Indexing based on Minutia Cylinder Code // IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 33, no. 5, pp. 1051–1057, May 2011.
9. *M. Ferrara, D. Maltoni* and *R. Cappelli*. Noninvertible Minutia Cylinder-Code Representation // IEEE Transactions on Information Forensics and Security, vol. 7, no. 6, pp. 1727–1737, December 2012.

ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ БИБЛИОТЕКИ ВРЕМЕНИ ИСПОЛНЕНИЯ ВИРТУАЛЬНОЙ JAVA МАШИНЫ CLDC HI ДЛЯ КИБЕРНЕТИЧЕСКОГО КОНТРОЛЛЕРА ТРИК

С. Г. Сарманова

студентка 4 курса кафедры системного программирования СПбГУ

E-mail: snezhana.sarmanova@gmail.com

Аннотация: В статье рассматривается виртуальная Java машина CLDC HI, инструкция по кросс-компиляции CLDC HI в исполняемый модуль, а также проектирование и разработка API на языке Java с поддержкой периферийных устройств кибернетического контроллера ТРИК.

Введение

Робототехника является одним из важнейших направлений научно-технического прогресса, в котором проблемы механики и новых технологий соприкасаются с проблемами искусственного интеллекта. К этой сфере проявляется большой интерес как со стороны опытных программистов, так и со стороны школьников, поскольку для последних это возможность наглядного изучения математики и информатики.

Информатика является одним из сложных школьных предметов, так как детям приходится учиться создавать абстракции, принципиально незримые и неосязаемые. Для облегчения понимания данного предмета в школах активно внедряются робототехнические конструкторы [5]. Примером такого конструктора является проект ТРИК, созданный командой опытных инженеров Математико-механического факультета СПбГУ, который включает в себя кибернетический контроллер и специальный конструктор, с помощью которых можно создавать и программировать роботов [1,4].

Контроллер — это основа робота. Он способен одновременно решать множество задач: обработка аудио- и видеоданных, синтез речи, навигация; управление сервоприводами и моторами; сбор показаний с аналоговых и цифровых датчиков; обмен информацией по беспроводной связи.

В IT индустрии существует множество языков программирования, которые в ознакомительных целях изучаются в школах. Поскольку проект ТРИК внедряется в общеобразовательные процесс, становится актуальна проблема управления роботами на разных языках [5]. По этой причине возникла необходимость внедрения объектно-ориентированного языка Java, одного из популярных и активно развивающихся языков на сегодняшний день.

На данный момент контроллер не поддерживает платформу Java. Как одним из выходов в данной ситуации может быть использование CLDC HotSpot Implementation JVM.

Использование данной виртуальной машины позволит решить проблему отсутствия платформы Java на контроллере ТРИК.

Поэтому целью данной работы является разработка удобного API с поддержкой требуемой периферии контроллера ТРИК для решения задач по программированию роботов на языке Java.

Используемые технологии

При решении поставленной задачи были задействованы следующие технологии.

Контроллер ТРИК

Кибернетический контроллер предназначен для управления роботами, беспилотными летательными аппаратами, средствами передвижения (включая балансирующие «сегвеи»), встраиваемыми устройствами и киберфизическими системами [4].

Контроллер совместим с широким спектром периферийных устройств, имеет в своем составе все необходимое оборудование для управления двигателями постоянного тока и сервоприводами, а также для приема и обработки информации от цифровых и аналоговых датчиков, микрофонов, видеосенсоров. Контроллер снабжён цветным сенсорным дисплеем, программируемыми кнопками, WiFi и Bluetooth, имеет встроенную защиту от перегрузки по току и от глубокой разрядки аккумулятора.

Центральным процессором контроллера ТРИК является OMAP-L138 C6-Integra™ DSP+ARM, разработанный компанией Texas Instruments.

Java Micro Edition

Java 2 Micro Edition (J2ME) — подмножество платформы Java для устройств, ограниченных в ресурсах. J2ME разработана под руководством Sun Microsystems и является заменой похожей технологии — PersonalJava.

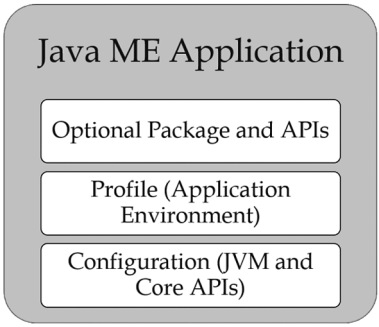


Рис. 1. Структура J2ME приложения

Устройства, на которых может работать J2ME-приложение, определяются поддерживаемой конфигурацией и профилем платформы.

Конфигурация описывает только низкоуровневую часть платформы: возможности языка Java, его виртуальной машины, и базовые классы. Конфигурация призвана объединять все устройства со сходными вычислительными возможностями, независимо от их назначения.

В настоящее время существует две конфигурации J2ME — CDC (Connected Device Configuration) и CLDC (Connected Limited Device Configuration), которые являются подмножестовм J2SE (см. рис. 2). Это означает, что любая библиотека J2ME должна быть доступна на J2SE и любое J2ME приложение должно работать на J2SE.

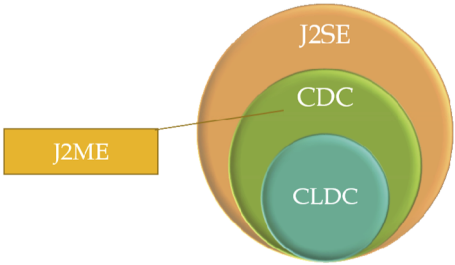


Рис. 2. J2ME конфигурации

Более высокоуровневой частью платформы является профиль. Предполагается, что профиль будет задаваться для каждого крупного класса устройств (мобильные телефоны, игровые автоматы, бытовые приборы). Т. е. профиль определяет тип устройств, поддерживаемых приложением. Профиль дополняет конфигурацию специфическими классами, определяющими область применения устройств.

MIDP (Mobile Information Device Profile) — общеиндустриальный стандартный профиль для встраиваемых устройств, который не зависит от разработчика и производителя устройства. Этот профиль построен на базе CLDC и обеспечивает стандартное окружение и динамическую передачу приложений на пользовательские устройства. MIDP содержит пакеты для работы с графикой, звуком, взаимодействия с консолью (клавиатура и экран), базовый набор классов для отображения на стандартных экранах.

Сочетание конфигурации CLDC и профиля MIDP являются распространенным на рынке встраиваемых систем.

Виртуальная Java машина CLDC HI

Connected Limited Device Configuration HotSpot™ Implementation Virtual Machine (CLDC HI VM) — высокопроизводительная виртуальная Java машина для устройств, ограниченных в ресурсах, разрабатываемая компанией Oracle (бывшей Sun). Это одна из виртуальных машин для «малых» устройств, позволяющая запускать J2ME-приложения на устройствах с ограниченным объемом памяти и вычислительной мощностью.

Приставка «CLDC» означает конфигурацию платформы Java, которая описывает только низкоуровневую её часть: возможности языка Java, его виртуальной машины, и базовые классы. Интерфейсы CLDC в основном являются подмножеством аналогичных интерфейсов Java SE: java.lang, java.io, java.util, java.lang.ref, а также интерфейс javax.microedition.io из Java ME [3].

CLDC HI JVM является оптимизированной виртуальной машиной, которая обеспечивает более быстрое исполнение байт-кода и более эффективное использование ресурсов по сравнению с другими доступными виртуальными машинами, такими как Squawk, KVM, Maxine, CVM и пр. Целевой платформой CLDC HI являются только ARM процессоры.

Кросс-компиляция CLDC HI в исполняемый модуль

После поглощения Oracle компании Sun разработка и поддержка CLDC HI как проекта с открытым исходным кодом были прекращены, поэтому на данный момент документации по сборке и запуску этой виртуальной машины в публичном доступе практически нет.

Была создана методика по сборке Java машины CLDC HI[2], включающая:

- минимальный набор продуктов, необходимых для сборки и компиляции;
- настройка среды кросс-компиляции на основе trikSDK (набор инструментального ПО для кросс-компиляции исходного кода с одной платформы на другую);
- локализованные и исправленные ошибки в исходном коде, связанные с неполной обратной совместимостью инструментов сборки;
- ключи компиляции для запуска на платформе с ARM архитектурой.

Архитектура библиотеки и реализация

На рис. 3 отображена архитектура виртуальной машины CLDC HI с библиотекой com.tril.control для периферийных устройств контроллера. Данный API выступает в роли JSR, который содержит основные интерфейсы и классы по управлению периферийными устройствами контроллера ТРИК.

На данный момент в библиотеке *com.trik.control* на языках C, Java реализованы основные интерфейсы и классы для управления периферией контроллера. На рис. 4 показана структура библиотеки.

Интерфейс Peripherial является основным, он позволяет контролировать доступ к I/O устройствам и управлять, а также хранит о них информацию. Класс PeripherialManager обеспечивает методы для открытия периферийных устройств, которые затем могут быть обработаны как экземпляры класса Peripherial. Интерфейс I2cComunicator поддерживает методы обмена данными по i2c шине между контролером и устройствами, такими как сервоприводы, силовые моторы, энкодеры, аналоговые сенсоры и др. Интерфейс GenericDevice отвечает за управление и контроль датчиками, работающие по принципу широтно-импульсной модуляции. Примерами таких устройств являются сенсоры положения в пространстве — гироскоп и акселерометр, сервоприводы.

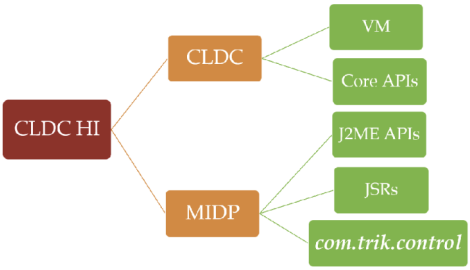


Рис. 3. Архитектура CLDC HI с библиотекой для ТРИК

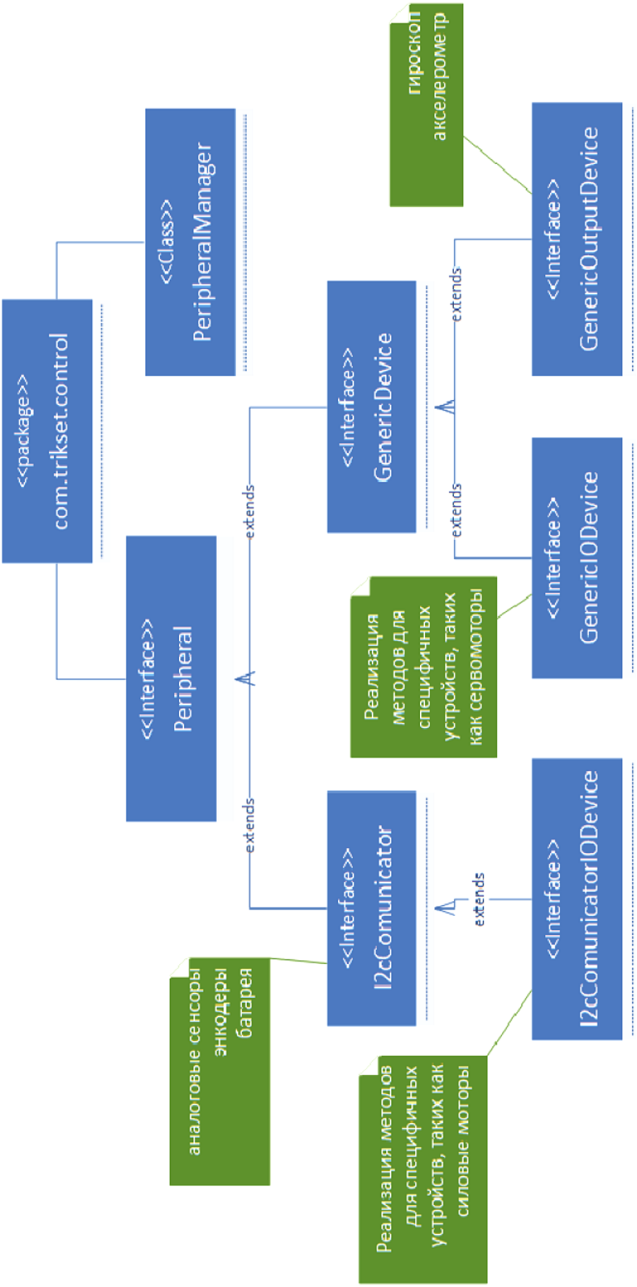


Рис. 4. Классы библиотеки com.trik.control

Заключение

На данном этапе работы в качестве результатов имеется частично реализованная с помощью языков программирования C, Java библиотека времени исполнения виртуальной Java машины CLDC HI для контроллера ТРИК. Библиотека является open source проектом и распространяется под GPLv2 лицензией.

Виртуальная машина CLDC HI позволяет решить проблему отсутствия Java платформы на контроллере. С помощью разработанной библиотеки была решена проблема программирования роботов на языке Java.

Планируется продолжение разработки библиотеки, её апробация на других встраиваемых систем, дополнение другими устройствами.

Литература

1. *Terekhov A. N., Luchin R. M., Filippov S. A*. Educational cybernetical construction set for schools and universities. Advances in Control Education, Vol. 9, 2012. pp. 430–435.
2. *Сарманова С. Г*. Сборка CLDC HotSpot Implementation для ARM // URL: http://habrahabr.ru/post/184952/, дата обращения: 12.04.2014.
3. CLDC HotSpot™ Implementation Architecture Guide CLDC HotSpot Implementation, Version 1.1.3 Java™ ME Platform Sun Microsystems, Inc. July 2006.
4. Конструктор ТРИК, URL: http://www.trikset.com/, дата обращения: 12.04.2014.
5. *Терехов А. Н., Литвинов Ю. В., Брыксин Т. А*. Среда для обучения информатике и робототехнике QReal:Robots // Девятая независимая научно-практическая конференция «Разработка ПО 2013» (CEE SEC(R)-2013), Москва, 24 октября 2013 года.

МОДЕЛИРОВАНИЕ ОДЕЖДЫ В РЕАЛЬНОМ ВРЕМЕНИ

Е. Егорова

Санкт-Петербургский государственный университет

Аннотация. Данная статья написана с целью ознакомить читателя с основными подходами к анимации одежды и показать на примере один из них.

Введение

В настоящее время анимация востребована во многих сферах деятельности человека: фильмы, игры, даже интернет-магазины, в которых используются виртуальные примерочные. Данная область достаточно хорошо изучена и использована, но все же остается обширное поле для исследований, способных повысить качество и реалистичность анимации.

Чаще всего используемая анимация одежды фиксирована, а для упрощения моделирования используются текстуры, имитирующие складки и неровности на одежде.

В связи с доступность существенных вычислительных мощностей и для повышения качества получаемого изображения возникает задача анимации одежды в реальном времени.

Обзор

Существует множество способов анимации одежды. Обычно моделирование анимации одежды происходит либо непосредственно при создании одежды, путем наложения одежды на тело и анимации сразу двух моделей, либо с помощью программных средств.

С помощью графических редакторов можно создать фиксированную анимацию одежды, которая будет статична и ее можно поменять только при изменении анимации тела и при условии, что модель тела и одежда соединены и используются в качестве одной модели.

Использование графических редакторов неудобно еще и тем, что в большинстве случаев требуется покупать дорогостоящее ПО, а также появляется необходимость тщательного и подробного изучения приобретаемого программного продукта.

С помощью специальных методов можно добиться реалистичных результатов моделирования одежды в реальном времени.

В Robotics Institute Carnegie Mellon University был разработан метод моделировании одежды на человеке при движении[4]. Этой метод подразумевает под собой разбиение одежды на полигоны, для каждого из которых вычисляется положение в пространстве и масса получившегося полигона. Далее,

в зависимости от прилагаемых сил, вычисляется новое положение полигона в пространстве. Также учитываются коллизии, возникающие при соприкосновении одежды с твердыми(телом) и мягкими(одеждой) телами. Для этого на фрагменты одежды налагаются специальные ограничения.

Для использования в играх был разработан метод[5], который изначально использует грубую приближенную модель одежды. Поведение одежды моделируется с учетом анимации тела и далее, с помощью дискретных операторов, которые основываются на физических законах, полученная модель детализируется и улучшается до приемлемого приближения. Метод можно использовать в real-time играх, потому что затрачивается минимальное время и память.

Frederic Cordier и Nadia Magnenat-Thalmann описывают метод моделирования движения одежды, который основывается на степени прилегания одежды к телу. Для близко прилегающих участков одежды выбирается ближайшая вершина на модели тела. Для менее облегающих участков(рукава,штанины) высчитываются специальные ограничивающие окружности, которые не позволяют модели тела выходить за пределы одежды. Для свободно прилегающей одежды составляется система масс и пружин, которая способствует свободному и реалистичному движению одежды.

Метод анимации одежды с использованием разделения одежды на участки

За основу взят метод Context-Specific Cloth Simulation[1], который требует меньше времени и затрат на вычисление, и показался наиболее перспективным.

Алгоритм адаптирован для технологии DirectX и реализован на C++, с помощью которой проводятся преобразования, расчеты и визуализация результатов.

В реализованном алгоритме участки одежды 2 типа(штанины, рукава) учитываются в качестве близко прилегающих фрагментов для облегчения вычислений.

Шаг 1. Разделение модели одежды на участки разной степени прилегания к телу.

Для каждого полигона модели одежды находится ближайшая вершина на модели тела. В зависимости от наибольшего и наименьшего расстояния от полигонов до вершин, вычисляется среднее расстояние и полигоны делятся на 2 типа:

1. Полигоны, находящиеся ближе, чем 0.25 среднего расстояния.
2. Полигоны, превышающие 0.5 среднего расстояния.

Данные оценки разделения полигонов были получены эмпирическим путем с целью усреднить количество полигонов каждого типа.

Шаг 2. Перенос анимации с тела на одежду для близко прилегающих к телу участков.

Для близко прилегающих к телу полигонов находится ближайшая вершина на модели тела. С помощью информации, извлеченной из модели тела с уже наложенной анимацией , получаем данные о влиянии костей на вершины модели тела. Сопоставляем соответствующие вершины одежды и тела. Для этого находим ближайшие вершины на модели тела. Далее переносим информацию о влиянии костей. Также для модели одежды копируются данные о строении скелета, анимации костей скелета.

Шаг 3. Проектирование модели пружин и весов для свободно прилегающих участков.

Для свободно прилегающих участков ткани создаются специальные структуры:

1. структуры, описывающие массы точек одежды;
2. пружины, соединяющие эти точки;
3. силы, действующие на точки;
4. структуры, представляющие собой объекты столкновение;
5. структуры, описывающие свободно прилегающие фрагменты одежды,на которые и будут воздействовать силы.

Шаг 4. Построение модели пружин и весов.

В зависимости от времени, прошедшего с момента начала анимации, силы меняются и, следовательно, меняется положение точек, с учетом масс и пружин. Для реалистичного движения одежды необходимо учитывать множество сил(гравитация, трение, движение тела), поэтому структуры должны быть гибкими и требовать минимальные вычислительные затраты.

Также на модели одежды должны быть поставлены специальные контрольные точки, которые всегда находятся на одном и том же месте на модели тела, и на которые не действуют силы. Эти точки необходимы для того, чтобы одежда не упала в следствие гравитации или ветра, поэтому они умеют нулевую массу. В качестве таких точек берутся вершины из близко прилегающих к телу фрагментов одежды.

Шаг 5. Проверка столкновений.

Модель одежды необходимо проверять на столкновения. Столкновения могу возникать в результате появления складок на одежде или же в результате соприкосновения одежды с телом. Для каждого случая вычисления происходят по-разному, поскольку столкновение мягкого объекта с таким же мягким(ткань) или же с твердым(модель тела) физически совершенно различны.

Поскольку для вычисления столкновения модели одежды и модели тела требуются большие затраты производительности, для проверки столкновений используется упрощенная модель тела. Каждая нога человека представлена в качестве 2 цилиндров(бедро и голень) и сферы(колено).

Коллизия происходит, если после наложения сил ткань попадает за внешние границы модели тела. При возникновении столкновения одежды с моделью тела вычисляется сила, которая «выталкивает» ткань из приближенной модели тела.

После построения модели «масс и пружин», можно приступать к моделированию анимации всей одежды в целом.

Шаг 6. Моделирование поведения одежды с учетом анимации тела.

Для каждого полигона и для каждое вершины модели одежды в определенные моменты времени высчитывается новое положение, которое зависит от времени, налагаемых сил в данный момент, положения модели тела, а также от прилегания одежды к телу.

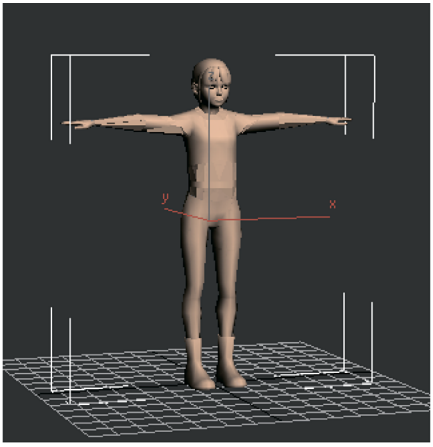
Шаг 7. Визуализация.

Полученный результаты моделирования анимации одежды визуализируются с помощью технологии DirectX.

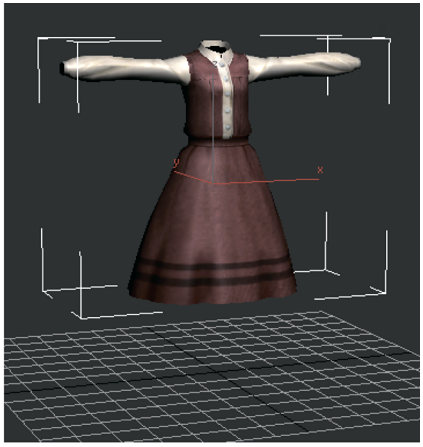
Проведенные эксперименты

Составляющие эксперимента:

1. Модель человеческого тела с анимацией в .X формате(7000 полигонов, 4500 врешин).
2. Модель одежды без анимации в .X формате(1500 полигонов, 1000 вершин).
3. DirectX SDK 10.
4. Среда выполнения для языка C++ (Microsoft Visual Studio 2013).



Модель тела



Модель одежды

В ходе эксперимента были получены следующие результаты:

1. Время обсчета одного кадра — 0.1 мс
2. Вычислительная сложность алгоритма создания анимации для близко прилегающих участков одежды — $O(n^2)$.
   Сложность обусловлена нахождением ближайших вершин на модели тела для каждой точки одежды.
3. Вычислительная сложность алгоритма создания анимации для свободно прилегающих участков одежды — $O(n^3)$.

Для каждой точки одежды рассчитываются силы, действующие на точку и происходит проверка на столкновение с каждым возможным объектом столкновений. Если происходит столкновение, для точки рассчитываются дополнительные выталкивающие силы.

Степень параллелизма данного алгоритма высока, поскольку вычисления для разных фрагментов одежды можно производить одновременно.

Для свободно прилегающих участков одежды система масс и пружин должна рассчитываться последовательно, поскольку движение точек влияет на поведение связанных с ними пружинами вершин.

Для близко прилегающих участков одежды последовательность вычисления положения вершин не важна.

Результаты

Результаты, полученные в ходе реализации алгоритма, говорят о перспективности полученной разработки для практики и дальнейших научных изысканий.

Литература

1. *Frederic Cordier, Nadia Magnenat-Thalmann*. Context-Specific Cloth Simulation.
2. *Джим Адамс*. DirectX. Продвинутая анимация.
3. *Фленов*. Искусство программирования игр на C++.
4. *David Baraff, Andrew Witkin*. Large Steps in Cloth Simulation. Robotics Institute Carnegie Mellon University.
5. *Ladislav Kavan, Dan Gerszewski, Adam W. Bargteil, Peter-Pike Sloan*. Physics-Inspired Upsampling for Cloth Simulation in Games.

Математическое и компьютерное моделирование

Прохоров
Владимир Валентинович

д.ф.-м.н., профессор УрФУ
генеральный директор
ООО «Научно-производственный центр „Видикор"»

ВИЗУАЛИЗАТОР МОДЕЛЕЙ МУЛЬТИМЕДИЙНЫХ CDN[1]

И. П. Манакова
аспирант УрФУ
E-mail: manakova.ip@gmail.com

Аннотация: В работе приводятся результаты реализации графического веб-сервиса по созданию и настройке моделей мультимедийных CDN (Content Distribution Network). Описаны основные характеристики модели, выбор среды, структура проекта, основной функционал.

Введение

Мультимедийная CDN (*Content Distribution Network*) — это сеть связанных мультимедийных серверов, основной задачей которых является предоставление клиентам мультимедийных услуг (например, мультимедиа по запросу, онлайн радио и видео, многостороннее общение и др.).

Мультимедийные услуги, предоставляемые через сеть Интернет, в настоящее время пользуются большой популярностью. Возрастает спрос к услугам мультимедийных CDN, которые, в свою очередь, являются генераторами больших объёмов сетевого трафика.

Несмотря на улучшение физических характеристик оборудования, ресурсы публичных каналов связи неоднородны и по-прежнему ограничены. Из-за ограниченности ресурсов сетевого оборудования перед провайдерами мультимедийных CDN возникает важная задача — оптимальное управление ресурсами для поддержания высокого качества обслуживания клиентов.

Изучая мультимедийные сети и способы управления ресурсами таких сетей, а также предлагая способы оптимизации процесса управления [1–5], мы пришли к идее разработки комплекса программ по анализу и управлению мультимедийными CDN.

Предлагаемое программное решение позволит исследовать мультимедийные CDN на примере компьютерных моделей, а также позволит проанализировать существующие и предлагаемые подходы к управлению ресурсами CDN. Использование компьютерного моделирования позволит провести исследования различных вариаций мультимедийных сетей и с разных сторон управления, что было бы сложно реализовать в реальных условиях.

Предлагаемое программное решение строится на следующих принципах:

1. Продукт должен представлять модульную структуру. Отдельно взятый модуль — это единое целое и может использоваться независимо от остальных частей, т.е. может использоваться в других сборках.

[1] Работа проводится при поддержке НПЦ «Видикор» и НТИ (ф) УрФУ. Научным руководителем работы является д.ф.-м.н., профессор УрФУ В. В. Прохоров.

2. Среда разработки должна быть наиболее приближена к реальной среде. Это позволит интегрировать отдельно взятые модули в реальную мультимедийную CDN.
3. Модульная структура должна позволять интегрировать в решение другие модули, разрабатываемые независимо в схожих средах.

Таким образом, были выделены следующие важные составляющие комплекса программ по анализу и управлению мультимедийными CDN:

1. «Визуализатор моделей мультимедийных CDN» — графическая среда для построения структуры сети, в которой можно определить все значимые для мультимедийной CDN параметры.
2. «Визуализатор нагрузки» — графическая среда установки нагрузки на мультимедийную CDN.
3. «Генератор нагрузки» — модуль, имитирующий работу мультимедийной CDN с установленной нагрузкой в заданный временной диапазон с учётом выбранных алгоритмов управления.
4. «Генератор отчётов» — модуль, строящий результирующие отчёты о проделанной работе.

Дальнейшая модернизация может проходить в виде разработки и интеграции дополнительных модулей, которые призваны расширить функционал комплекса по анализу и управлению мультимедийными сетями.

На данный момент нами был реализован «Визуализатор моделей мультимедийных CDN». Далее мы приводим результаты разработки данного решения.

Общие характеристики модели мультимедийной сети

Основываясь на ранее проведённые исследования [1–5], мы определили следующие отличительные особенности мультимедийных CDN для разработки компьютерной модели и её дальнейшего анализа в ракурсе управления:

1. Мультимедийная CDN может быть представлена в виде графа, где вершинами являются узлы, а рёбрами — линии связи.
2. Узлами мультимедийной CDN могут быть репликаторы, ретрансляторы, узлы управления и узлы IP-сети (IP-узлы).
3. IP-узлы представляют собой связующие звенья между мультимедийными узлами. Они определены в модели для назначения участков неподдающихся управлению в рамках мультимедийной CDN. В реальных условиях IP-узлы могут связывать разные сети и передавать любые данные. Нагрузка на такие узлы складывается из нагрузки на мультимедийную CDN и незапланированной нагрузки от других сетей.
4. Ретрансляторы могут получать потоки и передавать их далее клиентам или другим мультимедийным узлам. Они не могут создавать новые потоки, а лишь размножают уже имеющиеся.

5. Репликаторы могут быть источниками мультимедийных потоков или же выступать в роли ретранслятора. Таким образом, в модели могут быть ни с кем не связанных репликаторы, которые являются источниками данных.
6. Узлом управления может быть IP-узел, репликатор или ретранслятор. Одна мультимедийная CDN может иметь несколько систем управления, которые отвечают за связанные с ними узлы. Узел управления решает вопросы распределения нагрузки между другими узлами во время передачи мультимедийных потоков. Он включает в себя алгоритмы управления CDN.
7. Мультимедийная CDN имеет два типа связей: попарные связи узлов, определяющие топологию сети, и мультимедийные маршруты.
8. Мультимедийный маршрут представляет собой связанную последовательность от репликатора-источника до конечного узла. Один маршрут характеризует один поток. Таким образом, одни и те же узлы могут входить в разные маршруты. Кроме того, репликаторы могут создавать на основании входящего потока новые маршруты. IP-узлы не участвуют в формировании маршрутов.
9. Начальными параметрами узлов CDN являются: название узла, IP-адрес, максимальная пропускная способность входящего и исходящего канала, максимальный объём памяти, процессора, популярность узла. Эти параметры определяют максимальную границу ресурсов. Имитируя в дальнейшем работу CDN, мы будем обращаться к этим границам, чтобы определить нагрузку на узлы.
10. Параметрами мультимедийных потоков, которые раздаёт CDN, являются их качество и занимаемая пропускная способность канала.

Учитывая указанные особенности, мы определили необходимые классы. Разделили проект на несколько режимов. Выбрали среду разработки. Реализовали модуль для визуализации модели.

Функциональные возможности модуля «Визуализатор моделей мультимедийных CDN»

В качестве среды разработки был выбран веб-фрэймворк Django, язык программирования Python, база данных SQLLite.

Фрэймворк Django поддерживает модульную структуру и позволяет разрабатывать отдельные, слабо связанные по смыслу части независимо друг от друга. Язык программирования Python, который используется в Django, имеет большое количество готовых библиотек (включая специализированные математические пакеты и пакеты для создания сетевых приложений), используется для широкого класса задач. База данных SQLLite позволяет хранить структуры данных и накопленную статистическую информацию, необходимые для разрабатываемого комплекса.

Используя указанную выше связку, были реализованы четыре режима для визуализатора. Каждый режим представляет собой законченное целое

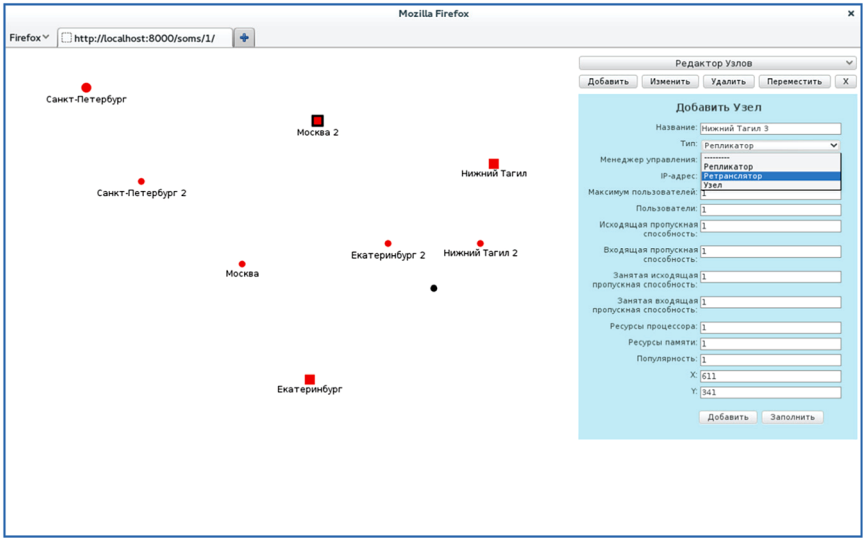


Рис. 1. Редактор узлов

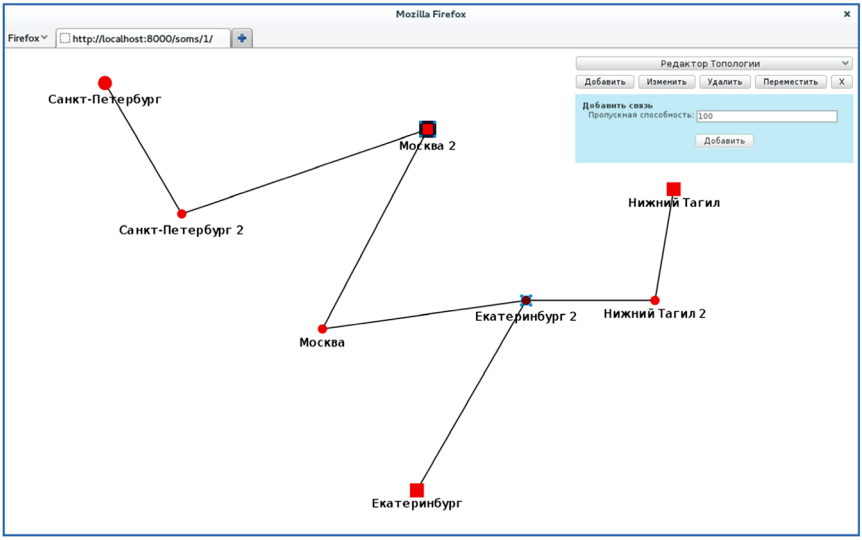


Рис. 2. Редактор топологии

и может использоваться независимо от остальных. Вся информация, задаваемая в режимах, помещается в базу данных. Все изменения также отражаются в базе данных.

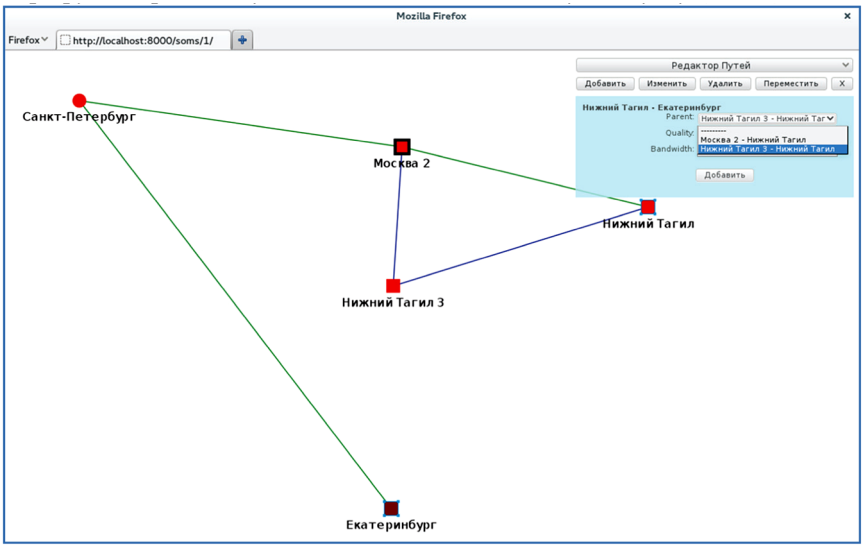


Рис. 3. Редактор мультимедийных маршрутов

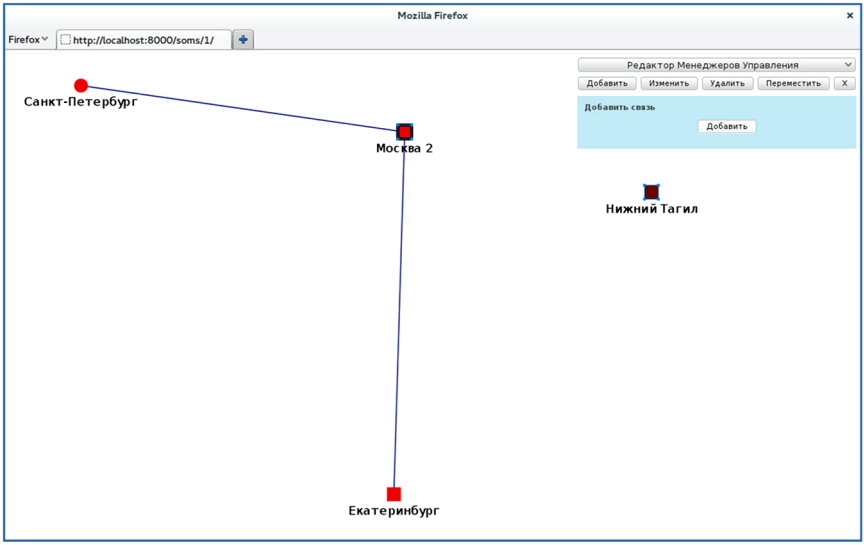


Рис. 4. Редактор систем управления

Режим «Редактор узлов» позволяет размещать на «веб-холсте» графические представления узлов в зависимости от их типа, задавать значения параметрам, перемещать узлы и редактировать их (Рис. 1).

Режим «Редактор топологии» позволяет задавать и визуализировать попарные связи узлов, а также позволяет управлять ими, включая функции запрета на дублирование связей (Рис. 2).

Режим «Редактор путей» позволяет задавать и визуализировать маршруты передачи мультимедийных потоков от узла к узлу (Рис. 3).

Режим «Редактор менеджеров управления» позволяет задать попарные связи узлов управления с другими узлами и визуализировать их (Рис. 4). На схеме отражаются только узлы мультимедийной CDN. В формировании связи обязательно участие хотя бы одного узла управления.

Функционал, реализованный в указанных выше режимах, строится на основных характеристиках мультимедийных CDN, особенностях узлов, среды разработки, выбранного подхода к визуализации. Модуль позволяет визуализировать модели и изменять их, а также хранить настройки в базе данных для дальнейшего использования.

Отдельные части визуализатора могут использоваться для решения схожих задач в других областях знаний. Использование представления в виде графа расширяет класс решаемых задач с использованием разработанного модуля.

Заключение

Реализованный «Визуализатор моделей мультимедийных CDN» является первым этапом разработки программного комплекса по анализу и управлению мультимедийными CDN. В работе описаны основные характеристики модели, выбор среды, структура проекта, основной функционал.

Дальнейшее направление реализации компьютерной модели — разработка остальных частей комплекса, которые были описаны ранее.

Автор работы благодарит научного руководителя В.В. Прохорова за постановку задачи и ценные замечания.

Литература

1. *Манакова И. П.* Менеджер управления мультимедиа-сетью. «СПИСОК-2013» / И. П. Манакова // Материалы всероссийской научной конференции по проблемам информатики, 23–26 апреля 2013 г., Санкт-Петербург. СПб.: Издательство ВВМ, 2013. С. 441–447.
2. *Манакова И. П.* Структура сетей передачи потокового мультимедиа. «Молодёжь. Наука. Инновации» // Сборник докладов 61-й Международной молодежной научно-технической конференции «Молодежь. Наука. Инновации», 21–22 ноября 2013 г. Владивосток: Мор. гос. ун-т, 2013. Т. 1. С. 189–192.
3. *Манакова И. П., Прохоров В. В.* К вопросу об оптимизации построения мультимедиа-сетей. «III Информационная школа молодого учёного»: Сб. научных трудов / И. П. Манакова, В. В. Прохоров // ЦНБ УрО РАН; отв. ред. П. П. Трескова; сост. О. А. Оганова. Екатеринбург: ООО «УИПЦ», 2013. С. 306–315.

4. *Манакова И. П., Петров К.* Б. К вопросу о подключении пользователей к мультимедиа-сети. «Инновации науке» / И. П. Манакова, К. Б. Петров // Материалы XVI международной заочной научно-практической конференции, 28 января 2013 г. Новосибирск: Изд. «СибАК», 2013. Ч. I. С. 94–108.
5. *Манакова И. П., Петров К. Б.* Распределение пользователей по видеосерверам онлайн трансляции с условием минимального перемещения зрителей. «Технические науки — от теории к практике» / И. П. Манакова, К. Б. Петров // Материалы X международной заочной научно-практической конференции, 28 мая 2012 г. / [под ред. Я. А. Полонского]. Новосибирск: Изд. «Сибирская ассоциация консультантов», 2012. С. 27–35.

МОДЕЛИРОВАНИЕ ПРОЦЕССОВ ЭЛЕКТРОЛИЗА МЕДИ

А. С. Лебедев

аспирант кафедры Интеллектуальных информационных технологий УрФУ

E-mail: sas.lebedev@gmail.com

И. Н. Обабков

канд. техн. наук,

директор Института Фундаментального Образования УрФУ

E-mail: i.n.obabkov@ustu.ru

К. Л. Яковлев

ведущий инженер-технолог ЛЭХП ИЦ ОАО «Уралэлектромедь»

E-mail: K. Yakovlev@elem.ru

Аннотация. Статья посвящена построению математических моделей физико-химических процессов, возникающих при электролизе меди. Исследования проводятся на площадке цеха электролиза меди ОАО «Уралэлектромедь».

Введение

Медь стали применять еще до нашей эры, производили тогда ее кустарным способом. С развитием техники развивалось и производство меди. Во второй половине XIX столетия с развитием электротехники и повышением требований к чистоте меди возник новый процесс в металлургии меди — электролитическое рафинирование, научной основой которого служит физическая химия.

С возникновением электроники и ряда других новых видов производств требования к чистоте меди значительно возросли. Появилась необходимость производить медь особо высокой чистоты, содержание основного металла в которой 99,99% и выше.

Электролитическим рафинированием получают медь достаточной чистоты и наиболее полно извлекают содержащиеся в выплавляемой меди драгоценные металлы и редкие элементы (селен и теллур).

Для производства меди необходимо тщательное соблюдение технологического режима и точный контроль. Развивающееся производство рафинированной меди требует постоянного совершенствования технологии электролиза, механизации и автоматизации производственных процессов. В результате электролитического рафинирования получают катодную медь, шлам, содержащий благородные металлы; селен; теллур и загрязненный

электролит, часть которого иногда используют для получения медного и никелевого купороса. Кроме того, вследствие неполного электрохимического растворения анодов получают анодные остатки (анодный скрап). Анодный скрап возвращается на огневое рафинирование

Так как производство электролитической рафинированной меди возрастает, то требуется постоянное совершенствование технологии рафинирования, механизация и автоматизация производственных процессов.

В феврале 2012 года в ОАО «Уралэлектромедь» введена в строй первая очередь нового цеха электролиза меди. По оснащенности оборудованием и уровню автоматизации цех электролиза соответствует самым современным мировым стандартам. Аналогов подобного производства в России на сегодня не существует. Благодаря высокой автоматизации процессов, оборудование данного цеха позволяет получить огромное число данных для обработки и последующего анализа. Ранее процесс электролиза был слишком подвергнут влиянию человеческого фактора, поэтому построение математических моделей было нецелесообразно, так как они в большинстве случаев сильно отличались от реальности.

Основные этапы исследования

Работа условно поделена на несколько этапов, каждый из которых представляет отдельную модель какого-либо процесса или его части, в совокупности же все разработанные модели будут объединяться в единую систему. Основываясь как на отдельную модель, так и на систему в целом, можно будет проанализировать работу цеха, оценить его количественные и качественные показатели.

В результате работы, основываясь на полученные модели, совместно с исследовательским центром ОАО «Уралэлектромедь» будет предложена методология для подготовки работников такого рода цехов.

В настоящее время ведется работа над моделированием влияния различных факторов на напряжение на ванне и расход электроэнергии на электролиз при производстве катодной меди. Эта модель была выбрана не случайно, так как расходы на электроэнергию, потребляемую цехом весьма существенны, поэтому, проанализировав работу цеха на модели, можно будет сделать вывод о том, какие показатели изменить, чтобы снизить потребление электроэнергии, и тем самым снизить затраты на производственный процесс. На рисунке 1 можно увидеть схему влияния различных факторов на напряжение на ванне и на расход электроэнергии.

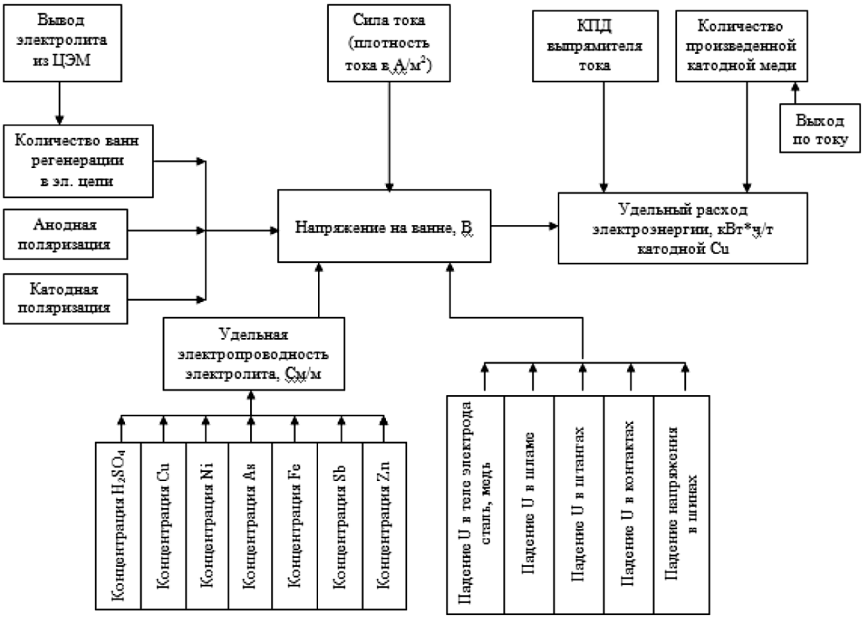


Рис. 1. Факторы, влияющие на напряжение на ванне и на количество электричества, потребляемого при производстве меди

Заключение

В результате проведенной работы был получен список факторов, которые влияют на затраты на электричество при производстве катодной меди. В настоящее время идет процесс построения математической модели по полученному списку факторов, после чего планируется перейти к тестированию модели на основе имеющихся статистических данных.

Литература

1 *Баймаков Ю. В., Журин А. И.* Электролиз в гидрометаллургии. Металлургиздат, 1963.

2. *Исаков Н. И.* Электролиз меди. М.: Металлургиздат, 1962.

3. *Уткин Н. И.* Металлургия цветных металлов,. М.: Металлургия, 1985.

4. *Халезов Б. Д.* Исследования и разработка технологии кучного выщелачивания медных и медноцинковых руд. 2008. Научная библиотека диссертаций и авторефератов http://www.dissercat.com/content/issledovaniya-i-razrabotka-tekhnologii-kuchnogo-vyshchelachivaniya-mednykh-i-mednotsinkovykh#ixzz32MYeEe39

СОЗДАНИЕ СТРУКТУРЫ И ВИЗУАЛЬНОГО ОТОБРАЖЕНИЯ СИСТЕМЫ УПРАВЛЕНИЯ УЧЕБНЫМ ПРОЦЕССОМ НА ОСНОВЕ СЕМАНТИЧЕСКОЙ СЕТИ

Е. В. Шадрина
студент кафедры ИТ, НТИ (ф) УрФУ, Россия, Нижний Тагил
E-mail: shadrina-ev@ntiustu.ru

Назаров М. А.
студент кафедры ИТ, НТИ (ф) УрФУ, Россия, Нижний Тагил
E-mail: nazarov-ma@ntiustu.ru

С. Э. Грегер
доцент кафедры ИТ, НТИ (ф) УрФУ, Россия, Нижний Тагил
E-mail: greger-se@ntiustu.ru

Аннотация. СУУП−система управления учебным процессом, средство для хранения информации о подразделениях института: кафедрах, академических группах, о сотрудниках и студентах. Каждая кафедра включает в себя одну или несколько академических групп, соответственно каждая студенческая группа включает в себя определенное количество студентов. Также система хранит в себе информацию об индивидуальном расписании каждой группы студентов.

Данная статья содержит основные этапы разработки СУУП на основе семантической сети, а также этапы создания шаблонов ввода и вывода информации.

Введение

Создание современного предприятия — сложный и трудоемкий процесс. Управленческая деятельность выступает в современных условиях как один из важнейших факторов функционирования и развития организации. Эффективное управление представляет собой ценный ресурс организации. Следовательно, повышение эффективности управленческой деятельности становится одним из направлений совершенствования деятельности учебного учреждения в целом.

Для эффективного управления учебным процессом, необходимо создать удобный и доступный сервис. Наиболее подходящее для этого средство − Web-сайт. Создание мощной и гибкой системы для подобных сайтов, требования к которой постоянно изменяются — непростая задача.

Целью данной работы является проектирование структуры учебного учреждения, на примере разработки структуры управления учебным процессом, а также создание основных визуальных отображений.

Данная система предназначена для обеспечения взаимодействия всех подразделений предприятия, в данном случае взаимодействия студентов, академических групп, кафедр института. Функциями данной системы является:

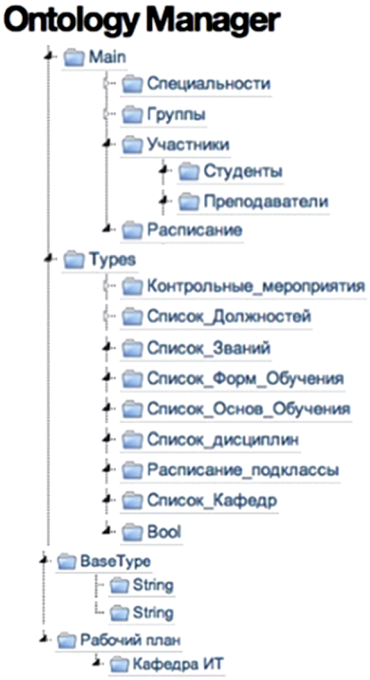
- заполнение объектной базы данных;
- получение информации о структурных подразделениях.

Реализация поставленной цели может быть достигнута выполнением ряда сопутствующих задач:

- описать информационную систему и принцип ее работы;
- проанализировать проектируемую систему и выделить базовые (основные) элементы;
- разработать объектную базу данных, для хранения всей необходимой информации, с учетом определенных требований;
- реализовать удобные шаблоны для заполнения информационной системы и вывода необходимой информации (студентов, сотрудников).

Техническая реализация

После рассмотрения поставленных задач, функций, выполняемых системой, можно приступить к технической реализации.



Для более удобной работы с объектной базой данных была разработана семантическая сеть классов − онтология, на основе структуры описанной системы, которая представляет собой информационную модель предметной области, имеющую вид ориентированного графа, вершины которого соответствуют объектам предметной области, а дуги (рёбра) задают отношения между ними (рисунок 1).

В ходе работы были выделены пять основных классов: «Участник», «Группа», «Кафедра», «Расписание» и «Рабочий план». Для класса «Участник» дополнительно были созданы подклассы «Сотрудник» и «Студент».Для класса «Расписание» было создано большое количество дополнительных классов: «День недели», «Вид недели», «Номер пары», «Занятие», «Вид занятия», «Номер аудитории», «Преподаватель».

Рис. 1. Структура онтологии

Для разработки данной системы был выбран OntoEditor— продукт для Plone, созданный для работы с онтологиями, их классами, объектами классов, свойствами и т. п.

Первым этапом реализации пользовательского интерфейсаявляется создание структуры сайта. При помощи объекта типа Folder были созданы основные разделы: «Расписание», «Кафедры» и «Новости». Для этого данные элементы были помещены в корневую «папку» сайта RootFolder.

На главной странице будут помещаться актуальные новости учебного учреждения, а также ссылка на расписание академических групп. Для их размещения и отображения в качестве ссылок был использован стандартный редактор «Collection». Для отображения новостей на соответствующей вкладке «Новости» был использован стандартный портлет «News». Дополнительно созданы подразделы: «Группы», «Сотрудники», «Студенты».

Для ввода и вывода данных информационной системы необходимо создать визуальный интерфейс.

На основании рассмотренного ранее класса «Участник» и подклассов «Сотрудник» и «Студент» были созданы новыешаблоны ввода-вывода «Сотрудники» и «Студенты», которые содержат все поля, соответствующие спроектированной системе.

Процесс создания шаблонов ввода можно условно разделить на несколько этапов:

- получение всех свойств и связей класса, элемент которого необходимо создать;

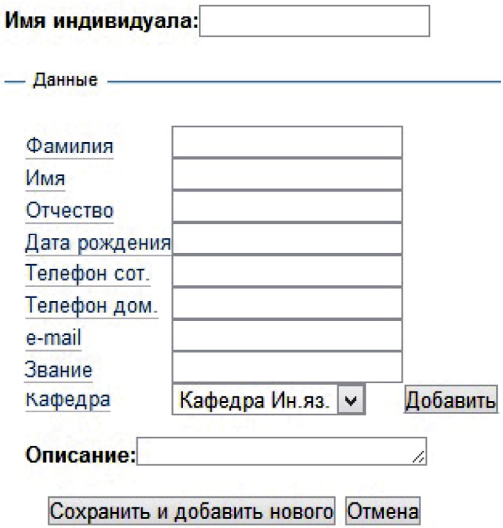


Рис. 2. Шаблон добавления элемента на примере добавления элемента в класс «Сотрудник»

- динамическое создание необходимых элементов для внесения данных в соответствующие свойства и связи, например текстовое поле для свойства «Фамилия»;
- создание обработчиков для кнопок добавления элемента по заполненной форме ввода.

Результат представлен на рисунке 2.

Процесс создания шаблоноввывода так же можно условно разделить на несколько этапов:

1) Получение списка всех объектов одного класса;
2) Получение всех свойств и связей для каждого из объектов;
3) Получение значений свойств и связей соответствующих объектов;
4) Структурирование информации в удобный для понимания вид.

Используя стандартные возможности дополнительного продукта Command, были созданы Command-портлеты «Список сотрудников», «Список студентов», «Список групп» и «Расписание». Созданные портлеты будут использоваться в качестве замены стандартного портлета «Навигация».

Результат представлен на рисунке 3.

Следующимшаблоном для взаимодействия с системой является форма вывода расписания для конкретной студенческой группы. Процесс создания формы можно разбить на следующие этапы:

Титова Е.Ю.

Фамилия	Титова
Имя	Елена
Отчество	Юрьевна
Дата рождения	28.02.1972
Звание	К.н.
Кафедра	Кафедра Ин.яз.

Грегер С.Э.

Фамилия	Грегер
Имя	Сергей
Отчество	Эдуардович
Дата рождения	31.08.1957
Звание	Доцент
Кафедра	Кафедра ИТ

Кобалева Д.А.

Фамилия	Кобалева
Имя	Дина
Отчество	Александровна
Кафедра	Кафедра ГО

Рис. 3. Шаблон просмотра элементов класса на примере класса «Сотрудник»

4 курс			
День недели/Номер пары		БО-401102-ИСТ	БО-401101-ПОВТ
ПОНЕДЕЛЬНИК	1	207,Лекция,Грегер С.Э.,Web	-
	2	207,Практика,Овечкина Е.В.,ИТ	207,Семинар,Грегер С.Э.,ИТ
	3	207,Практика,Овечкина Е.В.,ИТ	207,Лекция,Овечкина Е.В.,Web
	4	207,Практика,Овечкина Е.В.,ИТ	207,Практика,Грегер С.Э.,Web
	5	207,Практика,Овечкина Е.В.,ИТ	207,Семинар,Овечкина Е.В.,Web

Рис. 4. Шаблон просмотра расписания

1) получение всех классов «день» для соответствующей группы;
2) получение всех элементов класса «день»;
3) получение всех свойств и связей, соответствующих элементов каждого класса «день»;
4) получение значений свойств и связей;
5) Структурирование для удобного отображения полной информации расписания.

Результат представлен на рисунке 4.

Заключение

В результате проделанной работы была спроектирована информационная система управления учебным процессом. Были выполнены поставленные задачи.

Спроектированная система имеет простой и удобный интерфейс. Она может стать доступным средством хранения информации учебного учреждения. Для более полного использования системы возможна ее модернизация, создание дополнительных шаблонов ввода-вывода.

Достоинством данной системы является легкая масштабируемость, в случае увеличения подразделений учебного учреждения, а также возможность удаленного использования посредством сети Интернет.

Литература

1. *Грегер С. Э.* Администрирование и интерфейс пользователя CMSPlone / С. Э. Грегер // Федер. агентство по образованию, ГОУ ВПО «УГТУ-УПИ им. первого Президента России Б. Н. Ельцина», Нижнетагильский технол. ин-т (фил.). Нижний Тагил: НТИ (ф) УГТУ-УПИ, 2009. 140 с.
2. *Грегер С. Э.* Сервер приложений «Zope»: Учебное пособие для вузов. М.: Горячая линия—Телеком, 2009. 256 с.: ил.

МОДЕЛИРОВАНИЕ ПРОЦЕССА ПАТРОНИРОВАНИЯ УНИТАРНЫХ ВЫСТРЕЛОВ

И. Б. Литус

гл. конструктор направления «Моделирование физических процессов полигонных испытаний», начальник лаборатории анализа результатов и моделирования процессов при испытании боеприпасов ФКП «НТИИМ»

E-mail: 93@ntiim.ru

А. А. Кукченко

студент кафедры «Информационных технологий» НТИ (ф) УрФУ

E-mail: dezmondwest@mail.ru

Аннотация. Работа, о которой идёт речь в данном докладе, является частью дипломного проекта. Проектирование осуществляется на ФКП «Нижнетагильский институт испытания металлов», которое является военным предприятием и имеет степень секретности, поэтому часть материалов и технических данных, имеющих отношение к данной работе, не могут быть опубликованы.

Введение

Виртуальное моделирование с каждым днём всё больше проникает во все сферы деятельности человека так или иначе связанные с наукой и инженерией. Переход с физической модели исследования на компьютерную обусловлен следующими преимуществами: экономическая выгода, упрощение процесса исследования и ускорение изучения свойств оригинала. Кроме того, что графический интерфейс позволяет более детально рассмотреть различные физические процессы, протекающие на определённых участках времени.

Создание виртуальной модели

Прежде всего, необходимо ознакомится с понятийным аппаратом исследования, для того, чтобы лучше понимать процессы, описанные в данной работе. Выстрелом, в военном деле, называется боеприпас к артиллерийскому орудию. По способу заряжания артиллерийские выстрелы различают:

1. унитарные — заряжаются в один приём и представляют собой цельную конструкцию;
2. раздельно-гильзового заряжания — снаряд не соединён с гильзой, поэтому заряжаются они отдельно.

Патронирование — это процесс механического обжатия гильзы, совмещённой с корпусом снаряда, результатом которого является унитарный ар-

тиллерийский выстрел. На ФКП «НТИИМ» данный процесс осуществляется в цехе, оснащённом специализированными станками.

Целью исследования является построение модели процесса патронирования. Для достижения данной цели необходимо решить следующие задачи:

- изучить предметную область, в частности процесс патронирования;
- изучить чертежи и конструкторскую документацию;
- построить геометрическую модель;
- определить свойства и материалы для соответствующих элементов модели;
- выбрать тип анализа и приложить нагрузки;
- проанализировать полученные результаты.

Актуальность исследования. На данный момент оптимальное усилие патронирования не известно и присутствует вероятность избыточного обжатия гильзы, что приводит к детонации выстрела в каморе орудия. Стоит отметить, что модель патронирования позволяет добиться поставленной задачи только в совокупности с моделью распатронирования и является лишь частью дипломного проекта. Данные разработки являются уникальными, так как работа по повышению эффективности процесса патронирования при помощи виртуального моделирования осуществляется впервые.

Созданию виртуальной модели предшествует процесс систематического сбора и анализа информации о предметной области и объекте исследования, а также изучение конструкторской документации и чертежей. Для данной модели необходимо построение следующих элементов выстрела: корпус, верхний и нижний ведущие пояски, переходная втулка и гильза.

Все этапы моделирования были выполнены в пакете программ ANSYS. Это современная система для автоматизации инженерных расчётов, основан-

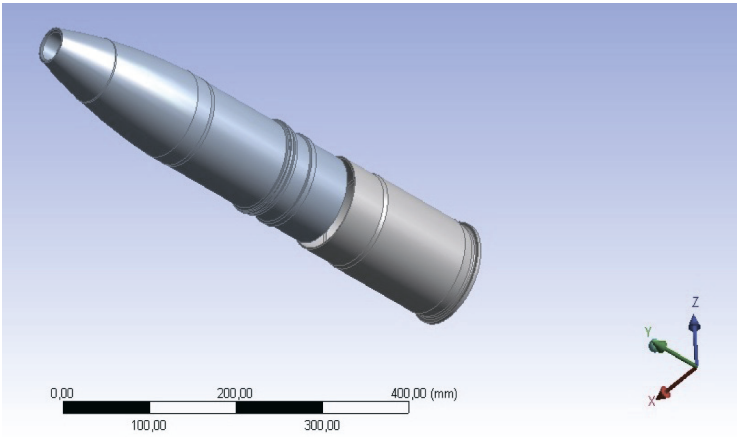


Рис. 1. Геометрическая модель выстрела

ная на численных методах решения дифференциальных уравнений. Первым этапом моделирования является создание геометрической модели всех частей выстрела, которые участвуют в процессе патронирования. Данная часть работы выполнена при помощи платформы ANSYS Workbench, которая содержит все необходимые инструменты системы автоматизации геометрического проектирования. Результат представлен на рисунке 1.

На следующем этапе работы необходимо задать материалы, из которых состоят части выстрела. Гильза состоит из брасса (сплав меди и цинка, в соотношении 30% к 40%-ому цинку), корпус из нержавеющей стали, пояски из меди, переходная втулка из алюминия, кулачки из стали V250.

Следующим этапом является построение сетки, то есть разбиение геометрической модели на конечное количество элементов. Предпочтительной формой элементов является гексаэдр (куб), поэтому применяется сетка hex dominant (гексагональная сетка), которая основана преимущественно на гексаэдрах и в меньшей степени на пирамидах и тетраэдрах.

Далее необходимо выбрать решатель. Так как в рассматриваемой задаче рассматриваются динамические нелинейные нагрузки и деформации, то расчёты необходимо проводить при помощи модуля Transient Structural, который использует встроенный решатель ANSYS. Данный решатель основан на методе конечных элементов, который является численным методом решения дифференциальных уравнений с частными производными, а также интегральных уравнений. Суть метода заключается в разбиении тела на конечное количество элементов и решения уравнений для каждого из них.

После выбора решателя необходимо приложить определённые нагрузки, для воссоздания процесса патронирования. Очередность и время нагрузок

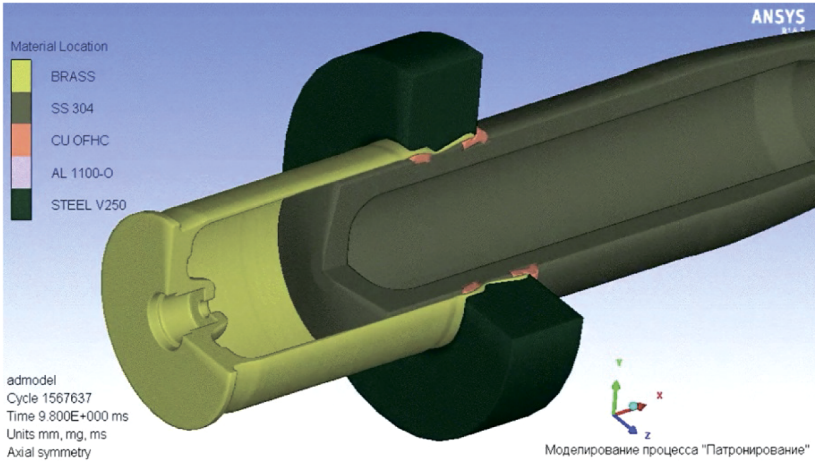


Рис. 2. Обжатие гильзы

устанавливаются при помощи шага нагружения. Для корпуса выстрела необходимо ограничить свободу перемещения, то есть сделать его неподвижным. Далее нужно приложить к гильзе усилие, по направлению к корпусу, при этом необходимо указать расстояние, которое пройдёт гильза, прежде чем окажется на верхнем ведущем пояске. Как только движение гильзы закончено усилие прикладывается к кулачкам, которые, опускаясь на определённое расстояние, деформируют гильзу, тем самым фиксируя её на поясках, что продемонстрировано на рисунке 2.

После создания модели возможно проанализировать физические процессы, сопутствующие данному процессу, к примеру механическое напряжение.

Заключение

В результате данной работы можно сделать вывод о несомненных преимуществах создания компьютерной модели и использования систем инженерного анализа при исследовании физических процессов в области военной промышленности.

Литература

1. *Каплун А. Б., Морозов Е. М., Олферьева М.* А. ANSYS в руках инженера: Практическое руководство. М.: Едиториал УРСС, 2003. 272 с.
2. *Басов К. А*. ANSYS в примерах и задачах / Под общ. ред. Д. Г. Красковского. М.: Компьютер пресс, 2002. 224 с.

Автоматное программирование, машинное обучение и биоинформатика

Шалыто
Анатолий Абрамович

д.т.н.
профессор,
заведующий кафедрой технологии программирования
СПбГУ ИТМО

ПОВЫШЕНИЕ ЭФФЕКТИВНОСТИ ЭВОЛЮЦИОННЫХ АЛГОРИТМОВ ПРИ ПОМОЩИ ОБУЧЕНИЯ С ПОДКРЕПЛЕНИЕМ В НЕСТАЦИОНАРНОЙ СРЕДЕ[1]

И. А. Петрова

студентка кафедры компьютерных технологий Университета ИТМО

E-mail: petrova@rain.ifmo.ru

А. С. Буздалова

студентка кафедры компьютерныхтехнологий Университета ИТМО

E-mail: abuzdalova@gmail.com

М. В. Буздалов

ассистент кафедры компьютерных технологий Университета ИТМО

E-mail: mbuzdalov@gmail.com

Аннотация. Одним из способов повысить эффективность алгоритмов однокритериальной оптимизации является использование вспомогательных критериев. В эволюционных алгоритмах для выбора вспомогательного критерия в процессе оптимизации существует метод EA+RL, основанный на обучении с подкреплением. В предыдущих исследованиях использовались алгоритмы обучения с подкреплением в стационарной среде. Однако если свойства вспомогательных критериев меняются в процессе оптимизации, может быть более эффективно использовать алгоритмы обучения с подкреплением в нестационарной среде.

В данной работе представлены результаты начальных исследований в области использования EA+RL в нестационарной среде. Предложен новый подход к обучению с подкреплением в нестационарной среде. Приводятся результаты его применения к модельной задаче, также проводится сравнение с ранее применявшимися алгоритмами обучения с подкреплением.

Введение

Существуют методы повышения эффективности эволюционных алгоритмов при помощи вспомогательных критериев [1, 2]. Оптимизируемый критерий может выбираться случайным образом [1] или при помощи некоторой эвристики [3]. Первый метод может быть применен для решения любой задачи с конечным набором вспомогательным критериев, но при этом не учитывает специфику задачи, а второй может быть применен для решения лишь конкретной задачи. Метод EA+RL не имеет этих недостатков [4].

[1] Работа выполнена при государственной финансовой поддержке ведущих университетов Российской Федерации (субсидия 074-U01).

В методе EA+RL для выбора вспомогательного критерия, используемого в качестве функции приспособленности (ФП) на данном шаге алгоритма, применяется обучение с подкреплением [5]. Эволюционный алгоритм (ЭА) выступает в роли среды обучения. эффективность данного метода была продемонстрирована и теоретически доказана на ряде задач [4, 6]. Предполагалось, что среда стационарна и поэтому применялись алгоритмы обучения в стационарной среде. Однако в случае когда свойства вспомогательных критериев зависят от этапа оптимизации, возможно эффективнее использовать обучение с подкреплением в нестационарной среде.

В обучении с подкреплением агент обучения применяет действие к среде, в результате чего среда переходит в новое состояние и возвращает агенту численную награду. В методе EA+RL действием является выбор ФП — целевой или одной из вспомогательных. Цель обучения с подкреплением — максимизация суммарной награды. В EA+RL в качестве награды выбирается разность значений целевой функции на лучшей особи в текущем и предыдущем поколениях ЭА. Поэтому при максимизации награды также максимизируется и значение целевой функии. Стоит отметить, что задача оптимизации вспомогательных ФП не ставится, они используются лишь для более быстрой оптимизации целевой ФП.

Существуют различные методы обучения с подкреплением в нестационарной среде [7]. Одним из наиболее эффективных методов, который может быть применим к выбору вспомогательных критериев, является алгоритм RLCD [8]. Однако результаты применения алгоритма RLCD для решения описанной ниже задачи оказались хуже, чем результаты, полученные с помощью методов, применявшихся ранее. Поэтому был предложен новый метод, описанный в следующем разделе.

Описание предлагаемого подхода

В новом подходе, в отличие от RLCD, используется алгоритм ε-жадного Q-обучения [5]. Также как в алгоритме классического Q-обучения, на каждой итерации агент применяет действие a к среде, находящейся в состоянии s. Затем обновляется значение ожидаемой награды $Q(s, a)$. В отличие от классического алгоритма Q-обучения при условии, что $Q(s, a) - Q(s', a') < \delta$ для какой-то пары (s, a) и (s', a'), обучение начинается заново. Перезапуск обучения связан с тем, что в описанном случае ожидаемая награда примерно одинакова для хотя бы одной пары действий и агент не может определить, какое из них более эффективно.

Модельная задача

Рассмотрим постановку модельной задачи с двумя вспомогательными критериями, которые могут быть как эффективными так и неэффективными на разных этапах оптимизации. В этой задаче особи представляются битовыми строками длины n. Пусть x — число бит, равных единице. Целевая

ФП задается формулой $g(x)=\left\lfloor \frac{x}{k} \right\rfloor$, где k — константа, $k<n$. Необходимо максимизировать значение целевой ФП. Вспомогательные ФП имеют следующий вид:

$$h_1=\begin{cases} x,\ x \le p_1 \\ p_1,\ p_1 < x \le p_2 \\ x,\ p_2 < x \le p_3 \\ p_3,\ p_3 < x \le p_4 \\ \dots \\ x,\ p_s < x \le n \end{cases} \qquad h_2=\begin{cases} p_1,\ x \le p_1 \\ x,\ p_1 < x \le p_2 \\ x,\ p_3 < x \le p_3 \\ p_3,\ x < x \le p_4 \\ \dots \\ n,\ p_s < x \le n \end{cases}.$$

В точках p_i вспомогательные ФП меняют свои свойства, будем называть их *точками переключения*. Вспомогательный критерий h_1 эффективен, когда $x \in [0, p_1]$, $(p_2, p_3]$, …, $(p_s, n]$, а h_2 эффективен во всех других случаях. Отметим, что использование правильного вспомогательного критерия позволяет различить особи с одинаковым значением целевой ФП, и выбрать ту, в которой содержится большее число единиц. Такая особь с большей вероятностью породит особь с более высоким значением целевой ФП.

Описание эксперимента

В ходе экспериментов сравнивались результаты применения метода EA+RL с использованием различных алгоритмов обучения с подкреплением к различным конфигурациям модельной задачи. Результаты работы каждого алгоритма усреднялись за 100 запусков. Рассматривались конфигурации задачи с пятью и десятью точками переключения. Точки переключения располагались равномерно по длине особи. В качестве параметра k были выбраны значения 10 и 25.

Поколение ЭА состояло из 100 особей. Оператор мутации изменял каждый бит с вероятностью 0.001. Оператор скрещивания[9] применялся с вероятностью 0.7. Выполнение ЭА останавливалось при достижении заданного числа итераций или максимального значения целевой ФП.

В качестве алгоритмов обучения с подкреплением использовались ε-жадное Q-обучение, отложенное Q-обучение[10] и предлагаемый подход. В качестве параметров для ε-жадного Q-обучения использовались: $\alpha=0.6$, $\gamma = 0.01$, $\varepsilon=0.03$ [4]. Отложенное Q-обучение использовалось с параметрами: $\alpha=0.6$, $\gamma=0.01$, $\varepsilon=0.4$, $m=5$ [4]. Параметры для предлагаемого подхода были выбраны в ходе предварительных экспериментов: $\alpha=0.6$, $\gamma=0.01$, $\varepsilon=0.0$ и $\delta=0.001$. Состояния представлялись в виде вектора порядковых номеров ФП, упорядоченных по значению $(f(x_c) — f(x_p))/f(x_c)$, где f — ФП, x_p — число бит, равных единице в лучшей особи предыдущего поколения, x_c — число бит, равных единице в лучшей особи текущего поколения [1].

Результаты экспериментов

Результаты экспериментов представлены в Таблице 1. В первой колонке указано число точек переключения, во второй и третьей — число итераций и длина особи соответственно. В последних трех колонках указаны средние значения целевой ФП, полученные при использовании предлагаемого подхода, ε-жадного Q-обучения и отложенного Q-обучения соответственно. Среднеквадратичное отклонение при использовании первых двух алгоритмов составило около 0.5%, при использовании отложенного Q-обучения около 20%. Для всех рассмотренных конфигураций модельной задачи результаты, полученные при использовании предложенного подхода, превосходят результаты существующих алгоритмов.

Для проверки того, что новый алгоритм отличим от предыдущих, был проведен статистический тест Уилкоксона [11]. Значения *p-value*, полученные при сравнении нового подхода с ε-жадным Q-обучением и отложенным Q-обучением, приведены в скобках в соответствующих колонках. Можно видеть, что в случаях, когда длина особи превышала 1000, *p-value*, полученные при сравнении нового подхода с ε-жадным Q-обучением, малы, что говорит о различимости этих подходов. Однако новый метод не всегда статистически различим с отложенным Q-обучением, несмотря на то, что среднее значение целевой ФП, полученное с помощью нового подхода, значительно выше. Это можно объяснить большим разбросом значений целевой ФП при применении отложенного Q-обучения.

Агент может выбирать на каждой итерации одну из вспомогательных ФП или целевую ФП. Наиболее эффективно выбирать ту ФП, которая в текущем поколении равна x. Будем называть выбор эффективной ФП *хорошим.* На Рис. 1 представлено число выборов ФП в ходе решения модельной задаче с пятью точками переключения, $k=10$, $n=750$. По горизонтали указан номер итерации, а ширина полосы соответствует числу выборов соответствующей

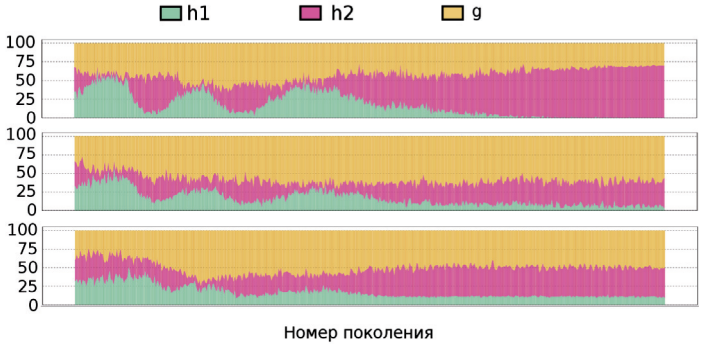


Рис. 1. Число выборов ФП при использовании нового метода (сверху), ε-жадного Q-обучения (в середине), отложенного Q-обучения (внизу)

Таблица 1

Результаты экспериментов

Число p_i	**Число итераций**	**Длина**	**Новый метод**	**ε-жадное Q-обучение**	**Отложенное Q-обучение**
			$k = 10$		
5	3000	750	74,52	74,49 (0,34)	68,39 ($4{,}4\times10^{-10}$)
		1000	99,55	99,47 (0,13)	87,39 ($2{,}2\times10^{-16}$)
		1250	124,44	124,19 ($8{,}3\times10^{-4}$)	113,69 ($2{,}2\times10^{-16}$)
		1500	149,03	148,02 ($1{,}5\times10^{-3}$)	133,5 ($8{,}1\times10^{-15}$)
		1750	173,98	173,63 ($8{,}4\times10^{-8}$)	156,93 ($1{,}9\times10^{-13}$)
	5000	2000	198,93	197,98 ($2{,}2\times10^{-16}$)	186,37 ($9{,}5\times10^{-14}$)
		2250	222,01	220,23 ($7{,}2\times10^{-11}$)	202,80 ($1{,}5\times10^{-3}$)
		2500	245,52	244,55 (3×10^{-3})	232,81 (0,99)
10	5000	2000	198,94	198,34 ($1{,}8\times10^{-12}$)	170,59 ($2{,}7\times10^{-13}$)
		2250	223,36	220,79 ($2{,}2\times10^{-16}$)	184,47 ($1{,}1\times10^{-12}$)
		2500	245,28	244,61 ($1{,}2\times10^{-4}$)	204,42 (0,72)
	9000	2750	269,38	269,14 (5×10^{-3})	226,14 (0,57)
		3000	294,22	293,73 ($2{,}8\times10^{-5}$)	249,96 (0,96)
		3250	318,92	318,70 (0,014)	268,66 (0,97)
		3500	343,79	343,33 (4×10^{-5})	285,76 (0,99)
		3750	368,52	367,90 ($1{,}3\times10^{-5}$)	307,72 (0,99)
			$k = 25$		
5	3000	750	29,45	29,40 (0,25)	26,01 ($2{,}2\times10^{-16}$)
		1000	39,18	39,14 (0,22)	36,20 ($8{,}9\times10^{-14}$)
		1250	49,02	49,00 (0,24)	45,86 ($2{,}8\times10^{-9}$)
		1500	59,00	58,92 (6×10^{-3})	54,77 ($4{,}3\times10^{-8}$)
		1750	68,96	68,02 (4×10^{-15})	61,17 ($1{,}8\times10^{-11}$)
	5000	2000	78,17	77,23 ($4{,}9\times10^{-16}$)	70,54 (0,19)
		2250	87,30	87,12 (8×10^{-3})	79,70 (0,98)
		2500	97,01	96,89 (0,004)	84,62 (0,53)

Таблица 2

Число выборов ФП

Число p_i	Длина	Число хороших выборов, %		
		Новый метод	ε-жадное Q-обучение	Отложенное Q-обучение
5	750	62	50 ($1{,}3\times10^{-5}$)	46 (8×10^{-3})
	1000	55	51 (0,01)	47 (0,26)
	1250	51	43 (7×10^{-4})	36 ($1{,}7\times10^{-5}$)
	1500	50	39 ($2{,}2\times10^{-16}$)	35 ($1{,}01\times10^{-10}$)
	1750	48	39 ($2{,}2\times10^{-16}$)	37 ($9{,}2\times10^{-10}$)
	2000	46	39 ($2{,}2\times10^{-16}$)	29 ($2{,}1\times10^{-15}$)
	2250	37	23 ($2{,}2\times10^{-16}$)	37 ($8{,}5\times10^{-7}$)
	2500	30	17 ($2{,}2\times10^{-16}$)	26 ($4{,}1\times10^{-14}$)
10	2000	47	49 ($1{,}3\times10^{-8}$)	33 ($6{,}9\times10^{-13}$)
	2250	42	29 ($2{,}2\times10^{-16}$)	36 ($3{,}6\times10^{-6}$)
	2500	26	19 ($2{,}2\times10^{-16}$)	38 ($1{,}2\times10^{-4}$)
	2750	26	21 ($2{,}2\times10^{-16}$)	41 ($3{,}9\times10^{-3}$)
	3000	31	26 ($2{,}2\times10^{-16}$)	33 ($5{,}1\times10^{-7}$)
	3250	37	29 ($2{,}2\times10^{-16}$)	36 ($1{,}6\times10^{-5}$)
	3500	41	33 ($2{,}2\times10^{-16}$)	38 ($4{,}6\times10^{-5}$)
	3750	43	35 ($2{,}2\times10^{-16}$)	37 ($4{,}6\times10^{-5}$)

ФП в 100 запусках. Можно заметить, что новый подход делает хороший выбор чаще, чем ε-жадное Q-обучение, и ε-жадное Q-обучение делает хороший выбор чаще, чем отложенное Q-обучение.

В Таблице 2 представлен усредненный процент числа хороших выборов ФП для конфигураций со значением параметра $k=10$. При $k=25$ результаты аналогичны и для краткости не представлены. В первой колонке указано число точек переключения, во второй — длина особи. В последних трех колонках указаны средние значения числа выборов хорошей ФП в процентах от общего числа выборов ФП, полученного при использовании предлагаемого подхода, ε-жадного Q-обучения и отложенного Q-обучения соответственно. Среднеквадратичное отклонение при использовании первых двух алгоритмов составило около 8%, при использовании отложенного Q-обучения — около 100%.

Можно видеть, что предлагаемый подход делает хорошие выборы чаще, чем ε-жадное Q-обучение. Однако существуют конфигурации задачи, на которых отложенное Q-обучение делает больше хороших выборов, чем новый

подход. В то же время среднее значение целевой ФП, полученное при применении отложенного Q-обучения хуже, чем при применении нового метода. Это можно объяснить тем, что среднеквадратичное отклонение для отложенного Q-обучения гораздо больше, чем для нового подхода.

В последних двух колонках Таблицы 2 в скобках указаны результаты сравнения нового метода с соответствующими алгоритмами, проведенного с помощью теста Уилкоксона. Можно видеть, что новый метод отличим от существующих для всех рассмотренных конфигураций задачи.

Заключение

В работе предложен новый подход к обучению с подкреплением, который может быть использован в методе EA+RL. Данный подход применим в случае нестационарности, заключающейся в изменении свойств вспомогательных критериев в зависимости от этапа оптимизации. Предлагаемый подход был применен для решения модельной задачи. Полученные результаты превосходят результаты работы ε-жадного Q-обучения и отложенного Q-обучения.

Литература

1. *Jensen M. T*. Reducing the Run-time Complexity of Multiobjective EAs: The NSGA-II and Other Algorithms. Transactions on Evolutionary Computation. 2003. P. 503–515.
2. *Knowles J. D., Watson R. A., Corne D*. Reducing Local Optima in Single-Objective Problems by Multi-objectivization // In Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization. 2001. P. 269–283.
3. *Lochtefeld D. F., Ciarallo F. W*. Deterministic Helper-Objective Sequences Applied to Job-Shop Scheduling // In Proceedings of Genetic and Evolutionary Computation Conference. 2010. P. 431–438.
4. *Afanasyeva A., Buzdalov M*. Optimization with Auxiliary Criteria using Evolutionary Algorithms and Reinforcement Learning // In Proceedings of 18th International Conference on Soft Computing MENDEL. 2012. P. 58–63.
5. *Sutton R. S., Barto A. G*. Reinforcement Learning: An Introduction // MIT Press, Cambridge, MA, USA. 1998.
6. *Buzdalov M., Buzdalova A., Shalyto A*. A First Step towards the Runtime Analysis of Evolutionary Algorithm Adjusted with Reinforcement Learning // In Proceedings of the International Conference on Machine Learning and Applications. 2013. Vol. 1. P. 203–208.
7. *Granmo O.-C., Berg S*. Solving non-stationary bandit problems by random sampling from sibling kalman filters // In IEA/AIE. 2010. P. 199–208.
8. *B. C. da Silva, Basso E. W., Bazzan A. L. C., Engel P. M*. Dealing with non-stationary environments using context detection // In Proceedings of the 23rd International Conference on Machine Learning, ICML'06. 2006. P. 217–224.
9. *Strehl A. L., Li L., Wiewiora E., Langford J., Littman M. L*. PAC Model-free Reinforcement Learning // In Proceedings of the 23rd International Conference on Machine Learning. 2006. P. 881–888.

10. *Arkhipov V., Buzdalov M., Shalyto A*. Worst-Case Execution Time Test Generation for Augmenting Path Maximum Flow Algorithms using Genetic Algorithms // In Proceedings of the International Conference on Machine Learning and Applications. 2013. Vol. 2. P. 108–111.
11. *Derrac J., Garcia S., Molina D., Herrera F*. A practical tutorial on the use of non-parametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms // Swarm and Evolutionary Computation. 2011. P. 3–18.

СРАВНИТЕЛЬНЫЙ АНАЛИЗ МЕТОДА ВЫБОРА ВСПОМОГАТЕЛЬНЫХ КРИТЕРИЕВ И МЕТОДА СПУСКА СО СЛУЧАЙНЫМИ МУТАЦИЯМИ

М. В. Буздалов

ассистент кафедры компьютерных технологий Университета ИТМО

E-mail: mbuzdalov@gmail.com

Буздалова А. С. студентка кафедры компьютерных технологий Университета ИТМО

E-mail: abuzdalova@gmail.com

Аннотация. Рассматривается ранее предложенный метод выбора вспомогательных критериев оптимизации EA+RL, предназначенный для повышения эффективности эволюционных алгоритмов и основанный на применении обучения с подкреплением. Формулируется модельная задача, на примере которой сравнивается время работы этого метода и метода спуска со случайными мутациями. Приводится схема доказательства, показывающего, что отношение времен работы рассматриваемых методов выражается как экспонента от параметра модельной задачи, что подтверждает эффективность метода EA+RL.

Введение

Эффективность однокритериальной оптимизации в некоторых случаях может быть повышена путем введения вспомогательных критериев [1]. Насколько известно авторам, в последнее время ведутся активные исследования по применению вспомогательных критериев в задачах дискретной оптимизации, решаемых с помощью эволюционных алгоритмов [2–6].

Использование вспомогательных критериев позволяет избежать остановку процесса оптимизации в локальных оптимумах целевого критерия [4, 5], а также расширяет исследуемую область пространства поиска [2], за счет чего оптимальное значение целевого критерия может быть найдено за меньшее число итераций эволюционного алгоритма.

Вспомогательные критерии, как правило, формулируются в ходе анализа задачи. Например, может проводиться декомпозиция целевого критерия на вспомогательные [6, 7]. Также существует пример автоматической генерации вспомогательных критериев для задачи о генерации тестов против решений олимпиадных задач [8].

Обычно об эффективности вспомогательных критериев довольно сложно судить заранее. Более того, один и тот же вспомогательный критерий на различных этапах процесса оптимизации может как ускорять поиск оптимального значения целевого критерия, так и замедлять его [7]. В связи с этим воз-

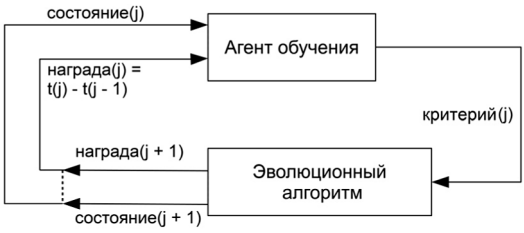


Рис. 1. Метод выбора критериев EA+RL,
t — целевой критерий, j — номер итерации

никает задача автоматического выбора вспомогательного критерия, наиболее эффективного на данном этапе оптимизации, из заранее подготовленного набора критериев.

Автоматический выбор вспомогательных критериев может производиться с помощью ранее предложенного метода EA+RL [9], основанного на выборе критериев оптимизации для эволюционного алгоритма с помощью обучения с подкреплением (см. рис. 1) [10, 11]. Агент обучения на каждой итерации эволюционного алгоритма выбирает критерий оптимизации из списка, состоящего из вспомогательных критериев и целевого. Выбранный критерий используется при формировании очередного поколения эволюционного алгоритма. Затем формируется некоторое представление состояния эволюционного алгоритма, а также награда, зависящая от роста целевого критерия. Награда используется для обновления оценки ожидаемой награды в данном состоянии. Выбирается критерий, максимизирующий оценку ожидаемой награды. В случае, когда оценка одинакова, критерии выбираются равновероятно.

Эффективность метода EA+RL была подтверждена экспериментально на примере решения ряда модельных задач, а также практической задачи генерации тестов [8]. Также существует теоретический результат, показывающий на примере одной модельной задачи, что метод EA+RL позволяет игнорировать неэффективный вспомогательный критерий [12]. Цель данной работы — показать теоретически, что применение метода EA+RL может повысить эффективность оптимизации целевого критерия.

Постановка модельной задачи

Опишем модельную задачу *XdivK* с одним вспомогательным критерием. Пространство поиска — битовые строки длины n. Пусть x — число единиц в битовой строке. Требуется максимизировать целевой критерий:

$$t = \left\lfloor \frac{x(s)}{k} \right\rfloor,$$

где k — целочисленный параметр, $n \bmod k = 0$. В качестве вспомогательного критерия будем использовать значение x.

Заметим, что оптимизация по вспомогательному критерию позволяет достичь оптимума целевого критерия за меньшее число итераций. Пусть есть две битовые строки a и b с числом единиц y и z соответственно, $y < z$ (см. рис. 2). Пусть также с точки зрения целевого критерия эти строки не различаются. Однако использование строки b с большим числом единиц позволяет улучшить значение целевого критерия с большей вероятностью, чем использование строки a. Вспомогательный критерий ускоряет процесс оптимизации целевого критерия за счет того, что он позволяет выбрать строку b.

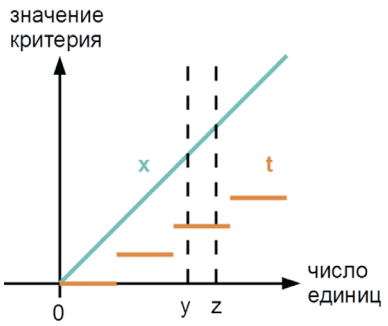


Рис. 2. Целевой и вспомогательный критерии в задаче *XdivK*

Далее на примере описанной модельной задачи будет сравнено время работы метода EA+RL и эволюционного алгоритма, не использующего вспомогательный критерий. Требуется показать, что метод EA+RL позволит использовать преимущества вспомогательного критерия и найти оптимальное значение целевого критерия за меньшее число итераций.

В качестве эволюционного алгоритма будем рассматривать метод спуска со случайными мутациями. В качестве обучения с подкреплением используется алгоритм Q-обучения [10, 11], поддерживающий текущую оценку ожидаемой награды, которая обновляется в соответствии с формулой: $Q(s, f) = Q(s, f) + \alpha(r + \gamma \max_{f'} Q(s', f') - Q(f, a))$, где f — выбранный критерий, состояние s — значение целевого критерия, награда r — разность значений целевого критерия на двух последовательных итерациях. Изначально Q инициализируется нулями.

Анализ времени работы метода EA+RL и метода спуска со случайными мутациями

В данном разделе представлена общая схема доказательства оценки времени работы метода EA+RL при решении рассматриваемой задачи. Полную версию доказательства можно прочитать в статье [13].

Представим процесс оптимизации как цепь Маркова (см. рис. 3). Общий вид цепи одинаков как для метода EA+RL, так и для метода спуска со случайными мутациями. Вершины цепи соответствуют числу единиц в строке. Цветом выделены вершины, в которых x нацело делится на k. Заметим, что из этих вершин отсутствуют переходы в вершины с меньшим значением x, так как эволюционный алгоритм выбирает строки с большим или равным значением текущего критерия.

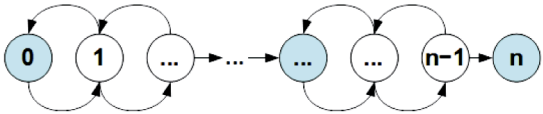


Рис. 3. Процесс оптимизации в виде цепи Маркова

Рассмотрим число итераций, необходимое для перехода из вершины x в вершину $x+1$. Пусть $Z_E(x)$ — математическое ожидание числа итераций метода спуска, $Z_R(x)$ — математическое ожидание числа итераций метода EA+RL. На рис. 4 для метода спуска и метода EA+RL рядом с каждым переходом цепи Маркова подписаны выбранный критерий, мутация и вероятность соответствующего перехода.

Оценим вероятности переходов в случае использования метода спуска (см. рис. 4, слева). Для состояний, в которых x делится нацело на k ($x \bmod k = 0$), выполняется соотношение:

$$Z_E(x) = \frac{n-x}{n} + \frac{x}{n}(1 + Z_E(x)) = \frac{n}{n-x},$$

в то время как для остальных состояний выполняется:

$$Z_E(x) = \frac{n-x}{n} + \frac{x}{n}(1 + Z_E(x-1) + Z_E(x)) = \frac{n}{n-x} + Z_E(x-1)\frac{x}{n-x}.$$

Рассмотрим теперь вероятности переходов в случае использования метода EA+RL, выбирающего на каждой итерации вспомогательный или целевой

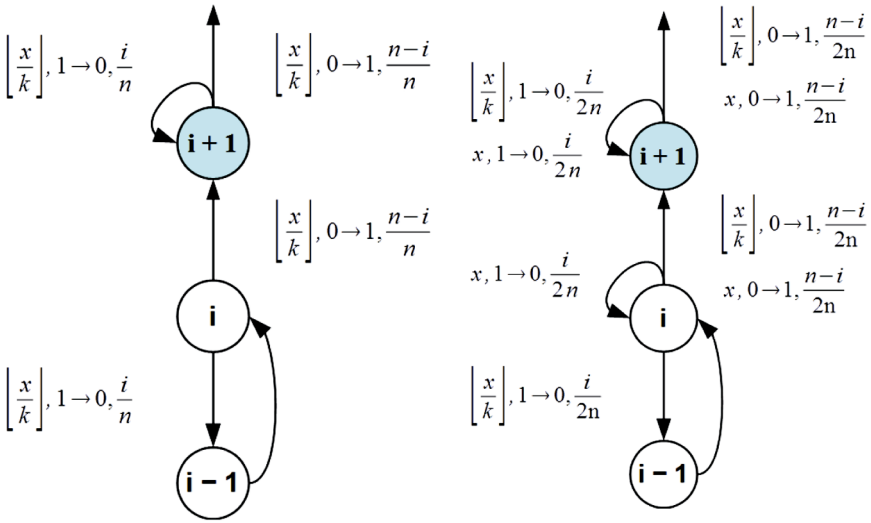


Рис. 4. Вероятности переходов в методе спуска (слева) и в методе EA+RL (справа)

критерий и передающий этот критерий в метод спуска со случайными мутациями (см. рис. 4, справа). В этом случае вероятность перехода в состояние с меньшим числом единиц ниже, так как при выборе вспомогательного критерия переход в строку с меньшим числом единиц осуществляться не будет.

Для состояний, в которых x делится нацело на k ($x \bmod k = 0$), выполняется соотношение:

$$Z_R(x) = \frac{n-x}{n} + \frac{x}{n}(1 + Z_R(x)) = \frac{n}{n-x},$$

для остальных состояний выполняется:

$$Z_R(x) = \frac{n-x}{n} + \frac{x}{2n}(1 + Z_R(x)) + \frac{x}{2n}(1 + Z_R(x-1) + Z_R(x)),$$

$$Z_R(x) = \frac{n}{n-x} + Z_R(x-1)\frac{x}{2(n-x)}.$$

Приведенные выражения представляются как:

$$Z_E(x) = \sum_{i=0}^{x \bmod k} \frac{\binom{n}{x-i}}{\binom{n-1}{x}}, \qquad Z_R(x) = \sum_{i=0}^{x \bmod k} 2^{-i} \frac{\binom{n}{x-i}}{\binom{n-1}{x}},$$

что можно доказать по индукции.

Для вычисления общего числа итераций, необходимого для достижения максимального значения целевого критерия, следует просуммировать полученные выражения вдоль рассмотренной цепи Маркова:

$$T_E(x) = \sum_{x=0}^{n-1} Z_E(x) \quad \text{и} \quad T_R(x) = \sum_{x=0}^{n-1} Z_R(x).$$

Можно показать, что $T_E(n, k) = T_R(n, k) = \Omega(n^k) = O(n^{k+1})$, где k — константа. Также можно показать, что если $n \to \infty$ при фиксированном k, то

$$\frac{T_E(n, k)}{T_R(n, k)} \geq 2^{k-2}(1 - O(1)).$$

Последнее выражение означает, что метод EA+RL позволяет решить модельную задачу *XdivK* не менее, чем в 2^{k-2} раза быстрее, чем метод спуска со случайными мутациями. Был проведен численный эксперимент, который показал, что данную оценку можно попытаться улучшить до 2^{k-1}.

Заключение

Проведен сравнительный анализ времени работы метода выбора вспомогательных критериев оптимизации EA+RL и метода спуска со случайными мутациями. Показано, что метод EA+RL позволяет решить рассмотренную модельную задачу экспоненциально быстрее в зависимости от параметра задачи. В сформулированной задаче присутствует один вспомогательный критерий, эффективный на протяжении всего процесса оптимизации. В рамках дальнейших исследований целесообразно провести анализ времени работы метода EA+RL на примере задачи, где эффективность вспомогательных критериев зависит от этапа оптимизации.

Литература

1. *Евтушенко Ю. Г., Жадан В. Г.* Точные вспомогательные функции в задачах оптимизации // Журнал вычислительной математики и математической физики. № 1. Т. 30. 1990. С. 43–57.
2. *Neumann F., Wegener I.* Can single-objective optimization profit from multiobjective optimization? // Knowles J., Corne D., Deb K., Chair D. (Eds.), Multiobjective Problem Solving from Nature, Natural Computing Series, Springer Berlin Heidelberg. 2008. P. 115–130.
3. *Segura C., Coello C. A. C., Miranda G., Léon C.* Using multi-objective evolutionary algorithms for single-objective optimization / 4OR 11 (3) . 2013. P. 201–228.
4. *Knowles J. D., Watson R. A., Corne D.* Reducing local optima in single-objective problems by multi-objectivization // Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization, EMO'01, Springer-Verlag, London, UK. 2001. P. 269–283.
5. *D. Brockhoff, T. Friedrich, N. Hebbinghaus, C. Klein, F. Neumann, E. Zitzler*. On the effects of adding objectives to plateau functions // IEEE Trans. Evolutionary Computation 13 (3). 2009. P. 591–603.
6. *Handl J., Lovell S., Knowles J.* Multiobjectivization by decomposition of scalar cost functions // Rudolph G., Jansen T., Lucas S., Poloni C., Beume N. (Eds.), Parallel Problem Solving from Nature PPSN X, Vol. 5199 of Lecture Notes in Computer Science, Springer Berlin Heidelberg. 2008. P. 31–40.
7. *Jensen M. T.* Helper-objectives: Using multi-objective evolutionary algorithms for single-objective optimisation: Evolutionary computation combinatorial optimization // Journal of Mathematical Modelling and Algorithms 3 (4). 2004. P. 323–347.
8. *Буздалов М. В.* Генерация тестов для олимпиадных задач по программированию с использованием генетических алгоритмов // Научно-технический вестник СПбГУ ИТМО. 2011. № 2 (72). С. 72–77.
9. *Buzdalova A., Buzdalov M.* Increasing Efficiency of Evolutionary Algorithms by Choosing between Auxiliary Fitness Functions with Reinforcement Learning // Proceedings of the Eleventh International Conference on Machine Learning and Applications, ICMLA 2012. Boca Raton: IEEE Computer Society. Vol. 1. 2012. P. 150–155.
10. *Sutton R. S., Barto A. G.* Reinforcement Learning: An Introduction. MIT Press, Cambridge, MA, 1998. 322 p.

11. *Николенко С. И., Тулупьев А. Л.* Самообучающиеся системы. М.: МЦНМО, 2009. 288 с.
12. *Buzdalov M., Buzdalova A., Shalyto A*. First Step towards the Runtime Analysis of Evolutionary Algorithm Adjusted with Reinforcement Learning // Proceedings of the Twelve International Conference on Machine Learning and Applications, ICMLA 2013. Boca Raton: IEEE Computer Society, 2013. Vol. 1. P. 203–208.
13. *Buzdalov M., Buzdalova A*. OneMax Helps Optimizing XdivK: Theoretical Runtime Analysis for RMHC and EA+RL // Proceedings of the Genetic and Evolutionary Computation Conference «GECCO-2014». 2014 (to be published).

АСИМПТОТИЧЕСКИ ОПТИМАЛЬНЫЕ АЛГОРИТМЫ ДЛЯ ВЫБОРА ВСПОМОГАТЕЛЬНЫХ КРИТЕРИЕВ ОПТИМИЗАЦИИ

М. В. Буздалов

ассистент кафедры компьютерных технологий Университета ИТМО

E-mail: mbuzdalov@gmail.com

А. С. Буздалова

студентка кафедры компьютерных технологий Университета ИТМО

E-mail: abuzdalova@gmail.com

Аннотация. Известны задачи оптимизации, в которых помимо целевого критерия оптимизации существуют другие, дополнительные критерии, оптимизация которых может как ускорить, так и замедлить процесс оптимизации целевого критерия. В данной работе рассматривается случай, когда оптимизация дополнительного критерия может привести к оптимизации целевого критерия.

Предлагается три алгоритма выбора вспомогательных критериев. Данные алгоритмы концептуально схожи, однако требуют различных объемов дополнительной памяти, а также различных интерфейсов к алгоритму оптимизации. Доказывается, что число итераций алгоритма оптимизации до достижения оптимума целевого критерия во всех случаях не превышает $C(K+1)\min_O E_O$, где C — некоторая константа, K — число дополнительных критериев, E_O — матожидание числа итераций до достижения оптимума целевого критерия при оптимизации по критерию O.

Введение

Однокритериальная оптимизация часто может быть ускорена за счет введения дополнительных критериев [1–3]. Известные подходы включают в себя сведение к многокритериальной оптимизации (англ. multi-objectivization [1]), реализуемое как оптимизация одновременно всех дополнительных критериев с использованием многокритериального алгоритма оптимизации, а также подход, называемый использованием вспомогательных критериев (англ. helper-objectives [3]), сводящийся к тому, что целевой критерий оптимизируется одновременно с одним или несколькими дополнительными критериями, причем набор критериев изменяется в ходе оптимизации.

Указанные подходы рассчитаны на использование с дополнительными критериями, которые специально создаются для ускорения процесса оптимизации целевого критерия (и поэтому их часто называют вспомогательными). Однако, это условие может не выполняться, особенно если свойства дополнительных критериев неизвестны, в частности, если они создаются автома-

тически. Для преодоления этого недостатка был разработан метод EA+RL [2], в котором в качестве алгоритма оптимизации обычно используется однокритериальный эволюционный алгоритм, а оптимизируемый критерий выбирается из множества, состоящего из целевого и дополнительных критериев, на основании информации об изменении значения целевого критерия с помощью алгоритма обучения с подкреплением (англ. reinforcement learning, RL).

Настоящая работа посвящена рассмотрению случая, когда оптимум целевого критерия может быть достигнут путем оптимизации одного из дополнительных критериев. Предлагается три алгоритма, управляющих одним или несколькими экземплярами алгоритма оптимизации, для которых число итераций алгоритма оптимизации до достижения оптимума целевого критерия во всех случаях не превышает $C(K+1)\min_O E_O$, где C — некоторая константа, K — число дополнительных критериев, E_O — матожидание числа итераций до достижения оптимума целевого критерия при оптимизации по критерию O. Это позволяет заявить об их «асимптотической оптимальности» — при условии фиксированного K и зависимости E_O от размера задачи, число итераций для достижения целевого критерия будет иметь асимптотическую сложность, равную наименьшей асимптотической сложности среди всех E_O.

Предлагаемые алгоритмы

Зафиксируем используемый однокритериальный итеративный алгоритм оптимизации. Пусть имеется K дополнительных критериев оптимизации. Пусть математическое ожидание числа итераций для каждого критерия оптимизации O (включая целевой и вспомогательные) равно E_O (если матожидание не существует, предполагаем, что оно равно бесконечности). В настоящей работе предлагается три алгоритма, осуществляющих управление одним или несколькими экземплярами алгоритма оптимизации в части их инициализации, выбора критерия оптимизации и запуска итерации.

Алгоритм 1

В данном алгоритме используется $K+1$ экземпляр алгоритма оптимизации, по одному экземпляру на каждый критерий (целевой и дополнительные). Управление этими экземплярами алгоритма осуществляется следующим образом: сначала все они осуществляют по очереди первую итерацию, затем вторую и так далее. Как только в каком-либо экземпляре достигнут оптимум целевого критерия, процесс оптимизации останавливается и возвращается полученное значение.

Нетрудно показать, что матожидание общего числа итераций экземпляров алгоритма оптимизации, необходимых для достижения оптимума целевого критерия, не превышает $(K+1)\min_O E_O$.

Недостатком этого алгоритма является то, что для его работы требуется запуск $K+1$ экземпляра алгоритма оптимизации, а значит, в $K+1$ раз больше памяти. С указанным недостатком справляется алгоритм 2.

Алгоритм 2

В отличие от предыдущего алгоритма, в данном алгоритме используется один экземпляр алгоритма оптимизации. Управление алгоритмом оптимизации осуществляется по *мета-итерациям* следующим образом:

- на нулевой мета-итерации каждый из критериев оптимизации оптимизируется в течение одной итерации;
- на первой мета-итерации каждый из критериев оптимизации оптимизируется в течение двух итераций;
- на t-ой мета-итерации каждый из критериев оптимизации оптимизируется в течение 2^{t-1} итераций.

Перед каждым переключением критериев оптимизации запускается инициализация алгоритма оптимизации (можно понимать как перезапуск указанного алгоритма). При обнаружении оптимума целевого критерия, процесс оптимизации останавливается и возвращается полученное значение.

Для данного алгоритма можно доказать, что матожидание общего числа итераций экземпляров алгоритма оптимизации, необходимых для достижения оптимума целевого критерия, не превышает $8\,(K+1)\,\min_O E_O$.

Недостатком данного алгоритма является то, что он должен обладать возможностью вызывать инициализацию, или перезапуск, алгоритма оптимизации. Это не всегда возможно. Так, например, управляющий алгоритм в методе EA+RL [2] может лишь изменить оптимизируемый критерий. Данный недостаток учитывается в алгоритме 3.

Алгоритм 3

Единственным отличием алгоритма 3 от алгоритма 2 является то, что при переключении критериев инициализация алгоритма оптимизации не производится. Это позволяет использовать алгоритм 3 в качестве «агента» обучения с подкреплением в методе EA+RL.

Однако, про данный алгоритм можно доказать лишь следующее: матожидание общего числа итераций экземпляров алгоритма оптимизации, необходимых для достижения оптимума целевого критерия, не превышает $8\,(K+1)\,\min_O F_O$, где F_O — матожидание числа итераций, требуемых для нахождения оптимума целевого критерия при оптимизации по критерию O, наибольшее по всем возможным состояниям алгоритма оптимизации.

Данное условие можно интерпретировать следующим образом. Рассмотрим в качестве примера алгоритм дифференциальной эволюции [4]. Данный алгоритм не может найти решение задачи оптимизации, если в одном поко-

лении нет четырех различных особей, и будет испытывать существенные затруднения, если расстояния между всеми особями мало. В случае использования алгоритма 2 для управления алгоритмом дифференциальной эволюции, процедура инициализации может всегда генерировать различающихся особей, и алгоритм не будет испытывать затруднений. Однако при использовании алгоритма 3 дифференциальная эволюция может быть приведена в состояние с мало различающимися особями при оптимизации по предыдущему критерию, что существенно затруднит оптимизацию по текущему критерию.

Заключение

В данной работе представлены три алгоритма, позволяющих эффективно производить оптимизацию целевого критерия при наличии дополнительных критериев. Алгоритмы обладают следующими свойствами:

1. Алгоритм 1 использует $K+1$ экземпляр алгоритма оптимизации, где K — число дополнительных критериев. Совокупное число итераций всех экземпляров не превышает $(K+1)\ \min_O E_O$, где E_O — матожидание числа итераций при оптимизации по критерию O.
2. Алгоритм 2 использует один экземпляр алгоритма оптимизации. Число итераций этого алгоритма не превышает $8(K+1)\ \min_O E_O$.
3. лгоритм 3 использует один экземпляр алгоритма оптимизации и осуществляет только переключение критериев, но не перезапуск алгоритма оптимизации (а следовательно, может быть применен в методе EA+RL). Число итераций алгоритма оптимизации не превышает $(K+1)\ \min_O F_O$, где F_O — матожидание числа итераций при оптимизации по критерию O, наибольшее по всем возможным состояниям алгоритма оптимизации.

Рекомендации по использованию того или иного алгоритма зависят от условий поставленной задачи. Например, если большой объем дополнительной памяти менее важен, чем скорость оптимизации, имеет смысл использовать алгоритм 1. Если же цель исследования состоит в сравнении, например, различных алгоритмов обучения с подкреплением в рамках метода EA+RL, в качестве такого алгоритма для сравнения можно взять алгоритм 3.

На практике, алгоритм 3 может работать как хуже, так и лучше алгоритма 2, что зависит от свойств дополнительных критериев. Так, если оптимизация по каждому из дополнительных критериев улучшает и целевой, и все вспомогательные критерии, то ожидается, что алгоритм 3 будет работать лучше алгоритма 2.

Литература

1. *Knowles J. D., Watson R. A., Corne D.* Reducing Local Optima in Single-Objective Problems by Multi-Objectivization // Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization EMO'01. London, UK: Springer-Verlag. 2001. P. 269–283.

2. *Buzdalova A., Buzdalov M*. Increasing Efficiency of Evolutionary Algorithms by Choosing between Auxiliary Fitness Functions with Reinforcement Learning // Proceedings of the Eleventh International Conference on Machine Learning and Applications, ICMLA 2012. Boca Raton: IEEE Computer Society, 2012. Vol. 1. P. 150–155.
3. *Jensen M. T*. Helper-objectives: Using multi-objective evolutionary algorithms for single-objective optimization / J. Math. Model. Algorithms 3(4). 2004. P. 323–347.
4. *Storn R., Price K*. Differential Evolution — a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces // Journal of Global Optimization. 1997. Vol. 11, No. 4. Pp. 341–359.

ПРИМЕНЕНИЕ МЕТОДА НАРУШЕНИЯ СИММЕТРИИ В АЛГОРИТМАХ ПОСТРОЕНИЯ УПРАВЛЯЮЩИХ КОНЕЧНЫХ АВТОМАТОВ[1]

Д. С. Чивилихин
аспирант кафедры компьютерных технологий Университета ИТМО
E-mail: chivdan@rain.ifmo.ru

В. И. Ульянцев
аспирант кафедры компьютерных технологий Университета ИТМО
E-mail: ulyantsev@rain.ifmo.ru

А. А. Шалыто
д.т.н., профессор, заведующий кафедрой технологий программирования Университета ИТМО
E-mail: shalyto@mail.ifmo.ru

Аннотация. Предложен способ применения метода нарушения симметрии (symmetry breaking) в алгоритмах построения управляющих конечных автоматов. Проведенные эксперименты с генетическим и муравьиным алгоритмами построения автоматов показали, что предложенный подход позволяет существенно увеличить эффективность этих алгоритмов.

Введение

Построение управляющих конечных автоматов является одной из основных задач, возникающих при применении автоматного программирования [1]. Для построения автоматов часто применяются метаэвристические алгоритмы, осуществляющие направленный перебор решений. Качество каждого промежуточного решения оценивается с помощью так называемой функции приспособленности (ФП).

Существенной сложностью при применении метаэвристических алгоритмов к построению автоматов является то, что пространство поиска содержит большое число решений, идентичных с точностью до изоморфизма. Это означает, что для любого автомата с N_{states} состояниями существует $N_{\text{states}}! - 1$ других автоматов, имеющих такую же структуру и отличающихся лишь нумерацией состояний. Проблема заключается в том, что хотя все эти автоматы будут иметь одинаковое значение ФП, алгоритму построения автоматов в процессе поиска в худшем случае придется вычислять это значение для каждого автомата в отдельности.

[1] Работа выполнена при государственной финансовой поддержке ведущих университетов Российской Федерации (субсидия 074-U01) и при частичной финансовой поддержке РФФИ в рамках научного проекта № 14-01-00551 а.

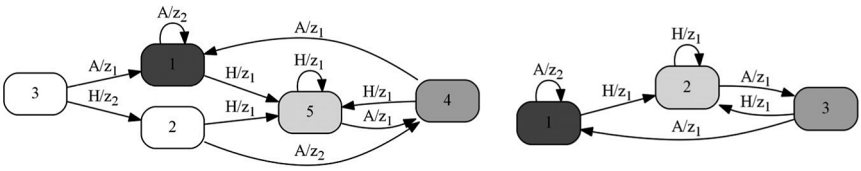


Рис. 1. Пример конечного автомата (а) и его BFS-представления (б), одинаковые состояния помечены одним цветом

Нарушение симметрии в конечных автоматах

Для решения указанной проблемы в данной работе предлагается применить метод нарушения симметрии и специальный кэш автоматов. Используемый метод нарушения симметрии похож на метод MTF (Move-To-Front), предложенный в работе [2]. Идея метода заключается в приведении нумерации состояний автоматов к некому единому виду. Для этого из начального состояния автомата запускается обход в ширину (BFS), при этом порядок обхода детей каждой вершины (состояния) определяется списком входных событий, отсортированным в лексикографическом порядке. Предлагаемое представление автомата, которое мы будем называть BFS-представлением, отличается от метода MTF тем, что недостижимые состояния и переходы в BFS-представлении отбрасываются, в то время как в методе MTF они помещаются в конец строки, кодирующей автомат. Пример конечного автомата и его BFS-представления приведен на Рис. 1. BFS-представление обладает существенным и полезным свойством: все изоморфные друг другу автоматы сводятся к одному автомату в BFS-представлении.

Способы применения BFS-представления в алгоритмах построения автоматов

Существует, по крайней мере, два способа применения BFS-представления автоматов для повышения производительности алгоритмов их построения. Рассмотрим в качестве примера генетический алгоритм. В первом способе все автоматы в популяции должны находиться в BFS-представлении. Потенциально такой подход уменьшает размер пространства поиска в $N_{\text{states}}!$ раз, однако предварительные эксперименты не выявили существенного улучшения производительности алгоритмов при его применении.

Вторым способом применения BFS-представления является введение специального множества автоматов, хранящихся в BFS-представлении. Мы будем называть это множество BFS-кэшом. BFS-кэш хранится в виде хэш-таблицы, в которой ключами являются автоматы в BFS-представлении, а значениями — значения ФП этих автоматов. Когда алгоритм построения

автоматов (например, генетический) получает новый автомат A, производится вычисление его BFS-представления A_{bfs}.

Далее проверяется наличие автомата A_{bfs} в BFS-кэше. Если такой автомат содержится в кэше, то значение ФП автомата A не вычисляется, а берется из кэша. В противном случае значение ФП вычисляется и добавляется в BFS-кэш вместе с автоматом A_{bfs}. Эксперименты показали, что «попадания» в кэш довольно локальны, поэтому его размер поддерживается равным 1000: при превышении данного значения более старые автоматы удаляются из кэша. Таким образом, мы почти никогда не будем дважды вычислять значение ФП изоморфных автоматов.

Эксперименты

В экспериментальном исследовании рассматривалась задача построения управляющих конечных автоматов по сценариям и темпоральным формулам на языке Linear Temporal Logic (LTL). Полное описание этой задачи можно найти в [3]. Было рассмотрено два алгоритма — генетический [3] и муравьиный [4]. Перед запуском экспериментов была проведена автоматическая настройка параметров этих алгоритмов с помощью пакета *irace* [5].

Рассматривались автоматы, содержащие от 4 до 10 состояний. Для каждого значения числа состояний было сгенерировано по 100 автоматов, по каждому из них был построен набор тестовых сценариев и набор LTL формул. В каждом случае генетическому и муравьиному алгоритму разрешалось произвести не более 100000 вычислений ФП. При этом каждый алгоритм запускался как с применением BFS-кэша, так и без него. Для сравнения эффективности алгоритмов использовалась доля успешных запусков, которая вычислялась для каждого значения числа состояний. Запуск считался успешным, если в его результате был построен автомат, удовлетворяющий всем сценариям работы и LTL формулам.

На Рис. 2, *а* приведены графики доли успешных запусков генетического алгоритма с BFS-кэшом и без него, соответствующие графики для муравьиного алгоритма приведены на Рис. 2, *б*. Из рисунков видно, что применение BFS-кэша позволяет увеличить производительность как генетического, так и муравьиного алгоритмов. В результате применения BFS-кэша доля успешных запусков генетического алгоритма увеличилась на 20–120%, а эффективность муравьиного алгоритма возросла на 5–240%.

Для проверки статистической значимости полученных результатов был применен тест Вилкоксона [6]. Для генетического алгоритма значения p-value лежат в интервале от 8.97×10^{-5} ($N_{\text{states}} = 9$) до 0.02 ($N_{\text{states}} = 4$). Значения p-value для муравьиного алгоритма меньше 0.05 для $N_{\text{states}} = 6,8,9,10$, а для других значений числа состояний статистической значимости не наблюдается. Для случаев четырех и пяти состояний это неудивительно, ведь доля успешных запусков муравьиного алгоритма без кэша здесь близка к 100%.

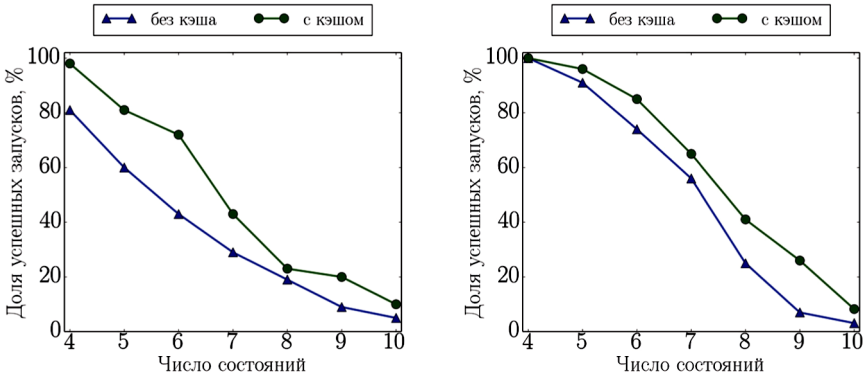


Рис. 2. Доля успешных запусков генетического (а) и муравьиного (б) алгоритмов с использованием BFS-кэша и без него

Заключение

В данной работе предложен способ применения метода нарушения симметрии в задачах построения управляющих конечных автоматов. Экспериментальное исследование показало, что предложенная эвристика существенно улучшает эффективность генетического и муравьиного алгоритмов построения управляющих конечных автоматов по сценариям работы и темпоральным формулам.

Литература

1. *Поликарпова Н. И., Шалыто А. А.* Автоматное программирование. 2011. СПб.: Питер. 176 с.
2. *Chambers D. L.* Handbook of Genetic Algorithms: Complex Coding Systems (Volume III). CRC, 1999.
3. *Tsarev F., Egorov K.* Finite state machine induction using genetic algorithm based on testing and model checking // In Proceedings of the 13th annual conference companion on Genetic and evolutionary computation (GECCO'11). New York, NY, USA: ACM, 2011, pp. 759–762.
4. *Chivilikhin D., Ulyantsev V.* MuACOsm: A new mutation-based ant colony optimization algorithm for learning finite-state machines // In Proceedings of the fifteenth annual conference on Genetic and evolutionary computation (GECCO'13). New York, NY, USA: ACM, 2013, pp. 511–518.
5. *López-Ibáñez M., Dubois-Lacoste J., Stützle T., Birattari M.* The irace package, iterated race for automatic algorithm configuration // IRIDIA, Université Libre de Bruxelles, Belgium, Tech. Rep. TR/IRIDIA/2011–004, 2011.
6. *Wilcoxon F.* Individual comparisons by ranking methods // Biometrics Bulletin, 1(6):80–83, 1945.

АВТОМАТИЗИРОВАННОЕ ПОСТРОЕНИЕ УПРАВЛЯЮЩИХ АВТОМАТОВ В СРЕДЕ STATEFLOW ПРИ ПОМОЩИ МЕТОДОВ МАШИННОГО ОБУЧЕНИЯ[1]

Н. В. Ведерников
студент кафедры компьютерных технологий Университета ИТМО
E-mail: vedernikovnv@gmail.com

В. Ю. Демьянюк
студент кафедры компьютерных технологий Университета ИТМО
E-mail: chavit92@gmail.com

П. В. Кротков
студент кафедры компьютерных технологий Университета ИТМО
E-mail: krotkov.pavel@gmail.com

В. И. Ульянцев
аспирант кафедры компьютерных технологий Университета ИТМО
E-mail: ulyantsev@rain.ifmo.ru

А. А. Шалыто
зав. каф. технологий программирования Университета ИТМО
E-mail: shalyto@mail.ifmo.ru

Аннотация. В последнее время при проектировании систем управления все чаще применяются управляющие конечные автоматы. Так, все более широкое распространение получают высокоуровневые средства проектирования управляющих систем, ключевыми элементами в которых являются управляющие автоматы. Построение данных автоматов производится специалистом вручную, после чего генерация кода и программирование целевого устройства производятся автоматизировано.

Настоящая работа направлена на повышение уровня автоматизации и уменьшение влияния человеческого фактора на этапе построения управляющих автоматов в указанных средствах проектирования. В качестве примера проводится автоматизированное построение управляющих автоматов среды *Stateflow*, входящей в пакет *MATLAB/Simulink*, по экспертным сценариям работы при помощи существующих методов машинного обучения.

[1] Работа выполнена при государственной финансовой поддержке ведущих университетов Российской Федерации (субсидия 074-U01), при поддержке РФФИ в рамках научного проекта № 14-07-31337 мол_а.

Введение

Для построения управляющих систем, как в индустрии, так и в некоммерческих целях все чаще применяются среды высокоуровневого проектирования. Примерами таких сред являются *MATLAB/Simulink*[1], *LabVIEW*[2], *VisSim*[3]. После визуального моделирования управляющих автоматов в данных средах можно провести автоматизированную симуляцию работы созданной реактивной системы, автоматически преобразовать ее в код для множества целевых платформ на таких языках, как *C*, *C++*, *VHDL*, *Verilog*.

Перечисленные среды предоставляют возможность проектировать управляющие элементы в виде взаимодействующих управляющих автоматов, что соответствует парадигме автоматного программирования, при использовании которой программа или ее фрагмент осмысливается как модель какого-либо формального автомата [1]. Основными преимуществами такого подхода являются наглядность представления логики системы со сложным поведением, а также возможность формальной верификации системы [2], которая затруднительна при применении других подходов к ее программированию.

Для решения многих задач управляющие автоматы удается построить эвристически [3], однако существуют задачи [4, 5], для которых построение автоматов вручную затруднительно. Для автоматизированного построения управляющих автоматов, являющихся решениями таких задач, в последнее время разработано множество методов машинного обучения. Входными данными для данных методов являются или экспертные данные, накладывающие ограничения на идентифицируемую систему, или функция приспособленности управляющего автомата.

На данный момент авторам не известны примеры применения указанных методов машинного обучения для автоматизированного построения управляющих автоматов в средах высокоуровневого проектирования. **Целью настоящей работы** является демонстрация возможности применения разработанных методов для автоматизации проектирования управляющих автоматов в данных средах. Авторами разработано программное средство, позволяющее идентифицировать, загрузить и корректно отобразить управляющий автомат в среде *Stateflow*, входящей в пакет *MATLAB/Simulink*.

Методы машинного обучения для построения управляющих конечных автоматов

Управляющим конечным автоматом называется детерминированный конечный автомат, каждый переход которого помечен событием, последовательностью выходных воздействий и охранным условием, представляющим

[1] http://www.mathworks.com/products/simulink/
[2] http://sine.ni.com/np/app/main/p/docid/nav-104/lang/ru
[3] http://www.vissim.com/

собой логическую формулу от входных переменных [1]. Автомат получает события от так называемых *поставщиков событий* (в их роли могут выступать внешняя среда, интерфейс пользователя и т.д.) и генерирует выходные воздействия для объекта управления. При поступлении события автомат выполняет переход в соответствии с охранными условиями и значениями входных переменных. При выполнении перехода генерируются выходные воздействия, которыми он помечен, и автомат переходит в соответствующее состояние. Пример управляющего автомата с двумя состояниями приведен на рис. 1.

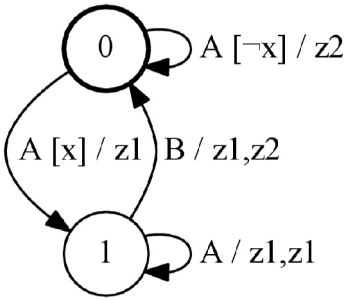


Рис. 1. Пример управляющего конечного автомата

Вкратце опишем известные подходы к построению конечных автоматов. Отметим, что в общем случае решаются *NP*-трудные задачи построения автоматов, для которых актуальна разработка новых алгоритмов решения. Для построения управляющих автоматов по заданной функции приспособленности используются такие эволюционные алгоритмы, как: эволюционные стратегии [6], генетические алгоритмы [4], а также специализированные муравьиные алгоритмы [5].

В работе [7] решается частная задача идентификации управляющих конечных автоматов по экспертным данным, заданным в виде *сценариев работы программы*.

В основе данного метода лежит сведение к задаче о выполнимости булевой формулы (*SAT*), а скорость его работы на несколько порядков превышает скорость работы методов, основанных на эволюционных алгоритмах. При этом метод гарантирует нахождение решения, в случае его существования, в отличие от методов, основанных на эволюционных алгоритмах.

В настоящей работе в качестве метода машинного обучения используется последний из описанных, основанный на методах решения задачи о выполнимости булевой формулы. При помощи метода решается задача построения управляющего конечного автомата, удовлетворяющего следующим требованиям:

- число состояний автомата равно заданному числу C;
- автомат удовлетворяет заданному множеству сценариев работы S;
- каждый переход автомата подтвержден хотя бы одним сценарием работы.

После работы программного средства, реализующего метод, в случае существования решения будет выведен управляющий автомат в формате *DOT*[1].

[1] http://www.graphviz.org/content/dot-language

Применение метода машинного обучения для построения моделей среды Stateflow

Преимущества использования управляющих автоматов привели к появлению множества специализированных пакетов или полноценных программных продуктов, предназначенных для визуального построения управляющих автоматов. Одним из наиболее распространенных пакетов является интерактивная среда *Stateflow*, являющаяся частью программного пакета *MATLAB/Simulink*. Среда *Stateflow* предоставляет большой набор разнообразных инструментов, позволяющих спроектировать автоматную модель, промоделировать ее работу на тех или иных входных данных, проверить ее работоспособность с помощью тестирования, преобразовать автоматную модель в программный код и интегрировать его в реальную среду. Опишем процесс построения управляющих автоматов для данной среды — после построения управляющего автомата в формате *DOT* преобразуем его во внутренний формат *Stateflow*.

В качестве целевого формата, в который будет преобразовываться построенный с помощью методов машинного обучения автомат, выбран формат *MDL*, являющийся внутренним форматом программного комплекса *MATLAB/Simulink*.

Основное отличие *MDL* от других форматов хранения данных, используемых средой *MATLAB*/*Simulink*, является представление данных в текстовом, понятном человеку формате. По сути, описание модели, сохраненное в формате *MDL*, является списком всех объектов, участвующих в модели, вместе с перечислением их параметров и значений этих параметров. Таким образом, данный формат является наиболее удобным для автоматического преобразования построенных моделей систем управления.

Множества событий, выходных воздействий и входных переменных извлекаются из сценариев работы, поданных на вход методу машинного обучения. Данные объекты сохраняют свою суть и свойства и при преобразовании их в целевой формат *MDL*.

Формат *MDL* предполагает закрепление на плоскости каждого выводимого на экран элемента. Поэтому наибольшую сложность при преобразовании построенной модели в формат *MDL* представляет собой укладка на плоскости ориентированного графа с пометками на ребрах — именно так представляется управляющий автомат. Главное требование к представлению полученного изображения автомата заключается в том, что по его внешнему виду пользователь (архитектор управляющей системы), должен понять его общую структуру. Это, в свою очередь, необходимо для того, чтобы пользователь мог внести какие-либо модификации в структуру автомата, или же промоделировать его работу на различных входных данных и сделать выводы о закономерностях, которые он наблюдает в процессе работы автомата.

Проблема изображения непланарных графов на плоскости также достаточно хорошо изучена, существуют различные технические решения этой

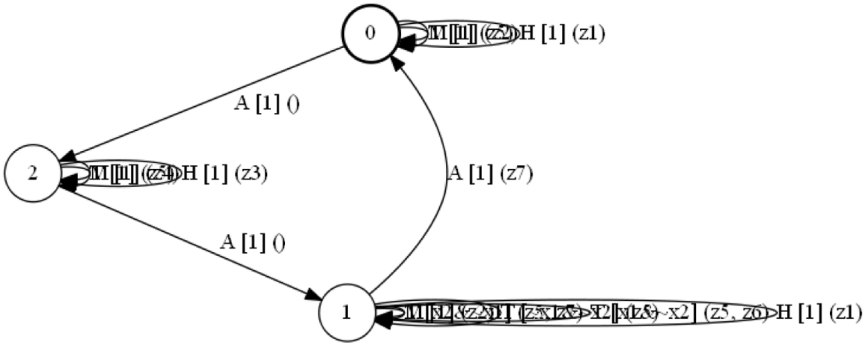


Рис. 2. Пример укладки автомата на плоскость, полученной в результате запуска программы *Graphviz*

проблемы. Одним из наиболее известных программных продуктов, реализующим наиболее качественные из этих решений, является программный продукт *Graphviz* — открытый пакет утилит для визуализации графов.

К сожалению, результат работы *Graphviz* на управляющих автоматах, полученных описанным выше методом, может оказаться неудовлетворительным с точки зрения наглядности управляющей системы. Одна из специфичных черт, которыми обладают полученные управляющие — наличие большого числа петель (ребер, ведущих из вершины в саму себя). В предлагаемых *Graphviz* вариантах расположения графов на плоскости пометки на ребрах-петлях перекрываются друг с другом (рис. 2). При этом расположение остальных вершин, ребер и пометок является вполне наглядным и может быть использовано для визуального представления автомата.

Формат данных *EPS*, являющийся выходным для *Graphviz*, является хорошо документированным форматом хранения изображений в векторном виде, вследствие чего он хорошо подается синтаксическому разбору. Итоговый алгоритм преобразования построенной автоматной модели системы управления из формата *DOT* в формат *MDL* выглядит следующим образом:

- конвертация множеств событий, входных переменных и выходных воздействий;
- построение изображения модели на плоскости с помощью *Graphviz*;
- распознавание и модификация построенного изображения в формате *EPS*: расположение ребер-петель для наглядного представления автомата;
- сохранение управляющего автомата в работающую модель *MATLAB/Simulink* (формат *MDL*) с указанием построенных координат для изображения на плоскости.

Описанный алгоритм был реализован на языке программирования *Java*. Откроем полученный в результате работы программы *MDL*-файл в среде *Stateflow*. Полученное представление управляющего автомата в среде при-

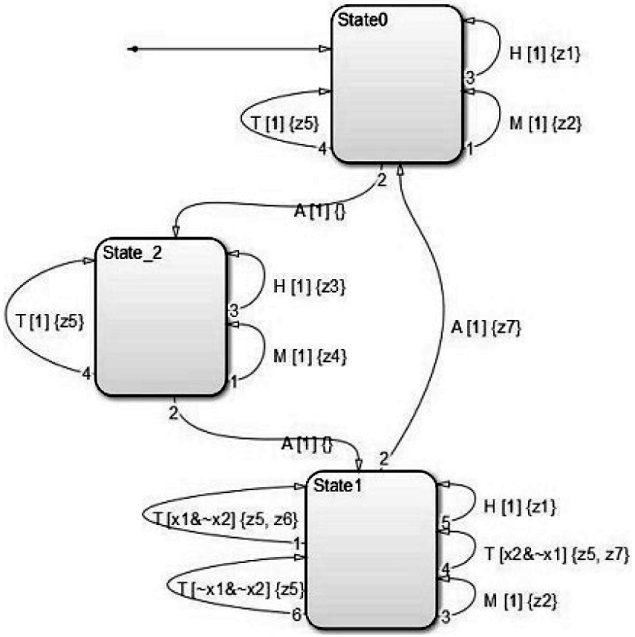


Рис. 3. Представление построенного управляющего автомата в среде *Stateflow* (фрагмент изображения экрана)

ведено на рис. 3. Сравнив рис. 2 и рис. 3 несложно заметить, что проделанные преобразования повысили наглядность управляющего автомата.

Разработанный метод возможно применять для преобразования построенных управляющих автоматов с большим числом состояний и переходов, граница применимости значительно превосходит границу применимости рассматриваемых методов машинного обучения (20–30 состояний).

Заключение

В настоящей работе рассмотрена возможность применения методов машинного обучения для автоматизированного построения управляющих автоматов в высокоуровневых средствах проектирования систем. В качестве примера метода машинного обучения используется метод, описанный в [7], а в качестве высокоуровневого средства проектирования систем управления рассматривается среда *Stateflow*, входящая в *MATLAB/Simulink*. Разработан алгоритм, позволяющий загрузить полученный при помощи метода машинного обучения управляющий автомат в среду проектирования. Разумеется, архитектор системы управления волен выбирать для решения задачи как метод машинного обучения, так и средство проектирования.

Таким образом, автоматизацию процесса программирования целевой системы можно поднять на еще одну ступень — у архитектора имеется возможность взамен эвристического построения управляющего автомата задать примеры его поведения, а построенный автомат при желании модифицировать. В дальнейшем, к примеру, среда *Stateflow* предоставляет широкие возможности для моделирования условий, в которых управляющие автоматы будут находиться. Пользователь может поместить построенную модель системы управления в созданную им модель окружающей среды, после чего проверить работоспособность готовой модели при помощи моделирования и тестирования.

Предложенный подход также можно использовать для построения моделей существующих систем управления со сложным поведением. Для этого можно по существующей системе построить сценарии ее корректного поведения в различных ситуациях, после чего подать на вход указанному методу машинного обучения. После этого можно построенный управляющий автомат преобразовать при помощи разработанного в настоящей работе алгоритма в формат *MDL*, открыть при помощи *Stateflow* и провести полноценное тестирование построенной модели.

Литература

1. *Поликарпова Н. И., Шалыто А. А*. Автоматное программирование. 2-е изд. СПб.: Питер, 2011.
2. *Вельдер С. Э., Лукин М. А., Шалыто А. А., Яминов Б. Р*. Верификация автоматных программ. СПб.: Наука, 2011. 242 с.
3. *Янкин Ю. Ю., Шалыто А. А*. Автоматное программирование ПЛИС в задачах управления электроприводом // Информационно-управляющие системы. 2011. № 1. С. 50–56.
4. *Александров А. В., Казаков С. В., Сергушичев А. А., Царев Ф. Н., Шалыто А. А*. Применение эволюционного программирования на основе обучающих примеров для генерации конечных автоматов, управляющих объектами со сложным поведением // Известия РАН. Теория и системы управления. 2013. № 3. С. 85–100
5. *Chivilikhin D., Ulyantsev V*. Learning Finite-State Machines with Ant Colony Optimization // Lecture Notes in Computer Science. 2012. Vol. 7461/2012. Pp. 268–275.
6. *Chivilikhin D., Ulyantsev V., Tsarev F*. Test-Based Extended Finite-State Machines Induction with Evolutionary Algorithms and Ant Colony Optimization // Proceedings of the 2012 GECCO Conference Companion on Genetic and Evolutionary Computation. NY.: ACM. 2012. Pp. 603–606
7. *Ulyantsev V., Tsarev F*. Extended Finite-State Machine Induction using SAT-Solver // Proceedings of the Tenth International Conference on Machine Learning and Applications, ICMLA 2011, Honolulu, HI, USA, 18–21 December 2011. IEEE Computer Society, 2011. Vol. 2. Pp. 346–349.

РАЗРАБОТКА ЭФФЕКТИВНОГО МЕТОДА ОПРЕДЕЛЕНИЯ САМОПЕРЕСЕЧЕНИЙ БЕЛКОВОЙ ЦЕПИ

И. Б. Сметанников
магистрант кафедры КТ НИУ ИТМО
E-mail: smeivan2@gmail.com

М. В. Буздалов
аспирант кафедры КТ НИУ ИТМО
E-mail: mbuzdalov@gmail.com

Аннотация. Данная работа посвящена разработке эффективного метода определения самопересечений белковой цепи. Исследование белковых молекул является актуальной и малоизученной темой. Результатом работы является быстрый и достаточно точный метод определения наличия самопересечений белковой цепи.

Введение

При решении задачи построения конформационного движения белковых цепочек [1, 2] используются различные генетические и эволюционные алгоритмы. При работе данного вида алгоритмов возникает огромное число промежуточных решений различной точности, зачастую даже частично противоречащих изначально заданной модели [3]. Для оценки функции приспособленности (функции штрафов) в качестве одной из компонент используется штрафная функция за самопересечения или излишне близкое расположение элементов белковой цепи. При расчёте функции штрафов возникают две вычислительные проблемы: как наиболее эффективно отсеивать ребра цепи, которые заведомо не пересекаются (т.к. полный перебор занимает слишком много времени), а также проблема непосредственного быстрого поиска наличия пересечений уже у отобранных кандидатов. В процессе решения указанных проблем было предложено несколько методов, каждый из которых был реализован и опробован на тестовом наборе.

Методы отсеивания отрезков

В данном разделе описаны рассмотренные методы отсеивания рёбер цепи, которые заведомо не пересекаются.

Отсеивание параллелепипедами

Данный метод фактически является оптимизацией наивного алгоритма [4]. На каждом шаге работы, алгоритм покрывает начальное и конечное

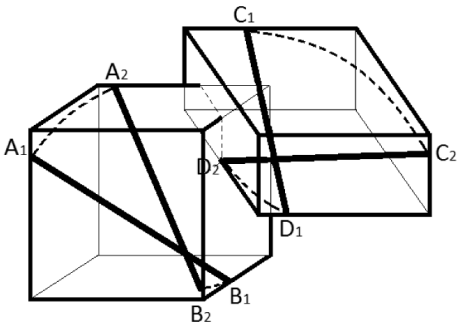


Рис. 1. Отрезок A_1B_1 переходит в отрезок A_2B_2, отрезок C_1D_1 переходит в отрезок C_2B_2. Параллелепипеды пересекаются — есть подозрение на пересечение отрезков

состояние каждого отрезка минимальным параллелепипедом со сторонами, параллельными осям, как показано на рисунке 1.

Затем, алгоритм проверяет для каждой пары отрезков наличие пересечений у этих параллелепипедов. Если параллелепипеды пересекаются, то у данной пары отрезков есть подозрение на пересечение, и далее они проверяются на пересечение более точными методами.

Отсеивание с помощью координатной сетки

Для корректной работы с координатной сеткой [4] введём следующее условие: отрезки не должны двигаться «слишком далеко». Под этим понятием будем понимать следующее. Пусть длина отрезка N. Пусть, мы можем гарантировать, что все отрезки, попавшие в структуру, движутся не дальше, чем на расстояние kN. Тогда минимальным возможным размером ячейки нашей сетки возьмём так же kN. Подобное утверждение позволит нам искать отрезки, с которыми возможно пересечение данного отрезка не дальше, чем в соседних ячейках с той, в которой лежит сам отрезок, как это показано на рисунке 2.

Для поддержки утверждения, что отрезки движутся «не слишком далеко», предварительно отфильтруем отрезки. Для каждого отрезка проверим, удовлетворяет ли он заданным ограничениям, и если нет, то будем проверять его на возможное пересечение

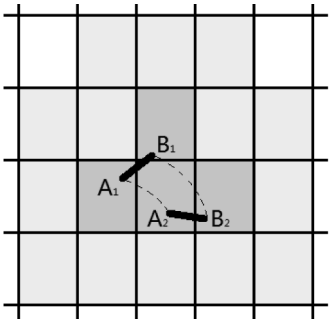


Рис. 2. Отрезок A_1B_1 переходит в отрезок A_2B_2. Проверка на возможное пересечение выполняется только с отрезками из соседних ячеек, выделенных серым цветом

со всеми остальными отрезками, проводя предварительное отсеивание параллелепипедами. Если же отрезок удовлетворяет ограничениям, то добавим его в сетку.

Отсеивание с помощью октодерева

Для корректной работы октодерева [5, 6] будем также поддерживать утверждение, что отрезки движутся «не очень далеко». Для этого воспользуемся таким же фильтром, как и в предыдущем подразделе.

Сама структура работает следующим образом. В начале необходимо построить прямоугольный параллелепипед, ограничивающий всё возможное пространство, где могут располагаться отрезки. Этот параллелепипед будет стартовой вершиной в октодереве. Затем, будем добавлять по одному ребру из списка отфильтрованных рёбер в дерево. Если в текущем элементе дерева количество отрезков превысило некоторый порог T, то разобьём этот параллелепипед на 8 малых параллелепипедов равных размеров и распределим рёбра по ним. Данное разбиение можно осуществить, если после его исполнения не нарушается ограничение на минимальный размер грани параллелепипеда kN (см. предыдущий подраздел). В случае, когда из-за разбиения параллелепипеда, данное ограничение нарушится — оно не производится, даже несмотря на превышение ограничения числа рёбер в текущем элементе дерева. Затем запускается этап проверки рёбер на пересечение, при этом кандидатами на возможное пересечение с данным ребром считаются рёбра из его ячейки и из соседних ячеек, как показано на рисунке 3.

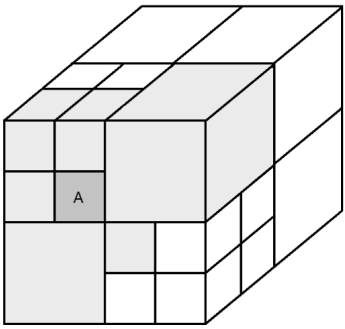


Рис. 3. Пусть рассматриваемый отрезок полностью проходит в элементе октодерева *A*. Проверка на возможное пересечение выполняется только с отрезками из соседних элементов, выделенных серым цветом

Более точная проверка отрезков с подозрением на пересечение

В данном разделе описаны более точные методы проверки отрезков на возможное пересечение. В процессе движения отрезки могут двигаться как угодно, то есть не только параллельными переносами, но и вращениями.

Проверка с помощью аппроксимации сферами

В данном методе каждый отрезок аппроксимируется сферой. Затем, два отрезка, имеющие подозрение на пересечение, проверяются путём попыт-

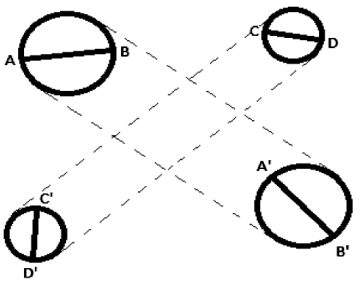


Рис. 4. Отрезок AB переходит в отрезок $A'B'$, отрезок CD переходит в отрезок $C'D'$. Если в какой-то момент аппроксимирующие сферы пересеклись, то и сами отрезки считаются пересёкшимися

ки итерационного пересечения их сфер, как показано на рисунке 4. Данный метод полностью игнорирует возможные вращения отрезков в пространстве, соответственно, в некоторых случаях, может иметь ложные срабатывания — когда аппроксимирующие сферы пересекаются, а сами отрезки нет.

Тем не менее, данный метод является достаточно быстрым и даёт довольно высокую точность определения пересечений произвольно движущихся в пространстве отрезков.

Точный поиск пересечения двух отрезков через поиск минимального расстояния между ними

Задачу поиска наличия пересечения двух отрезков можно свести к нахождению минимального расстояния между ними во время движения этих отрезков. Аналитическое нахождение этой величины сводится к нахождению корней многочленов не менее пятой степени, а функция расстояния от времени может быть мультимодальной. В данной работе минимальное расстояние ищется численным методом. Вначале расстояние между отрезками вычисляется для моментов времени с шагом 0,1. После этого минимальное расстояние вычисляется в окрестности точки с помощью тернарного поиска. Минимальное расстояние, равное нулю, соответствует пересечению отрезков. Данный метод достаточно точно находит пересечение между двумя движущимися отрезками, однако обладает достаточно высокой вычислительной сложностью.

Результаты

Данные методы были опробованы на трёх белках различного размера: 94, 148 и 480 пептидов в главной полипептидной цепи. Также, для наглядности сравнения, был произведен замер времени работы при отсутствии фильтрования — поиске пересечений наивным алгоритмом.

В первой серии опытов было измерено время работы приведённых методов отбора отрезков с подозрением на пересечение и дальнейшей проверкой на пересечение с помощью аппроксимации сферами. Результаты данных комбинаций методов можно увидеть на рисунке 5.

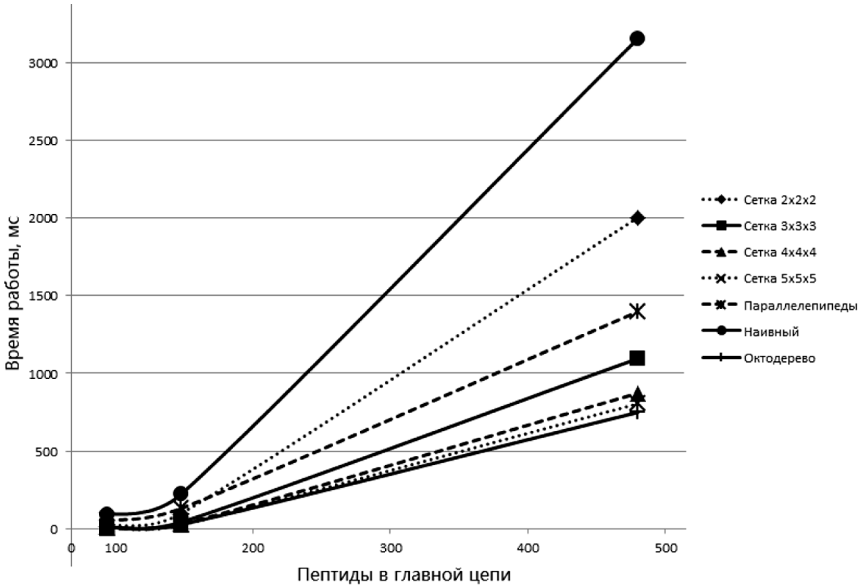


Рис. 5. Зависимость времени работы от количества пептидов в основной цепи при аппроксимации отрезков сферами

Как видно из рисунка 5, при аппроксимации отрезков сферами по времени работы октодерево выигрывает у других методов при любой длине цепи.

Во второй серии опытов было измерено время работы приведённых методов отбора отрезков с подозрением на пересечение и дальнейшей проверкой на пересечение через поиск минимального расстояния между ними. Результаты показаны на рисунке 6.

Как видно из построенного на рисунке 6 графика, при использовании данного метода проверки отрезков на пересечение, скорость работы всех алгоритмов значительно ухудшилась. Кроме того, для небольших белков, октодерево, отсечение параллелепипедами и сетка размера $5\times5\times5$ выдают примерно одинаковую производительность.

В третьей серии опытов были скомбинированы метод аппроксимации сфер и поиск расстояния между отрезками. Сначала отрезки с подозрением на пересечение проверялись более быстрым методом — с помощью аппроксимации сферами. Если сферы указывали на пересечение, то запускалась проверка с помощью поиска расстояния между отрезками, с целью отсеивания ложных срабатываний. Результаты представленного метода можно увидеть на рисунке 7.

Как видно из построенного на рисунке 7 графика, итоговое время работы для комбинированного метода больше времени работы для чистой аппрок-

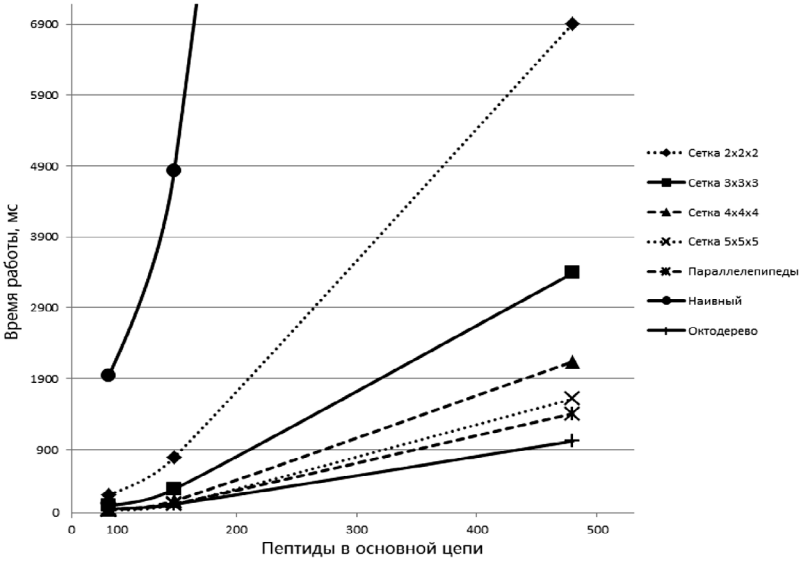


Рис. 6. Зависимость времени работы от количества пептидов в основной цепи при вычислении расстояния между отрезками

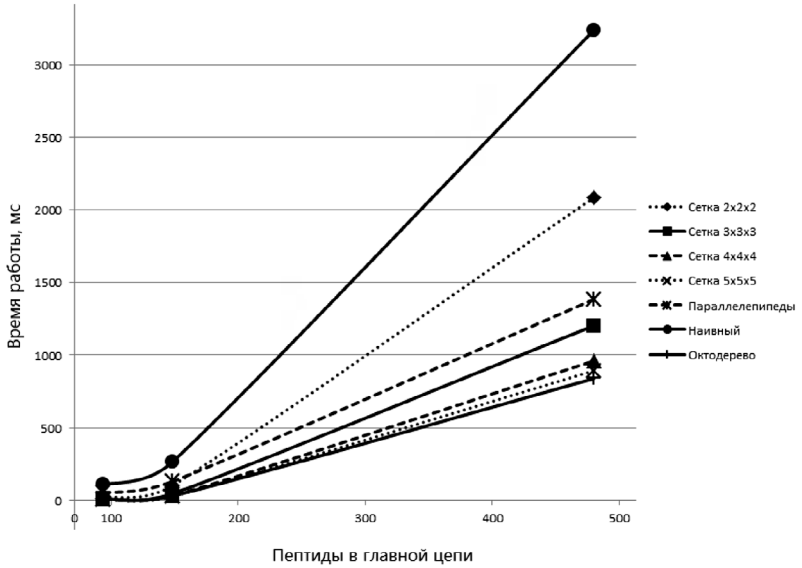


Рис. 7. Зависимость времени работы от количества пептидов в основной цепи при комбинировании сфер и расстояния между отрезками

симации сферами всего на 5–25% для различных методов. Но, в отличие от чистой аппроксимации сферами, данный метод обладает гораздо большей вычислительной точностью и отсутствием ложных срабатываний. В данном методе первое место по производительности для всех длин так же занимает октодерево.

Таким образом, итоговый алгоритм, наиболее быстро находящий самопересечения белковой цепи будет выглядеть следующим образом:

1. Отсеять «далеко движущиеся» отрезки с помощью отсеивания параллелепипедами и перейти для них к шагу 3.
2. Все оставшиеся отрезки сложить в октодерево, отсеять с его помощью заведомо не пересекающиеся отрезки, а для остальных перейти к шагу 3.
3. Для всех пар отрезков, у которых есть подозрение на пересечение, провести проверку на пересечение с помощью аппроксимации сферами. Все пары отрезков, сферы которых пересеклись, переходят к пункту 4.
4. Все оставшиеся пары отрезков проверяются на точное пересечение через подсчёт расстояния между ними.

Заключение

В представленной работе была решена задача быстрого поиска самопересечений белковой цепи. Сам алгоритм был разбит на два этапа — отсеивание заведомо непересекающихся отрезков и проверка наличия пересечения между оставшимися отрезками. Методы, представленные для каждой из частей алгоритма, были реализованы и протестированы на реальных данных. Также в работе был указан итоговый метод, который позволяет получить довольно высокую производительность и высокую точность проверки наличия самопересечений белковой цепи. В качестве продолжения работы, данная методика может быть внедрена в реальный проект по построению траекторий движения белка.

Литература

1. *Бронштейн М. П.* Атомы и электроны. М.: Наука, 1980. 152 с.
2. *Овчинников Ю. А*. Биоорганическая химия. М.: Просвещение, 1987. 815 с.
3. *D. Frenkel, B. Smit*. Understanding Molecular Simulation: From Algorithms to Applications. 1996.
4. *Ming Chieh Lin*. Efficient collision detection for animation and robotics. University of California, Berkley. 1993.
5. *Jeff Prosise*. Wicked Code // Microsoft systems journal. 1996.
6. *Erwin Coumans*. Continuous collision detection and Physics. Sony computer entertainment. 2005.

РАЗРАБОТКА МЕТОДА СБОРКИ ТРАНСКРИПТОМА НА ОСНОВЕ АНАЛИЗА КОМПОНЕНТ СВЯЗНОСТИ ГРАФА ДЕ БРЁЙНА

В. О. Долганов

магистрант кафедры компьютерных технологий СПб НИУ ИТМО

E-mail: dolganov.vlad@gmail.com

Аннотация. В данной работе рассматривается задача сборки транскриптома без учета референса (de novo). Был разработан быстрый метод сборки транскриптома, использующий небольшое количество памяти. Предложенный метод был реализован на языке программирования Java и сравнен с одним из наиболее популярных сборщиков, Trinity, на данных из реального эксперимента. Результаты показали, что предлагаемый метод работает быстрее, а качество сборки выше у Trinity.

Введение

В настоящее время биоинформатика — одно из популярных направлений в информатике, а сборка траскриптома является одной из важных задач биоинформатики. Знание транскриптома организма позволяет ускорить изучение генов, открывать новые изоформы транскриптов, анализировать экспрессию генов.

Белки играют важную роль в жизни организма. Одним из этапов биосинтеза белка является процесс транскрипции, при котором создается молекула РНК, называемая транскриптом. Транскрипт хранит информацию о гене, участке ДНК. Совокупность всех транскриптов, синтезируемых клеткой или группой клеток, называется транскриптомом.

Для получения информации о нуклеотидной последовательности, хранящейся в транскрипте, используется технологи секвенирования. В результате работы секвенатора образуются небольшие последовательности нуклеотидов, содержащиеся в транскриптах. Задача сборки транскриптома заключается в восстановлении наибольшего числа транскриптов, которые содержатся в исходных данных. Задача сборки транскриптома похожа на задачу сборки генома, но имеет ряд значительных отличий. Так, в задаче сборки генома, покрытие генома чтениями является равномерным. В задаче сборки транскриптома покрытие чтениями различных транскриптов различно и является функцией от экспрессии генов. Так же при сборке транскриптома необходимо восстановить последовательности ограниченной длины, и присутствуют дополнительные сложности, такие как наличие альтернативного сплайсинга.

В настоящее время существуют различные сборщики транскриптома. Наиболее популярными из них являются *Trinity* [1], *Velvet/Oases* [2], *Cufflinks*

[3]. Многие из них требуют больших вычислительных мощностей и большого количества памяти. В связи с этим в данной работе была поставлена задача создания алгоритма для быстрой предварительной сборки траскриптома без знания референса, использующего небольшое количество памяти. Для решения данной задачи был разработан метод, основанный на анализе компонент связности в графе де Брёйна [4]. Данный метод был реализован на языке программирования *Java*, и было произведено его сравнение с наиболее популярным *de novo* сборщиком Trinity на транскриптомных данных организма домовая мышь.

Разработанный метод

Предложенный алгоритм основан на параллельном анализе компонент связности графа де Брёйна. Алгоритм состоит из четырех этапов:

1. Сбор информации о частоте *k*-меров и их фильтрация
2. Построение графа де Брёйна для оставшихся *k*-меров и поиск в нем компонент связности без учета ориентации ребер
3. Разбиение больших компонент
4. Анализ каждой компоненты и выделение транскриптов

Далее каждый этап рассмотрен подробнее.

Подготовка k-меров

Вначале рассматриваются все *k*-меры, присутствующие в чтениях. Для них собирается статистика их появления в начальных данных. После этого *k*-меры с частотой ниже заданной удаляются из рассмотрения. Такие *k*-меры считаются ошибочными и в дальнейшей сборке не участвуют.

Построение графа де Брёйна и поиск компонент связности

После того, как все ошибочные *k*-меры были удалены, по всем оставшимся *k*-мерам строится граф де Брёйна. После этого в нем осуществляется поиск компонент связности. При этом ориентация ребер не учитывается.

Разбиение больших компонент

В результате поиска находятся множество небольших компонент и одна или несколько больших компонент. Большие компоненты имеют вид сильно связных подграфов и дальнейшему анализу не подлежат. В связи с этим, данные компоненты разбиваются на более мелкие путем удаления ошибочных ребер. Ребра удаляются с учетом частоты *k*-меров, которых они соединяют. После данного этапа получается множество небольших компонент, которые в дальнейшем анализируются.

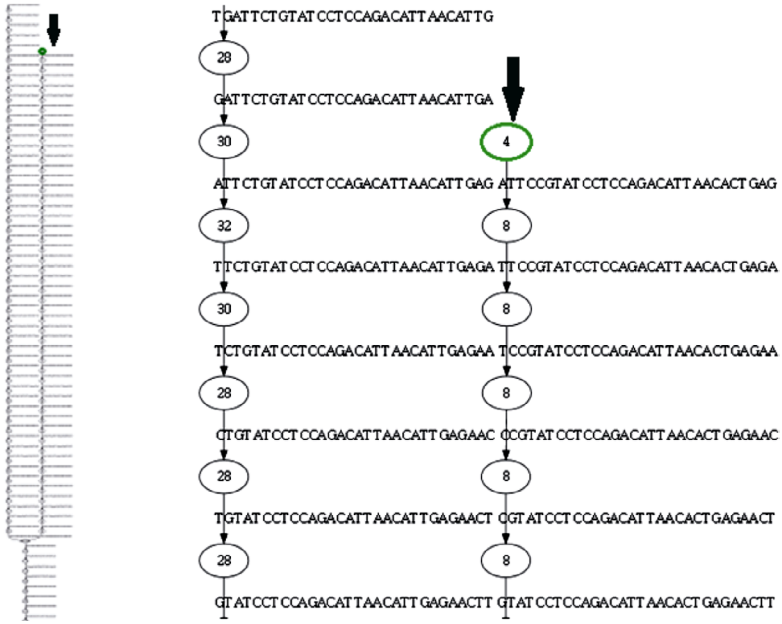


Рис. 1. Стартовая вершина

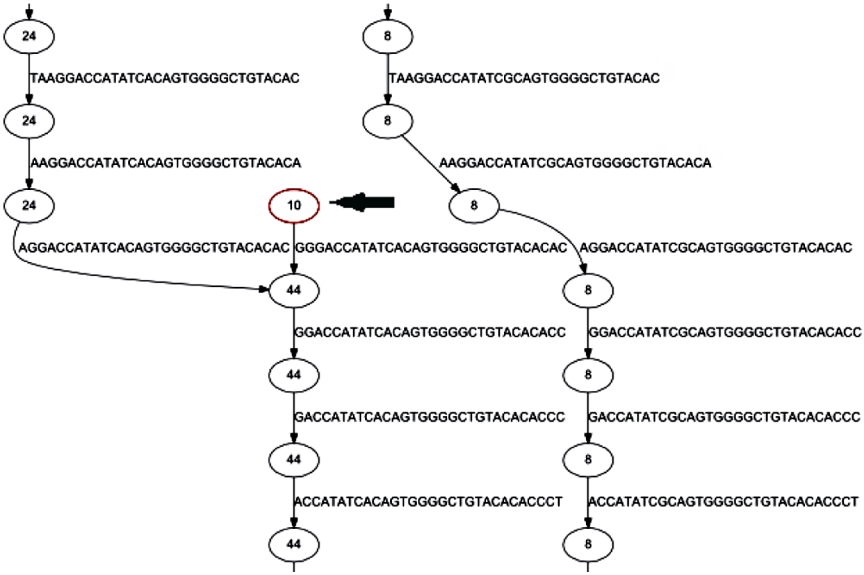


Рис. 2. Ошибочный отросток

Анализ компонент

На последнем шаге алгоритма каждая компонента независимо анализируется. Вначале ищутся стартовые вершины: такие вершины степени один, что расстояние до вершины степени не меньше трех больше заданного (Рис. 1). Затем удаляются ошибочные отростки — цепочки вершин степени два с вершиной степени один, расстояние от которой до вершины со степенью хотя бы три меньше заданного (Рис. 2). После этого происходит поиск путей в компоненте, начинающихся в стартовых вершинах и заканчивающихся в вершинах степени один. При этом учитывается покрытие *k*-меров для отбрасывания ложных веток (Рис. 3).

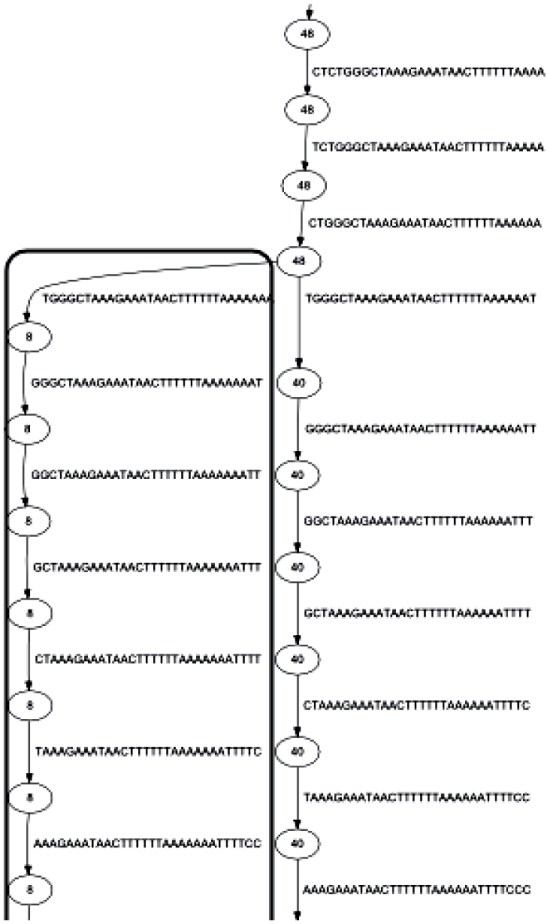


Рис. 3. Ложная ветка

Эксперименты

Описанный метод был реализован на языке программирования *Java* и был сравнен с одним из наиболее популярных *de novo* сборщиков Trinity. Сравнение производилось на реальных транскриптомных данных организма домовая мышь [5]. Данные содержали около 32 миллионов пар чтений, длиной около 100 нуклеотидов (16 ГБ). Сравнение производилось на вычислительном узле НИУ ИТМО, имеющий следующие характеристики: 24 ядра, 128 ГБ оперативной памяти. Результаты тестирования представлены в Таблице 1. Как видно из результатов, предложенный сборщик работает в несколько раз быстрее, и длина самого длинного транскрипта не сильно отличается в обоих случаях.

Таблица 1

Сравнение разработанного сборщика и *Trinity*

Показатель	**Разработанный сборщик**	***Trinity***
Число контигов	223 432	478 826
Суммарная длина контигов	191 279 463	555 989 630
Максимальная длина контига	23 701	24 756
Минимальная длина контига	200	201
Средняя длина контига	856	1161
Время работы	10 часов	41 час 17 минут

Заключение

В данной работе был предложен метод для быстрой предварительной сборки траскриптома без знания референса, использующего небольшое количество памяти. Данный метод был реализован, и было произведено его сравнение с одним из популярных сборщиков *Trinity* на реальных данных. Результаты экспериментов показывают, что предложенный метод работает быстрее и требует меньше памяти. Дальнейшая работа будет направлена на улучшения качества сборки путем модернизации отдельных этапов алгоритма.

Литература

1. *Manfred G. Grabherr, Brian J. Haas, Moran Yassour, Joshua Z. Levin, Dawn A. Thompson, Ido Amit, Xian Adiconis, Lin Fan, Raktima aychowdhury, Qiandong Zeng, Zehua Chen, Evan Mauceli, Nir Hacohen, Andreas Gnirke, Nicholas Rhind, Federica di Palma, Bruce W. Birren, Chad usbaum, Kerstin Lindblad-Toh, Nir Friedman* and *Aviv Regev.* «Full-length transcriptome assembly from RNA-Seq data without a reference genome». Volume 29, number 7, 2011. Nature biotechnology. Pp. 644–654.

2. *Marcel H. Schulz, Daniel R. Zerbino, Martin Vingron* and *Ewan Birney.* «Oases: robust de novo RNA-seq assembly across the dynamic range of expression levels». Volume 28, number 8, 2012. Bioinformatics. Pp. 1086–1092
3. *Cole Trapnell, Brian A. Williams, Geo Pertea, Ali Mortazavi, Gordon Kwan, Marijke J. van Baren, Steven L. Salzberg, Barbara J. Wold* and *Lior Pachter.* «Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation». Volume 28, number 5, 2010. Nature biotechnology. Pp. 511–517.
4. *de Bruijn N. G*. A combination problem // Koninklijke Nederlandse Akademie. V. Wetenschappen. 1946. Volume 49. Pp. 758–764.
5. The Sequence Read Archive http://sra.dnanexus.com/runs/SRR203276

ПОСТРОЕНИЕ ГЕНЕТИЧЕСКИХ КАРТ ПО НЕПОЛНЫМ И ЗАШУМЛЕННЫМ ДАННЫМ

А. С. Крамар

студентка 5 курса Математико-механического факультета СПбГУ

Предметная область

Краткая медицинская экнциклопедия [1] дает следующее определение слову «генетика»: наука о наследственности и изменчивости организма. Согласно законам наследования все основные признаки и свойства любых организмов определяются и контролируются единицами наследственной информации — генами, локализованными в специфических структурах клетки — хромосомах [2]. В связи с этим, основной задачей генетики является на основе первичной структуры биополимеров (молекул ДНК или РНК) определить фенотип особи, механизмы наследования тех или иных признаков и т.п. Получение информации о первичной структуре ДНК называется секвенированием [3]. К сожалению, современные методы секвенирования не в состоянии предоставить информацию о полной нуклеатидной последовательности в рамках конкретной хромосомы [4]. Более того, нуклеатидная последовательность не содержит прямой информации о происхождении гена (маркера), и становится затруднительным определить, от какого предка был унаследован тот или иной признак. Для определения шаблонов и выявления закономерностей наследования признаков и предназначены генетические карты хромосом [5].

Генетическая карта — это схема расположения генов на хромосоме (здесь и далее подразумевается структурный ген, т.е. ген, имеющий отличимое нуклеатидное представление, которое позволяет его идентифицировать) и маркеров. Зачастую, наличие генов не так важно, как наличие маркеров, потому что гены имеют свойство наследоваться неполностью [6].

Генетические карты строятся на основе нуклеатидных последовательностей или последовательностей маркеров, получаемых после севенирования молекуды ДНК конкретной особи. Стоит упомянуть, что задача построения генетической карты хромосомы имеет смысл для диплоидных особей и лучше решается, когда исследуемые особи находятся в родстве. Задача так же решается проще на семействах особей, которые размножаются в большой скоростью, поэтому большее количество карт на данный момент имеют хромосомы дрозофилов, кошек, мелких грызунов и насекомых.

Основным применением генетической карты является диагностирование тяжелых наследственных болезней (болезнь Альцгеймера, гемофилия) как после выявления синдромов, так и до наступления их проявления, основываясь только на ДНК анализе.

Существующие методы построения генетических карт

Так как входными данными является последовательность маркеров, выявленных при секвенировании, основной задачей становится получить информацию о расположении маркера на молекуле ДНК (физическое или относительное, но это не имеет большого значения, потому что они приводимы друг к другу). Предположения о генетической природе наседования были сделаны Менделем, во времена становления Менделевской генетики. Его ученик и получил опытным путем первую генетическую карту (1913) [7], основываясь на предположении, что чем дальше маркеры отстают друг от друга на молекуле, тем больше вероятность того, что они перепутаются (явление кроссинговера). Это предположение позволило получить первые программные продукты, осуществляющие генетическое картирование. В наше время самыми употребимыми инструментами для построения карт являются программа CRIMAP [8] и LM_MAP [9]. Основной принцип обоих инструментов заключается в том, что на основе полученных наборов геномов особей получить данные о том, от какого предка унаследовался признак методом максимизации правдоподобия. Но тем не менее, каждый алгоритм производит вычисление матрицы попарных расстояний между маркерами, что практически эквивалентно матрице частоты рекомбинаций между генами.

Нахождение матрицы попарных расстояний

Так как мы не рассматриваем многократный кроссинговер между маркерами, то в нашем случае вероятность рекомбинации равна расстоянию между генами в сантиморганах (1 сМ). В прототипе рассматриваемого мной алгоритма вычисление расстояния между двумя генами в сантиморганах осуществлялось за счет выявления «отца» маркера за счет попарного рассмотрения локуса, при этом изначально храня отцовскую и материнскую хромосомы. В таком случае, если особь в локусе гомозиготна, то она унаследовала от обоих родителей одинаковый набор, что позволяет идентифицировать данный локус родителя. В случае гетеродиготности в рассматриваемом локусе, для определения происхождения участка стоит рассмотреть родителей и если среди них есть гомозиготная по данному локусу особь, то это позволяет определить тот участок, что достался от найденного гомозиготного родителя, а второго гродителя в этом локусе проинициализировать дополнением. После заполнения хромосом происходит подсчет рекомбинаций. Основной проблемой рассматриваемого мной прототипа является его неуниверсальность: заточенность под один вход, нечувствительность к неоднократному кроссоверу и неоптимальная реализация на языке python. Достоинствами алгоритма является асимптотически меньшая сложность, чем у аналогов, корректность и простая реализация. В связи с этим моей задачей было превращение прототипа в рабочую версию. В таблице показано время работы представленного алгоритма и общепринятого.

Таблица 1

	Sysoev	**Lander**
192 особи, 35 маркеров	1с	2 + 5 с

Унификация и повышение вычислительной скорости алгоритма

Первой трудностью, с которой я столкнулась при рассмотрении алгоритма оказалось различие входов. С одной стороны, первой мыслью было написать транслятор входных данных, пригодных для различных инструментов с целью стравнения их эффективности и точности получаемых значений. CRIMAP и LM_MAP используют строковые литералы, работа с которыми (нахождение комплиментарного, сравнение) выполняется процессором медленнее. В нашем случае используются примитивы (битовые последовательности, целочисленный тип данных). Для этого потребовалось создать объектную модель, оперирующую экземплярами класса с определенными для него методами, что семантически является более верным, так как тогда алгоритм становится проще для рефакторинга, внесения изменений и переиспользования. Как правило, методы ООП замедляют работу алгоритма, но в случае объекта добавляются «ссылочные возможности», что ускоряет поиск предков и потомков конкретной особи, а так же пропадает завязка на индексы в первоначальном кортеже, что уменьшает шанс допустить ошибку. Преставленный в табл. 2 пример показывает, что удалось получить выйгрыш в скорости работы, расширяя семантику входных данных.

Таблица 2

	Initial	**With Objects**
192 особи, 35 маркеров	1 с	1 с
2007 особей, 321 маркер	24 с	22 с

К тому же, сейчас ведется работа над сохранением корректности с учетом неоднократного кроссовера (так как в случае этого явления, получаемые данные могут быть неточными [7]) которая будет поддерживаться путем нескольких итерирований и корректировкой получаемой матрицы попарных расстояний.

Заключение

В ходе проводимой работы было выяснена и проверена работоспособность прототипа метода Сергея Сысоева, расширена возможность применения данного прототипа, а так же улучшение его производительности из-за оптимизации реализации. Так же была доказана корректная работа в случае

однократного кроссовера между участками хромосом. В дальнейшем планируется расширить область действия в виде возможности анализа данных с вероятным неоднократным кроссовером. В случае успешности последнего соображения планируется выпуск web-приложения, позволяющего строить генетические карты.

Литература

1. Краткая Медицинская Энциклопедия. «Советская Энциклопедия», 2-е изд., 1989, Москва.
2. *Griffiths, Anthony J. F.; Miller, Jeffrey H.; Suzuki, David T.; Lewontin, Richard C.; Gelbart*, eds. (2000). «Genetics and the Organism: Introduction». An Introduction to Genetic Analysis (7th ed.). New York: W. H. Freeman. ISBN 0-7167-3520-2.
3. *Альбертс Б., Брей Д., Льюис Дж., Рэфф М., Робертс К., Уотсон Дж.* Молекулярная биология клетки: в трех томах. 2. Москва: Мир, 1994. Т. 1. 517 с.
4. *Strachan T., Read E*. Human Molecular Genetics, 2nd Edition. NY: Willey-Liss. 1999
5. *Morgan, Thomas Hunt; Alfred H. Sturtevant, H. J. Muller* and *C. B. Bridges*. The Mechanism of Mendelian Heredity. — New York: Henry Holt, 1923.
6. *Bateson, W*. (1907). «The Progress of Genetic Research». In Wilks, W. Report of the Third 1906 International Conference on Genetics: Hybridization (the cross-breeding of genera or species), the cross-breeding of varieties, and general plant breeding. London: Royal Horticultural Society.
7. *Griffiths A. J. F., Miller J. H., Suzuki D. T., Lewontin R. C., Gelbart W. M*. (1993). «Chapter 5». An Introduction to Genetic Analysis (5th ed.). New York: W. H. Freeman and Company.ISBN 0-7167-2285-2.
8. Documentation for CRI-MAP, version 2.4 (3/26/90) Phil Green, Kathy Falls, and Steve Crooks URL : (http://saf.bio.caltech.edu/saf_manuals/crimap-doc.html)
9. Introduction to lm_map. Elizabeth Thompson. URL: (http://www.stat.washington.edu/thompson/Genepi/MORGAN/morgan-tut-html-v29/morgan-tut_103.html)

Синтез элементов компьютерной архитектуры

Леонов
Геннадий Алексеевич

председатель оргкомитета конференции

д.ф.-м.н., профессор, чл.-корр. РАН
декан математико-механического факультета СПбГУ
заведующий кафедрой прикладной кибернетики СПбГУ

ДВУХФАЗНАЯ СХЕМА КОСТАСА И ГИПОТЕЗА БЕСТА

Г. А. Леонов

член-корреспондент Российской Академии наук, профессор, д. ф.-м. н., декан Математико-Механического факультета СПбГУ

E-mail: leonov@math.spbu.ru

Н. В. Кузнецов

к. ф.-м. н, доцент, ученый секретарь кафедры Прикладной кибернетики Математико-Механического факультета СПбГУ

E-mail: kuznetsov@math.spbu.ru

К. Д. Александров

аспирант кафедры Прикладной кибернетики Математико-Механического факультета СПбГУ

E-mail: Konstantin.239.Alexandrov@gmail.com

Аннотация. В данной работе описывается схема фазовой автоподстройки частоты и производится исследование соответствующей ей математической модели. Исследуемая модель сводится к обыкновенному дифференциальному уравнению второго порядка. Приводится гипотеза о качественном поведении рассматриваемой схемы, сформулированная Р. Бестом. Доказывается справедливость этой гипотезы для малых значений одного из параметров системы.

Введение

Схема Костаса была изобретена во второй половине XX века [1] и широко используется в схемах управления для восстановления несущей фазы сигнала [2, 3]. Изучение схемы Костаса является сложной задачей [4, 5], которая требует исследования соответствующей нелинейной математической модели. В данной работе рассмотрена гипотеза относительно области захвата двухфазной схемы Костаса, предложенная Рональдом Бестом, специалистом в области систем ФАП [6, 7], и показано, что эта гипотеза верна для случая малого параметра.

Далее рассматривается схема Костаса, описанная в [8, 9, 10], и в частности, ее двухфазная модификация (см. Рис. 1), которая подробно описана в работах [11, 12, 13, 14]. Элементами данной схемы являются эталонный генератор (Input), пропорционально-интегрирующий фильтр (Filter) с передаточной функцией

$$W(s) = \frac{\beta + as}{s},$$

где $a > 0$, $\beta > 0$ и подстраиваемый генератор, управляемый напряжением (VCO) с коэффициентом усиления $L > 0$. На вход схемы эталонным генера-

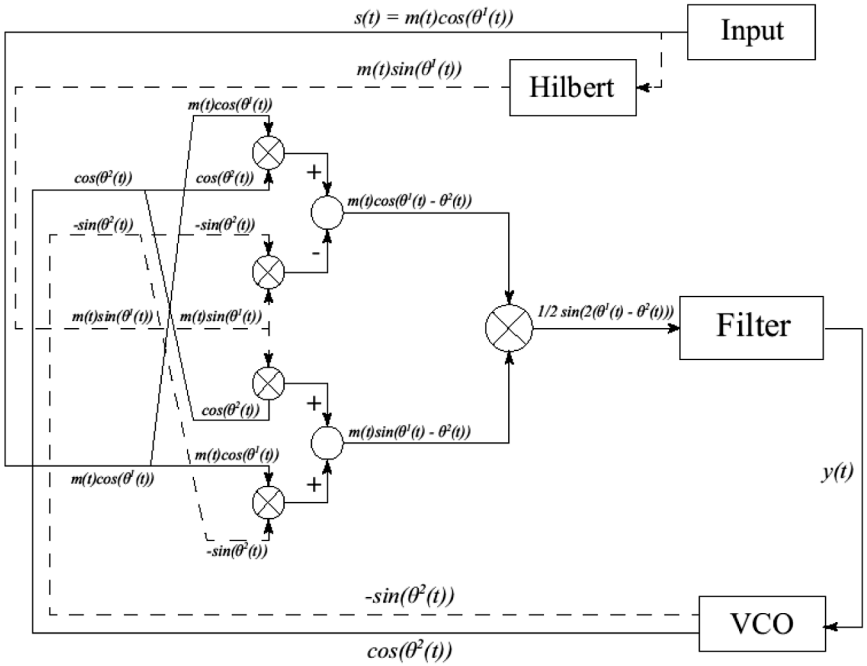


Рис. 1. Двухфазная модификация схемы Костаса

тором подается несущий сигнал $m(t)\cos(\theta^1)$, где $m(t) = \pm 1$ — относительно медленно меняющийся сигнал данных. Несущий сигнал преобразуется преобразователем Гильберта (Hilbert) в $m(t)\sin(\theta^1)$. VCO в свою очередь генерирует сигналы $\cos(\theta^2)$ и $-\sin(\theta^2)$. Для достижения фазовой синхронизации частоты используется комплексный мультипликатор, на вход которого поступают сигналы, генерируемые Input и VCO, и подает на вход пропорционально-интегрирующего фильтра Filter сигнал

$$\varphi(\theta^2 - \theta^1) = \frac{1}{2}\sin\left(2(\theta^2 - \theta^1)\right).$$

После фильтрации этот сигнал используется для изменения частоты VCO.

Ниже приведены уравнения, соответствующие двухфазной схеме Костаса. Производная по времени t обозначается $\dot{\theta}$, а производная по x обозначается y'.

$$\begin{cases} u(\theta^2 - \theta^1) = \varphi(\theta^2 - \theta^1), \\ \ddot{y}(\theta^2 - \theta^1) = a\dot{u}(\theta^2 - \theta^1) + \beta u, \\ \dot{\theta}^1 = \omega^1, \\ \dot{\theta}^2 = \omega^2_{free} + Ly(\theta^2 - \theta^1). \end{cases} \quad (1)$$

Первое уравнение системы (1) означает, что входной сигнал фильтра Filter есть $\varphi(\theta^2 - \theta^1)$. Второе соотношение является уравнением фильтра. Третье уравнение есть соотношение, по определению связывающее частоту и фазу блока Input. Четвертое уравнение системы (1) описывает процесс подстройки частоты.

Для системы (1) рассмотрим эквивалентные ей дифференциальное уравнение второго порядка относительно разности фаз $\theta_e = \theta^2 - \theta^1$ и соответствующую ему систему дифференциальных уравнений.

$$\ddot{\theta}_e + \frac{aL}{2}\cos(2\theta_e)\dot{\theta}_e + \beta L \sin(2\theta_e) = 0;$$

$$\begin{cases} \dot{x} = y, \\ \dot{y} = -aL\cos(x)\, y - \beta L \sin(x). \end{cases} \qquad (2)$$

Гипотеза Беста

Гипотеза, предложенная Р. Бестом, может быть сформулирована следующим образом:

Утверждение. *Для двухфазной схемы Костаса с передаточной функцией фильтра*

$$W(s) = \frac{\beta + as}{s},$$

$a > 0$, $\beta > 0$, описываемой уравнением (2) имеет место глобальная асимптотическая устойчивость.

В данной работе сформулированная гипотеза была подтверждена для малого значения параметра а следующей теоремой

Теорема. *Система (2) при малом значении параметра $0 < a \ll 1$ имеет свойство глобальной асимптотической устойчивости*.

Доказательство теоремы состоит в рассмотрении сепаратрис системы (2) на фазовой плоскости как решений от двух параметров *x* и *a* (см. Рис. 2). Доказывается, что качественному расположению сепаратрис на фазовой плоскости системы (2), обоснованному в ходе вычислений, соответствует глобальная асимптотическая устойчивость фазовых траекторий системы.

Для этого производится разложение исследуемых фазовых траекторий в ряд Тейлора с остаточным членом в форме Лагранжа. Обосновываются свойства гладкости и равномерной ограниченности для членов разложения. Аналитически доказывается качественно одинаковое поведение сепаратрис системы (2) и их приближений частичными суммами соответствующих рядов

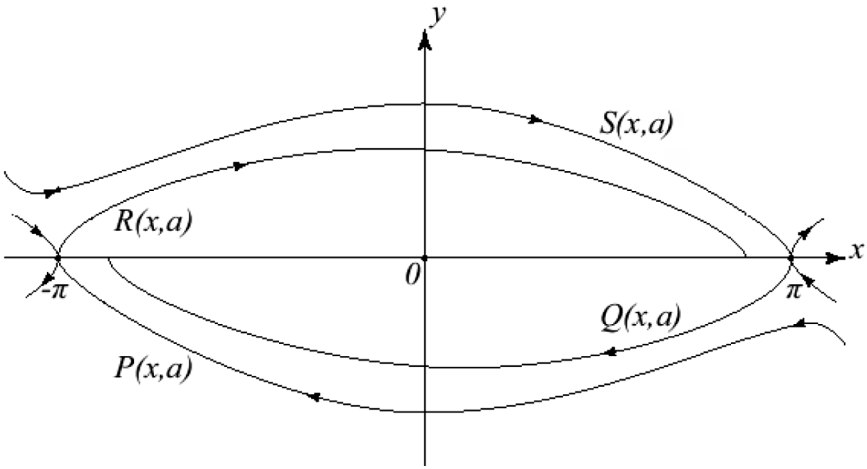


Рис. 2. Сепаратрисы фазовой плоскости системы

Тейлора. Описывается и аналитически обосновывается процедура, с помощью которой производится последовательное вычисление членов рядов Тейлора для каждой из сепаратрис. Для определения качественного поведения сепаратрис в рассматриваемой системе (2) достаточным является нахождение первых двух членов ряда Тейлора.

Заключение

Для описываемой в работе схемы фазовой автоподстройки частоты производилось исследование соответствующей ей математической модели. Исследуемая модель была сведена к обыкновенному дифференциальному уравнению второго порядка. Сформулированная Р. Бестом гипотеза о качественном поведении рассматриваемой схемы имеет важное прикладное значение. Теорема, которая доказана в данной работе и подтверждает данную.

Литература

1. *J. Costas*. Synchronous communications // Proc. IRE. 1956. Vol. 44. Pp. 1713–1718.
2. *R. Best*. Phase-lock loops: Design, simulation and application. Oberwill, Switzerland: McGraw Hill, 2003.
3. *W. Lindsey*. Synchronization systems in communication and control. New Jersey: Prentice-Hall, 1972.
4. *G. A. Leonov, N. V. Kuznetsov, M. V. Yuldashev, R. V. Yuldashev*. Differential equations of Costas loop // Doklady Mathematics, 86 (2). 2012. Pp. 723–728.
5. *D. Abramovitch*. Phase-locked loops: A control centric tutorial // Proceedings of the American Control Conference. 2002. Vol. 1. Pp. 1–15.

6. *Roland E. Bes*t. Phase-Lock Loops: Design, Simulation and Application. McGraw Hill, 2007.
7. *R. E. Best, N. V. Kuznetsov, G. A. Leonov, M. V. Yuldahsev, R. V. Yuldashev*. Nonlinear analisys of phase-locked loop-based Circuits. Springer International Publishing Switzerland, 2014. Pp. 169–192.
8. *G. A. Leonov, N. V. Kuznetsov*. Nonlinear Mathematical Models of phase-locked loops. Stability and Oscillations. Cambridge Scientific Publishers, 2014.
9. *G. A. Leonov*. Computation of phase detector characteristics in phase-locked loops for clock synchronization // Doklady Mathematics, 78 (1). 2008. Pp. 643–645.
10. *N. Kuznetsov, G. Leonov, S. Seledzhi*. Nonlinear analysis of the costas loop and phase-locked loop with squarer // Proceedings of the IASTED International Conference on Signal and Image Processing. Vol. 654. Pp. 1–7. ACTA Press, 2009.
11. *G. A. Leonov, N. V. Kuznetsov, M. V. Yuldahsev, R. V. Yuldashev*. Analytical method for computation of phase-detector characteristic // IEEE Transactions on Circuits and Systems. II: Express Briefs, 59 (10). Pp. 633–647, 2012.
12. *N. Kuznetsov, G. Leonov, M. Yuldashev, R. Yuldashev*. Analytical methods for computation of phase-detector characteristics and pll design // International Symposium on Signals, Circuits and Systems. IEEE press, 2011. Pp. 7–10.
13. *N. Kuznetsov, G. Leonov, P. Neittanmäki, M. Yuldashev, R. Yuldashev*. High-frequency analysis of phase-locked loop and phase detector characteristic computation // 8th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2011). INSTICC Press, 2011. Pp. 272–278.
14. *N. V. Kuznetsov, G. A. Leonov, M. V. Yuldashev, R. V. Yuldashev*. Computation of phase detector characteristics in synchronization systems // Doklady Mathematics, 84(1). 2011. Pp. 459–463.

ПРОГРАММА ВОССТАНОВЛЕНИЯ ПАРАМЕТРОВ ПОТОКА ОБРАЩЕНИЙ К УСТРОЙСТВУ ХРАНЕНИЯ

Г. Крайчик

математико-механический факультет, 3 курс

E-mail: george.kraychik@gmail.com

Аннотация. По последовательности запросов к устройству хранения требуется восстановить параметры потока обращений с целью повышения эффективности работы кэш-памяти. В работе представлены эмпирически выявленные закономерности распределения потока обращений к устройству хранения. Получено три алгоритма восстановления параметров распределения времени обращений, а также приведен их сравнительный анализ.

Введение

Важнейшей задачей организации работы системы хранения данных является увеличение скорости доступа к информации. Существуют различные способы решения данной задачи. Одним из способов является приобретение высокоскоростного оборудования, существенно увеличивающего общую производительность системы, например, оборудования на основе SSD устройств хранения. Однако данное решение является крайне дорогостоящим из-за потребности хранения чрезвычайно больших объемов данных, что приводит к выводу об экономической нецелесообразности такого подхода.

Оптимальным решением проблемы ускорения работы системы хранения является оснащение ее кэш-памятью. Эффективность использования кэш-памяти базируется на двух аспектах:

1. выбор оптимального размера кэш-памяти таким образом, чтобы достичь оптимального соотношения между затратами и повышением производительности системы;
2. разработка алгоритмов, позволяющих наилучшим образом управлять заполнением и освобождением кэша.

В простых системах хранения информации в качестве алгоритма управления кэш-памятью используется LRU-алгоритм (Last Recently Used). Его идея заключается в том, что в каждый момент времени с кэш-памятью ассоциирован некоторый стек, который отражает ее содержимое. Если при обращении к кэш-памяти происходит cache-miss, самая последняя ссылка вываливается из стека (и, соответственно, из кэша), а новая добавляется на его вершину. Если cache-hit, найденная ссылка переносится на вершину стека. При этом количество ссылок, находившихся над ней в стеке, называется ее стековым расстоянием. Преимущество данного алгоритма заключается в простоте его реализации и сравнительно высокой эффективности работы.

Сложные системы используют гораздо более продвинутые алгоритмы, основанные на сборе и анализе статистической информации, полученной из анализа загрузки системы запросами ввода-вывода при характерных для данной системы профайлах использования и предсказании дальнейшей загрузки системы хранения.

Поток обращений является последовательностью запросов, каждый из которых состоит из четырех компонент: время, адрес, размер и тип запроса. По сути каждая компонента запроса является случайной величиной, имеющей собственный набор параметров, которые могут меняться с течением времени. Можно считать, что каждая компонента независима от других, поэтому их можно анализировать независимо друг от друга.

1. Эмпирически установлено, что случайная величина, равная разности времен между двумя последовательными запросами распределена экспоненциально.
2. Известно, что в большинстве практических ситуаций, если вычислить стековые расстояния последовательности адресов (подобно LRU-алгоритму), то окажется, что они распределены по закону Парето
3. Размер запроса является дискретной случайной величиной, принимающий конечное множество значений, которое зависит от системы хранения. (Например, размер запросов в системе может принимать значения 2, 4, 8, 16 кбайт). Для анализа будет достаточно рассматривать вероятности появления запроса данного размера.
4. Имеется только два типа запросов: чтение и запись. Тип запроса будет анализироваться по аналогии с пунктом 3, так как он также принимает конечное множество значений.

В реальности параметры потока обращений изменяются во времени. Причинами этого являются смена дня и ночи (обычно днем нагрузка на устройство хранения выше, чем ночью), будний или выходной день, сезонность, изменение спроса к сервису, использующего данную систему хранения и прочее. Таким образом, параметры потока обращений постоянно меняются и, как следствие этого, требуется выявлять моменты переключения с одних параметров на другие.

В данной работе рассматривается проблема восстановления параметров потока обращений по времени (первая компонента запроса) при условии его неизменности для всех запросов потока обращений.

Восстановление параметров распределения потока обращений по времени

В данном разделе будут рассмотрены алгоритмы, анализирующие последовательность времен потока обращений.

Пусть $\{t_i\}_{i=0}^{n}$ — последовательность времен обращений к устройству хранения. Получим новую последовательность $\{x_i\}_{i=1}^{n}$, такую что $x_i = t_i - t_{i-1}$,

где $i = 1, 2, \ldots, n$. Известно, что $\{x_i\}_{i=1}^n$ была порождена экспоненциальным распределением с фиксированным параметром λ. По набору данных $\{x_i\}_{i=1}^n$ требуется получить наиболее правдоподобное значение параметра λ.

Существует множество методов восстановления параметра λ. Для экспоненциального распределения наиболее эффективным считается метод моментов (при присущей ему простоте реализации). Его идея заключается в уравнивании математического ожидания экспоненциального распределения (E_1) и выборочного среднего последовательности $\{x_i\}_{i=1}^n$ (E_2).

$$\boldsymbol{E}_1 = \frac{1}{\boldsymbol{\lambda}}, \quad \boldsymbol{E}_2 = \frac{1}{\boldsymbol{n}} \sum_{\boldsymbol{i}=1}^{\boldsymbol{n}} \boldsymbol{x}_{\boldsymbol{i}}, \quad \boldsymbol{E}_1 = \boldsymbol{E}_2.$$

Решая данное уравнение, получим

$$\lambda = \frac{\boldsymbol{n}}{\sum_{\boldsymbol{i}=1}^{\boldsymbol{n}} \boldsymbol{x}_{\boldsymbol{i}}}.$$

Данная оценка хорошо приближает значение параметра λ, однако не учитывает его вероятностное поведение. Например, для конкретной системы хранения может быть эмпирически установлено, что $\lambda \in [0, 1]$. При этом какие-то значения параметр λ принимает чаще, а какие-то реже. Отсюда следует, что еще до начала анализа данных $\{x_i\}_{i=1}^n$, λ уже обладает некоторым базовым распределением. Целесообразно придумать алгоритм, который бы, учитывая данное обстоятельство, вычислял бы λ более точно, чем метод моментов.

Формализуем вышесказанное. Имеется некоторая случайная величина Λ, являющаяся параметром экспоненциального распределения Y. Таким образом, функция распределения Y зависит от случайной величины Λ. Также имеется случайный вектор $X = (X_1, \ldots, X_n)$, где X_i — случайная величина, имеющая такое же распределение, как и Y. В данном случае $X_1, \ldots, X_n$ — n независимых испытаний случайной величины Y.

Предлагается рассмотреть условную плотность вероятности случайной величины Λ при условии, что $X = (x_1, \ldots, x_n)$. Полученную случайную величину назовем ξ. $p_\xi(\lambda) = p_{\Lambda|X}(\lambda|(x_1, ..., x_n)) = \dfrac{p_{\Lambda,X}(\lambda, x_1, ..., x_n)}{p_X(x_1, ..., x_n)}$. Числитель и знаменатель дроби вычисляется по формуле Байеса и формуле полной вероятности для абсолютно непрерывных случайных величин.

Случайная величина ξ описывает вероятность того, что Λ принимает то или иное значение, при условии, что при проведении n независимых испытаний, случайная величина Y выдала набор данных $(x_1, \ldots, x_n)$. До начала анализа набора данных имелось некоторое базовое знанием о распределении

Λ. Однако после анализа данных, базовое распределение будет откорректировано в соответствии с результатами независимых испытаний.

Под различные системы хранения необходимо подбирать различные базовые распределения Λ. Чем точнее базовое распределение будет отражать реальность, тем точнее будет восстановлен исходный параметр.

Рассмотрим самый простой случай, когда о случайной величине Λ не известно ничего за исключением множества ее возможных значений. Иными словами известно, что $\Lambda \in [a, b]$ для некоторых $0 < a < b$. В таком случае разумно считать Λ равномерно распределенной случайной величиной на отрезке $[a, b]$. Получим формулу $p_\xi(\lambda)$:

$$p_\xi(\lambda) = \frac{\frac{1}{b-a}\int_a^\lambda t^n e^{-t\sum_{i=1}^n x_i} dt + \frac{\lambda - a}{b-a}\left(\lambda^n e^{-\lambda\sum_{i=1}^n x_i}\right)}{\int_a^b t^n e^{-t\sum_{i=1}^n x_i} dt} \chi_{[a,b]}(\lambda),$$

$\chi_{[a, b]}$ — характеристическая функция множества $[a, b]$.

Очевидно, что среди всех точек $\lambda \in [a, b]$ нужно выбрать точку λ^*, в которой достигается наибольшее значение функции $p_\xi(\lambda)$. Оказывается, что максимум приведенной выше функции $p_\xi(\lambda)$ достигается ровно в одной точке

$$\lambda^* = \frac{ac + n + 2 + \sqrt{(ac - n + 2)^2 + 8n}}{2c},$$

где $c = \sum_{i=1}^n x_i$. Заметим, что λ^* зависит от a, но не зависит от b.

Однако, если мы не обладаем никакой информацией не только о распределении Λ, но и о ее области возможных значений, то метод перестает быть применимым. Отсюда возникает мысль о совмещении двух методов воедино.

Предлагается в качестве базового приближения значения λ взять выражение

$$\lambda^* = \frac{\boldsymbol{n}}{\sum_{\boldsymbol{i}=1}^{\boldsymbol{n}} \boldsymbol{x_i}}.$$

Далее в зависимости от входных данных $\{x_i\}_{i=1}^n$ выбрать значения a и b так, чтобы $\lambda^* \in [a, b]$ и вероятность того, что истинное значение $\lambda \in [a, b]$ было бы очень велико (при этом достаточно выбрать только число a). После этого для выбранных значений a и b можно применить второй алгоритм. Полученное значение будет приближать оптимальное значение λ лучше, чем λ^*.

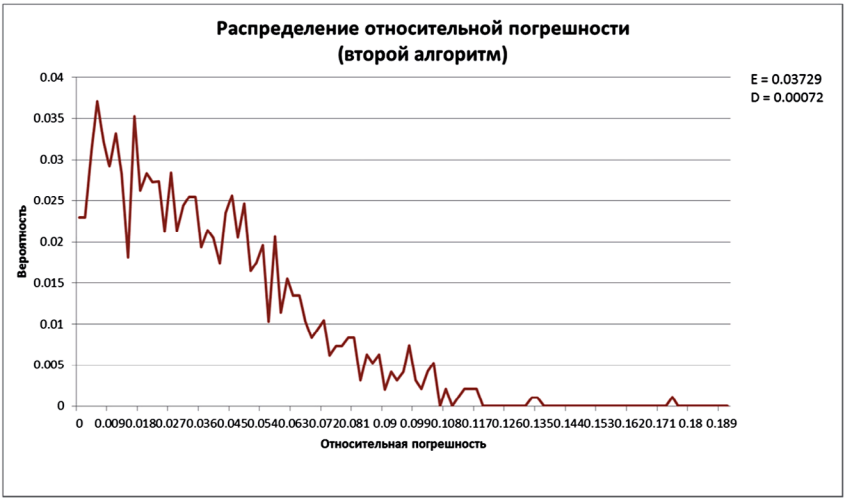
Ниже приведены графики распределения относительной погрешности трех алгоритмов. Каждый алгоритм тестировался на одних и тех же входных данных. Было получено 1000 файлов, каждый из которых состоял из 1000 чисел, сгенерированных по экспоненциальному распределению с известным параметром λ_i, где $i = 1, \ldots, 1000$. К данным каждого файла применялся алго-

ритм и, таким образом, было получено 1000 приближений исходных параметров λ_i^*. Зная истинные значения параметров, с помощью которых генерировались тестовые множества, можно получить относительную погрешность по формуле

$$k_i = \frac{\lambda_i - \lambda_i^*}{\lambda_i}.$$

Далее строились графики относительной погрешности. Справа от графиков указаны матожидание (E) и дисперсия (D) относительной погрешности.

Относительная погрешность алгоритма 1



Относительная погрешность алгоритма 2

Относительная погрешность алгоритма 3

Сравнение алгоритмов

Исходя из представленных выше графиков, можно сделать вывод, что первый алгоритм является более точным по сравнению с двумя другими, поскольку обладает наименьшим математическим ожиданием и диспер-

сией. Можно предположить, что второй и третий алгоритмы дадут более точные результаты по сравнению с методом моментов только в тех случаях, когда длина отрезка $[a, b]$ достаточно мала. Случаи, в которых базовое распределение Λ не является равномерным, требуют дополнительных исследований.

Будущие исследования

В будущем планируется применить те же идеи для восстановления параметров распределения Парето, а также получить алгоритм, восстанавливающий параметры с учетом их изменения во времени. Также планируется исследовать скорость сходимости приближенных значений к истинным значениям параметров в зависимости от объема входных данных.

Заключение

В работе были описаны основные характеристики потока обращений к устройству хранения, а также были выявлены эмпирические закономерности в распределении потока обращений. Рассмотрено три алгоритма восстановления параметров распределения потока обращений по времени. Получены графики, отражающие точность работы каждого алгоритма.

Литература

1. *Г. Б. Ходасевич.* Обработка экспериментальных данных на ЭВМ. http://dvo.sut.ru/libr/opds/i130hodo_part1/index.htm
2. *G. Almasi, C. Cascaval, D. Padua*. Calculating stack distances efficiently.

РАЗРАБОТКА МОДУЛЯ ВОССТАНОВЛЕНИЯ УТРАЧЕННЫХ ДИСКОВ В RAID (N + M) В АРИФМЕТИКЕ ПОЛЯ GF (28)

В. С. Зайберт

студентка кафедры системного программирования СПбГУ

E-mail: gethappy90@gmail.com

Научный руководитель:

А. В. Маров

сотрудник научно-исследовательской лаборатории RAIDIX

E-mail: marov.a@raidixstorage.com

Аннотация. В данной работе приведено краткое описание технологии RAID ($n + m$), а также рассмотрены различные алгоритмы восстановления данных в ней, приведены их теоретические оценки по количеству операций и представлены практические результаты исследования.

Введение

Объем данных, хранимый людьми довольно велик. Хочется иметь возможность быстрой работы с этими данными. Кроме того, хотелось бы защититься от возможных сбоев и утраты дисков или информации на них. С этой целью часто используют технологию RAID.

RAID (англ. *redundant array of independent disks* — избыточный массив независимых дисков), как понятно из названия, представляет собой объединение нескольких дисков в массив, информация между которыми будет распределяться по определенным правилам. Технология повышает скорость работы с дисками и надежность хранения информации на ней. Также, большинство видов технологии RAID позволяет восстанавливать данные после ошибок. Ошибки бывают двух типов: повреждения, место расположения которых нам известно, и скрытые повреждения, когда нам надо сначала найти место, в котором произошел сбой. Ошибки второго типа также называют SDC (Silent Data Corruption). Далее мы будем говорить об ошибках первого типа, если не сказано иного.

Расскажем для начала об известной технологии RAID6 (рис. 1). В ней каждый диск разбивается на блоки одинакового размера, блоки нумеруются внутри одного диска, затем блоки с одинаковыми номерами из разных дисков объединяются в страйп (англ. *stripe* — полоска). В каждом страйпе выбирается по два блока, в которых будут храниться специальным образом посчитанные контрольные суммы (или, как их еще называют, синдромы). Чтобы избежать разногласий в терминологии, стоит отметить, что модуль,

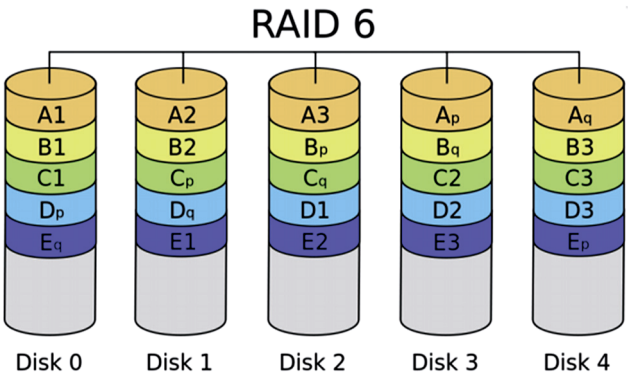


Рис. 1. Структура RAID6

выполняющий работу по восстановлению данных и подсчету контрольных сумм, работает в рамках одного страйпа. Поэтому, не умаляя общности в нашей работе можно считать, что каждый диск состоит из одного блока.

Данная технология позволяет восстанавливать до двух повреждений в каждом страйпе. То есть, если у нас вышло из строя два диска, мы сможем восстановить данные на них. А если больше?

Одно из решений это объединить несколько RAID6 в RAID60, в котором данные разбиваются на части и каждая часть записывается на отдельный RAID6. В этом случае в каждом RAID6 мы сможем восстановить до двух дисков, то есть всего 2 * *количество* RAID6. Скорость восстановления и расчета синдромов во всем RAID60 практически не отличается от скорости аналогичных операций в каждой его части. Однако, если хотя бы три поломки сосредоточенны в одном RAID6, то данные будут утеряны.

Можно реализовать модуль с тремя контрольными суммами, четырьмя и так далее. Этот подход может иметь свои плюсы, однако это не рационально с точки зрения времени разработки. Поэтому появилась технология RAID $(n+m)$, в которой n блоков в страйпе выделяется под хранение данных и m блоков для синдромов, причем значения n и m могут легко варьироваться в зависимости от желаний заказчика, без необходимости изменения кода. Однако, следует отметить, что в силу ограничений, накладываемых нашим полем, $m+n \leq 256$.

Мотивация

Зачем необходимо уметь восстанавливать данные после большого числа отказов и делать это быстро? Многие считают, что технологии сегодня достаточно развиты и сбои происходят крайне редко. Однако, как показывают наблюдения, большие системы до 30% времени работы находятся в состоянии Dergaded mode, в котором хотя бы один диск помечен как отказавший,

и либо запущен процесс восстановления, либо он отложен на потом[1]. Так же существует режим Advance reconstruction, суть которого заключается в следующем. Чтение с дисков — медленная операция. Более того, некоторые диски могут быть заняты чем-то еще и не ответить нам сразу. Тогда мы можем начать процесс восстановления данных на медленных дисках не дожидаясь ответа с них. Такой подход может увеличить быстродействие системы, однако мы заведомо отказываемся от возможности нахождения SDC.

Общая структура модуля

Наш модуль работает непосредственно с одним страйпом. Для его полноценной работы необходимо наличие трех основных функций.

- Расчет синдромов.
- Восстановление утраченной информации на дисках.
- Поиск скрытых повреждений.

Первая и последняя задачи сами по себе имеют не малый потенциал для изучения и исследования, мы не будем подробно на них останавливаться. Однако, так как нам для восстановления понадобятся уже готовые контрольные суммы, скажем о них пару слов. Поскольку вычисления основываются на систематических кодах Рида — Соломона, на синдромы накладывается следующее свойство. Пусть D_i — блок с данными, а S_i — блок с синдромами, тогда все контрольные суммы, подсчитанные от всего массива $(D_0, ..., D_{n-1}, S_0, ... S_{m-1}) = (Y_0, ..., Y_{N-1})$ должны равняться нулю. Контрольные суммы от всей последовательности блоков рассчитываются следующим образом:

$$\tilde{S}_j = \sum_{i=0}^{N-1} Y_i a^{j(N-i-1)}, \quad j = 0, ..., m-1,$$

где a — примитивный элемент поля. То есть если все диски целы, то все $\tilde{S}_j = 0$. Имея такого рода синдромы, можно произвести восстановление утраченных дисков.

Алгоритмы восстановления

Обращение матрицы

Пусть k_0, …, k_{l-1} — номера отказавших блоков, l − их число. Рассмотрим расчет контрольных сумм как умножение матрицы на вектор. Так как нам заранее известны номера сбитых дисков, то мы можем обнулить из значения или, что проще, не рассматривать соответствующие блоки при умножении:

$$\begin{pmatrix} 1 & 1 & \dots & 1 & 1 & \dots & 1 & 1 \\ a^{N-1} & a^{N-2} & \dots & a^{N-k_i} & a^{N-k_i-2} & \dots & a & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a^{(l-1)(N-1)} & a^{(l-1)(N-2)} & \dots & a^{(l-1)(N-k_i)} & a^{(l-1)(N-k_i-2)} & \dots & a^{l-1} & 1 \end{pmatrix} \begin{pmatrix} Y_0 \\ Y_1 \\ \dots \\ Y_{k_i-1} \\ Y_{k_i+1} \\ \dots \\ Y_N \end{pmatrix} = \begin{pmatrix} \tilde{S}_0 \\ \tilde{S}_1 \\ \dots \\ \tilde{S}_{l-1} \end{pmatrix}.$$

Тогда восстановление будет выглядеть следующим образом:

$$\begin{pmatrix} Y_{k_0} \\ Y_{k\_1} \\ \dots \\ Y_{k_l} \end{pmatrix} = \begin{pmatrix} 1 & 1 & \dots & 1 \\ a^{N-k_0-1} & a^{N-k_1-1} & \dots & a^{N-k_{l-1}-1} \\ \dots & \dots & \dots & \dots \\ a^{(l-1)(N-k_0-1)} & a^{(l-1)(N-k_1-1)} & \dots & a^{(l-1)(N-k_{l-1}-1)} \end{pmatrix}^{-1} \begin{pmatrix} \tilde{S}_0 \\ \tilde{S}_1 \\ \dots \\ \tilde{S}_{l-1} \end{pmatrix}.$$

Таким образом, стандартный алгоритм восстановления будет заключаться в умножении на обращенную матрицу Вандермонда. Нам потребуется $l(N-l)$ умножений и $l(N-l-1)$ сложений для расчета контрольных сумм и l^2 умножений и $l(l-1)$ сложений для второго шага. Так же нам необходимо найти обратную матрицу, что довольно трудоемко.

Однако, можно воспользоваться тем, что наша матрица является матрицей Вандермонда, а про обращение таких матриц известны некоторые полезные свойства. Мы не будем подробнее рассматривать этот алгоритм тут, его исследованием занимался мой коллега. Здесь мы чуть позже приведем лишь оценку по количеству операций, необходимых для восстановления по этому алгоритму.

Алгоритм Форни

Следующий алгоритм основан больше на свойствах нашего поля, а именно на действиях с полиномами в конечных полях Галуа. Его обоснование и вывод формул можно посмотреть в книге Берлекэмпа[2]. Для начала составим вспомогательный полином, корни которого $a^{-(N-k_0-1)}$, ..., $a^{-(N-k_{l-1}-1)}$:

$$\sigma(x) = \prod_{i=1}^{l} (1 + xa^{-(N-k_i-1)}) = \sum_{i=0}^{l} \sigma_i x^i.$$

Рассмотрим ключевое уравнение Ω, коэффициенты которого подсчитаны следующим образом:

$$\begin{pmatrix} \sigma_{l-1} & \sigma_{l-2} & \dots & \sigma_1 & \sigma_0 \\ \sigma_{l-2} & \sigma_{l-3} & \dots & \sigma_0 & 0 \\ \sigma_{l-3} & \sigma_{l-4} & \dots & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ \sigma_0 & 0 & \dots & 0 & 0 \end{pmatrix} \begin{pmatrix} \tilde{S}_0 \\ \tilde{S}_1 \\ \dots \\ \tilde{S}_{l-2} \\ \tilde{S}_{l-3} \end{pmatrix} = \begin{pmatrix} \grave{U}_{l-1} \\ \grave{U}_{l-2} \\ \dots \\ \grave{U}_1 \\ \grave{U}_0 \end{pmatrix}.$$

Введем еще одно обозначение:

$$\lambda_j = \left(\prod_{i=1, i \neq j}^{l} (1 + a^{k_j - k_i}) \right)^{-1}.$$

Используя эти обозначения можно восстановить данные по следующей формуле:

$$Y_{k_j} = \tilde{Y}_{k_j} + \grave{U}(a^{-(N-k_j-1)})\lambda_j,$$

где $\tilde{Y}_{k_j}$ — данные в сбитых блоках.

Сравнение алгоритмов

Необходимо проанализировать количество операций сложения и умножения для каждого алгоритма. Расчет идет следующим образом. Каждый блок в страйпе бьется на полоски (line) по 16 байт при использовании SSE и по 32 байта при использовании AVX. За один проход алгоритма обрабатывается по одной полоске с каждого блока. В каждом алгоритме есть еще вспомогательные вычисления, которые можно провести для всего страйпа сразу, так как они не зависят от данных на дисках.

Ниже представлена таблица, иллюстрирующая количество операций, необходимых для восстановления сбитых дисков тем или иным алгоритмом.

Таблица 1

Сравнение алгоритмов

Название алгоритма	**Число умножений**	**Число сложений**	**Дополнительные действия**
Стандартный алгоритм	Nl	$l(N-2)$	Обращение матрицы Вандермонда размером $l \times l$
Алгоритм с Вандермондом	$l(N-l)$	$l(N-l-1)$	Построение вспомогательной матрицы
Алгоритм Форни	$l\left(N + \frac{l+1}{2}\right)$	$l\left(N + \frac{l+5}{2}\right)+1$	Построение полинома с заданными корнями

Видно, что различные методы требуют разное количество операций сложения и умножений, к тому же, дополнительные действия иногда довольно

трудоемки. По этим причинам нельзя с точностью сказать, какой подход даст лучший результат. Далее в этой статье мы рассмотрим реализацию алгоритма Форни и сравним его по производительности со страндартным алгоритмом и алгоритмом, использующим свойства матрицы Вандермонда.

Реализация

Первый вопрос, который возникает, это вопрос о быстром построении полинома $\sigma(x)$ по его корням. Существует алгоритм, позволяющий нам решать эту задачу без использования дополнительной памяти за $O(l)$. Пусть $(q_1^{-1}, ..., q_l^{-1})$ — корни многочлена $\sigma(x)$. Будем строить $\sigma(x)$ итеративно:

$$\sigma_{1(x)} = 1 + xq_1,$$

$$\sigma_i(x) = (1 + xq_i)\sigma_{i-1}(x),\ i = 2 \ldots l.$$

Таким образом, корнями полинома $\sigma_l(x)$ степени l будут $(q_1^{-1}, .., q_l^{-1})$, то есть мы построили искомый многочлен. Распишем подробнее i-ый шаг:

$$\sigma_i(x) = (1 + xq_i)\sigma_{i-1}(x) = \sigma_{i-1}(x) + xq_i\sigma_{i-1}(x).$$

Если представить эти полиномы как массивы, где на -ом месте стоит коэффициент при x^i, то умножение многочлена на x это сдвиг всех коэффициентов влево. Затем все коэффициенты следует умножить на q_j и сложить с массивом до преобразований. Очевидно, этот алгоритм легко векторизовать с использованием инструкций SSE или AVX, что является его дополнительным достоинством.

Второй вопрос заключался в том, как считать λ_j, а точнее можно ли хранить заранее посчитанные значения λ_j? Казалось бы, они не зависят от данных в блоках и их можно заранее вычислить, записать в таблицу и потом к ним обращаться. λ_jзависят от номеров сбитых дисков и их количества, к сожалению, их число получается довольно большим, а хранить такие большие таблицы не целесообразно. Однако, можно хранить значения $(1 + a^{k_j - k_i})^{-1}$. Этих значений всего 256, перемножив нужные из них в зависимости от номеров сбитых дисков можно получить искомые λ_j.

Следующее решение, которое необходимо было принять, было о том, как считать Ω_j. Понятно, что реализовывать честное умножение матрицы такого рода на вектор в данном случае не желательно по памяти и производительности. Можно обойтись умножением векторов заданной длины, и при подсчете каждого коэффициента Ω уменьшать длину, передаваемую в функцию умножения. Тем самым мы сэкономим память, не создавая целую матрицу, и время, не производя умножения на нулевые элементы.

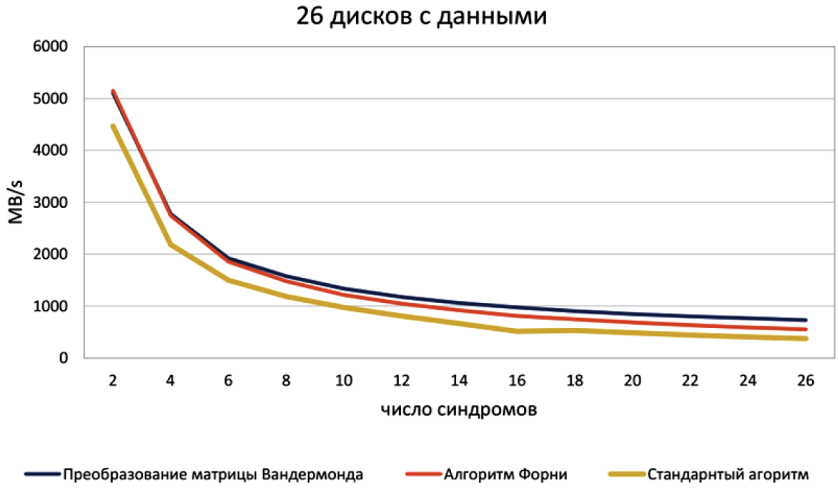
Последний самый главный вопрос заключался в том, в каком порядке производить восстановление. Пусть у нас подсчитаны контрольные суммы и все вспомогательные значения, а именно, найдены многочлены Ω и σ и зна-

чения λ_j. Тогда существует два подхода: либо восстанавливать по блокам (сначала полностью восстановить первый блок, потом второй и так далее), либо как и раньше восстанавливать по 16 байт в каждом блоке. Плюсом первого подхода является то, что нам не придется хранить $\hat{U}(a^{-(N-k_j-1)})$ для каждого k_j, однако, это нарушает логику нашей работы и из-за особенностей кеширования может пагубно сказаться на производительности. Как и получилось на практике.

Результаты

Нами были реализованы все три алгоритма, то есть у нас есть возможность сравнить их производительность на практике. Ниже представлены графики зависимости скорости восстановления от числа контрольных сумм для 26 дисков с данными. Число отказавших дисков совпадало с числом контрольных сумм, но алгоритмы работают корректно и для меньшего числа отказавших дисков.

Получилось, что алгоритм Форни и алгоритм, основанный на свойствах матрицы Вандермонда заметно лучше стандартного. Алгоритм Форни при небольшом числе синдромов выигрывает до 10% по производительности, однако с ростом числа синдромов этот выигрыш уменьшается и в определенный момент алгоритм Форни начинает проигрывать. Также существенным недостатком алгоритма является использование дополнительной памяти для хранения Ω.



Нашим полем накладываются ограничение на m и n: $n+m<256$ и $m\leq 64$ из-за использования SSE. Однако, m может быть увеличено в дальнейшем.

Заключение

Мы рассмотрели различные алгоритмы восстановления данных в RAID $m+n$ и особенности реализации одного из самых перспективных алгоритмов. Благодаря командной работе мы смогли сравнить эти алгоритм на практике увидеть, что они не слишком отличаются. Однако, существуют некоторые алгебраические оптимизации, которые могли бы улучшить тот или иной алгоритм. Развитие идей этих оптимизаций, их корректность и применимость в данной области будет рассмотрена нами в дальнейшем.

Литература

1. *Brian Beach*. What Hard Drive Should I Buy? http://blog.backblaze.com/2014/01/21/what-hard-drive-should-i-buy
2. *Берлекэмп Э*. Алгебраическая теория кодирования. М.: Мир, 1971. 748с.
3. *Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К*. М.: ИД «Вильямс», 2005. 1296 с.
4. *Утешев А*. Поля Галуа. http://pmpu.ru/vf4/gruppe/galois

РАСПОЗНАВАНИЕ ПРИЛОЖЕНИЙ ПО ЗАПРОСАМ БЛОЧНОГО УРОВНЯ[1]

И. И. Демьяненко

студент кафедры системного программирования СПбГУ

E-mail: ilya@demyanenko.pro

Аннотация. Задача предоставления QoS (Quality of Service) является перспективным направлением в области хранения данных.

Данная работа посвящена распознаванию клиентских приложений по их запросам к СХД с целью автоматического управления приоритетами обслуживания.

Введение

Системы хранения данных используются в разных областях, в том числе в сфере работы с мультимедиа. Рынок мультимедиа (производство фильмов, телепередач) специфичен тем, что многие задачи имеют жёсткие сроки, в которые их необходимо выполнить. При этом с увеличением разрешения и количества видеоматериалов объёмы и число задач постоянно растут, а возможности СХД конечны, поэтому клиентам хочется распределять ресурсы в соответствии с важностью задач.

Компания RAIDIX предоставляет программное обеспечение для систем хранения данных, которое даёт возможности по управлению Quality of Service. Администратор может предоставить приоритет одному или нескольким инициаторам, тогда запросам от них будет предоставлена гарантированная пропускная способность.

Недостатком существующей реализации является необходимость управлять приоритетом вручную, что не всегда удобно. Задачи могут распределяться по инициаторам динамически, тогда придётся часто вносить изменения в список приоритетного обслуживания. Также существует дефицит грамотных администраторов, из-за чего компаниям приходится нести повышенные расходы на их содержание. Решение о том, предоставлять ли приоритет инициатору, принимается на основе задач, которые он выполняет в данный момент, поэтому, если знать, какие приложения в данный момент исполняются на клиентских компьютерах и генерируют нагрузку на СХД, можно принять решение о предоставлении приоритетов автоматически.

[1] Работа выполнена по заказу компании RAIDIX

Исходные данные

ПО RAIDIX предоставляет блочный доступ к RAID-массивам, поэтому в качестве исходных данных доступен поток запросов следующего вида:

- имя инициатора;
- время поступления запроса в микросекундах;
- тип запроса (чтение / запись);
- название RAID-массива;
- адрес;
- размер операции в секторах.

Постановка задачи

В рамках работы были рассмотрены следующие задачи:

- поиск закономерностей в поведении приложений;
- изучение способов распознавания приложений с использованием найденных закономерностей;
- разработка прототипа распознающего модуля и тестирование его эффективности.

Выборочное распознавание

Описание

Поскольку задача ориентирована на мультимедиа-приложения, имеет смысл рассмотреть характерные особенности именно этого класса программ. Первой гипотезой было то, что приложения, работающие с мультимедиа, генерируют в основном последовательную нагрузку на СХД, что подтвердилось исследованиями бенчмарков, которые эмулируют работу видеоредакторов (Blackmagic Disk Speed Test, AJA System Test). Это позволяет ограничить класс распознаваемых приложений так, чтобы с ним было удобно работать.

Сузим круг до приложений, генерирующих линейную нагрузку и оперирующих блоками одинакового размера, которые, в случае бенчмарков, соответствуют кадрам видео. Условия значительно облегчают задачу, однако, после того, как запрос сгенерирован приложением, он проходит через стек ввода-вывода операционной системы и драйвер SCSI, в ходе чего может быть произвольным образом разбит на несколько запросов. Разрабатываемая система распознавания должна учитывать эти искажения.

Алгоритм работы

Я пришёл к системе, состоящей из двух компонентов: учителя и распознавателя.

Учитель:

1. запускается в момент, когда работает только целевое приложение;
2. группирует запросы по последовательностям, которые соответствуют операциям линейного чтения/записи;
3. для каждой последовательности вычисляет размер исходных блоков и возможные варианты их разбиения транспортом;
4. если не удаётся найти размер блока, покрывающий 70% последовательности, считает, что последовательность нам не подходит;
5. если удалось найти размер исходного блока для 90% всех запросов, считает, что обучение удалось;
6. сохраняет информацию о размерах блоков и вариантах их разбиения для дальнейшего использования распознавателем.

Распознаватель:

1. обрабатывает поступающие на СХД запросы в потоковом режиме;
2. ведёт учёт последовательностей за последние N секунд;
3. каждый запрос либо продолжает известную последовательность, либо создаёт новую:
   3.1. если запрос продолжает текущую последовательность, происходит её обновление;
   3.2. в противном случае на его основе создаётся кандидат в последовательности. Кандидат становится полноценной последовательностью, когда он полностью соответствует нескольким блокам от одного из известных приложений; в этом случае распознаватель считает, что обнаружил новое приложение;
4. если какая-то из последовательностей не продолжалась в течение N секунд, она удаляется;
5. если у приложения не осталось работающих последовательностей, оно считается завершившимся.

Преимущества и недостатки

Преимущества:

- алгоритм является детерминированным (основан на чётких данных);
- приложения из целевого класса определяются безошибочно.

Недостатки:

- ориентированность на узкий класс приложений;
- как выяснилось позже, реальные приложения для работы с видео ведут себя не так, как бенчмарки.

Вспомогательный клиентский модуль

Если мы хотим распознавать приложения со 100% достоверностью, можно задуматься о сборе дополнительной информации с клиента. Модуль, работающий на клиенте и перехватывающий системные вызовы приложений, имеет доступ к достаточному количеству информации для гибкого управления приоритетами. После сбора статистики по запросам приложений за небольшой промежуток времени модуль может отправить её на СХД либо по сети, либо с помощью собственных SCSI-команд. В этом случае СХД будет знать, какие приложения исполняются на СХД, какую нагрузку они генерируют, и с файлами какого типа они работают. Из файлов при этом могут извлекаться метаданные, что позволит различать даже работу с видео различного разрешения и автоматически определять необходимую пропускную способность по битрейту видео.

К преимуществам этого способа, помимо перечисленных выше, относятся низкие накладные расходы. На клиенте ведётся простой учёт операций, а на стороне СХД принимается решение по чётким правилам. Но также есть и недостатки: клиентскую часть придётся реализовывать под разные платформы (по меньшей мере, три), вносить изменения при некоторых обновлениях ОС, а также устанавливать дополнительное ПО на все клиентские компьютеры, что создаёт дополнительную работу для администраторов.

Машинное обучение

Описание

Машинное обучение используется во многих областях при решении задач любой степени сложности и имеет большие перспективы. Его суть заключается в том, что по тренировочному набору данных составляются вектора, хорошо характеризующие данные, затем на этом наборе векторов обучается модель. Далее по данным, которые нужно классифицировать, строятся вектора того же вида, которые распознаются моделью.

После анализа разных трейсов с реальных мультимедиа-приложений я обнаружил, что большинство из них всё же производит операции последовательного чтения/записи. Рисунок 1 показывает это на примере Apple Final Cut.

При детальном рассмотрении оказывается, что запросы не совсем последовательны, то есть, конец предыдущего запроса не всегда совпадает с началом следующего (Рисунок 2). Тем не менее, если незначительно ослабить условие линейности, выясняется, что 90% запросов от реальных видеоредакторов относятся к последовательным операциям.

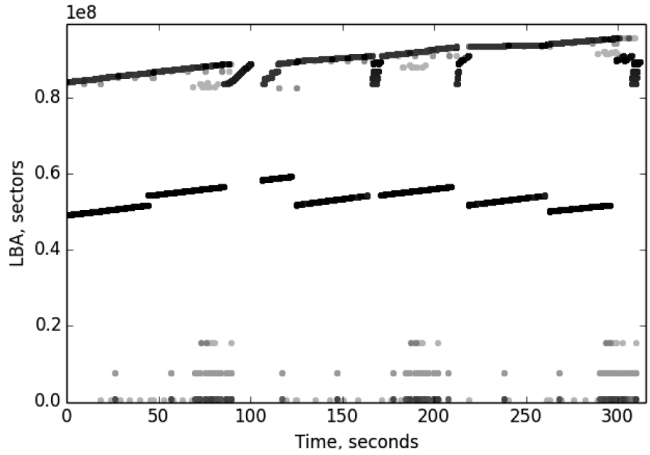


Рис. 1. Запросы к СХД от Apple Final Cut

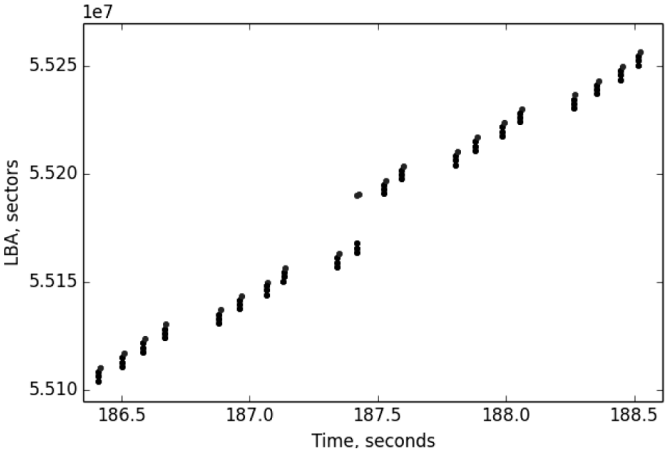


Рис. 2. Запросы к СХД от Apple Final Cut, уровень отдельных операций

Алгоритм

В итоге я пришёл к следующему алгоритму обучения:

1. Во множестве запросов производится поиск последовательностей по ослабленному условию.
2. Каждая последовательность разбивается на интервалы по 20 секунд (целевое время отклика алгоритма).
3. На каждом интервале для каждого из 2048 различных размеров запросов собирается набор статистик:

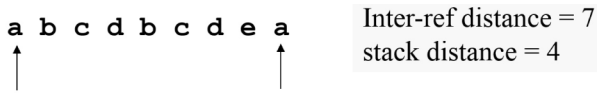


Рис. 3. inter-reference distance, stack distance

3.1. размер запроса;
3.2. доля запросов этого размера среди всех запросов интервала;
3.3. mean inter-reference distance — среднее число запросов между соседними запросами выбранного размера (Рисунок 3);
3.4. mean reuse distance[1]. Эта статистика пришла из алгоритмов кэширования, где она равна числу обращений к уникальным адресам между двумя обращениями к конкретному адресу (Рисунок 3). В моём случае вместо адресов используются размеры запросов, полученный результат для каждого размера усредняется по интервалу.

4. Набор статистик для каждого интервала объединяется в вектор.

Поскольку получилось много векторов, и многие вектора из соседних интервалов похожи, была предпринята попытка их кластеризации. Но из-за высокой размерности данных их не смогли кластеризовать за разумное время алгоритм k-means[2] и нейронная сеть Кохонена[3] из пакета WEKA. Также из-за неоднородности данных сложно выбрать метрику, которая позволила бы адекватно оценивать близость векторов.

Заключение

Было рассмотрено три подхода к решению задачи распознавания приложений.

Реализован модуль выборочного распознавания. Хотя он и ориентирован на узкого класса приложений, бенчмарки и операции резервного копирования определяются им безошибочно, что также имеет ценность для обеспечения QoS.

Подход со вспомогательным клиентским модулем рассмотрен с теоретической стороны, определены его достоинства и недостатки. Если признать разумной затрату ресурсов на реализацию и развёртывание клиентского модуля, этот способ будет самым надёжным, гибким и наименее требовательным к ресурсам СХД.

Машинное обучение является наиболее перспективным решением на стороне СХД. Возникшие проблемы могут быть преодолены путём уменьшения размерности векторов каким-либо способом, а также использованием алгоритмов, не нуждающихся в метриках.

Литература

1. *R. Mattson, J. Gecsei, D. Slutz, I. Traiger*. Evaluation Techniques for Storage Hierarchies // IBM Systems Journal. Vol. 9. No. 2. 1970.
2. *MacQueen J*. Some methods for classification and analysis of multivariate observations // Proc. Fifth Berkeley Symp. on Math. Statist. and Prob. Vol. 1. 1967. Pp. 281–297.
3. *Kohonen T*. Self-organized formation of topologically correct feature maps // Biological Cybernetics. Vol. 43. 1982. С. 59–69.

РАЗРАБОТКА ИМИТАЦИОННОЙ МОДЕЛИ ПРОИЗВОДИТЕЛЬНОСТИ МОДУЛЬНЫХ СИСТЕМ ХРАНЕНИЯ ДАННЫХ ACTIVE-ACTIVE

М. М. Заславский

магистрант кафедры «Системный анализ и управление», СПбГПУ

E-mail: mark.zaslavskiy@gmail.com

Аннотация. На сегодняшний день требования к быстродействию СХД растут быстрее, чем производительность отдельных элементов. Поэтому особую важность приобретает задача анализа производительности. В данной работе рассматривается имитационная модель модульной СХД Active-Active, позволяющая решить данную задачу с использованием теории конечных автоматов. Для упрощения в модели используется разделение общего состояния системы на набор подсостояний ее элементов и отдельных запросов. Составленная модель позволяет анализировать производительность системы под воздействием различных шаблонов нагрузки и при изменении параметров системы в течение численного эксперимента, оценивать влияние алгоритмов повышения производительности на работу системы.

Введение

С появлением высокоскоростных интерфейсов и технологий организация модульных СХД с использованием архитектуры Active-Active становиться все более привлекательным. Однако в ряде случаев достигаемого выигрыша за счет их внедрения не достаточно — необходима настройка параметров системы для конкретных вариантов ее использования. Применение экспериментального исследования производительности для определения наилучшей конфигурации связано с рядом проблем — оно требует реального оборудования, которое должно максимально точно воспроизводить исследуемой системы. Кроме того, с учетом постоянного развития интерфейсов и протоколов передачи данных между элементами «StorageProcessor» обеспечение их полной поддержки в испытательном стенде приведет к существенным финансовым затратам. Построение имитационной модели производительности позволит избежать данных проблем, так как ее использование требует меньших вычислительных затрат и не требует использования реального аппаратного обеспечения. В данной статье рассматривается реализация симулятора системы хранения данных, позволяющего исследовать параметры алгоритмов кэширования в архитектуре Active-Active.

Исследуемая система

Исследуемая система представляет собой модульную СХД, организованную по принципу Active-Active. Для упрощения модели, было принято допущение о том, что все запросы в системе это запросы на чтение.

Рассмотрим отдельные элементы исследуемой системы:

- «Приложение» представляет собой пользователя системы. Данный элемент генерирует нагрузку в соответствии с одним из шаблонов нагрузки — последовательным или случайным. Для случайного шаблона возможно наличие временной или пространственной локальностей.
- «Балансировщик нагрузки» осуществляет распределение запросов Приложения к обработчикам из числа элементов «StorageProcessor» по одному из двух алгоритмов — Round Robin или Least Queue Depth [1].
- «StorageProcessor» представляет собой элемент, реализующий один из трех сценариев обработки входящих запросов:
  - Независимая работа — оба элемента «StorageProcessor» работают независимо друг от друга. Кэш-промах запроса приводит к тому, что данные запрашиваются из элемента «RAID».
  - Оптимистичное кэширование подразумевает независимую работу «StorageProcessor», однако при кэш-промахе данные сначала запрашиваются из кэш-памяти другого элемента «StorageProcessor» через шину Interconnect, в случае повторного кэш-промаха данные запрашиваются из элемента «RAID».

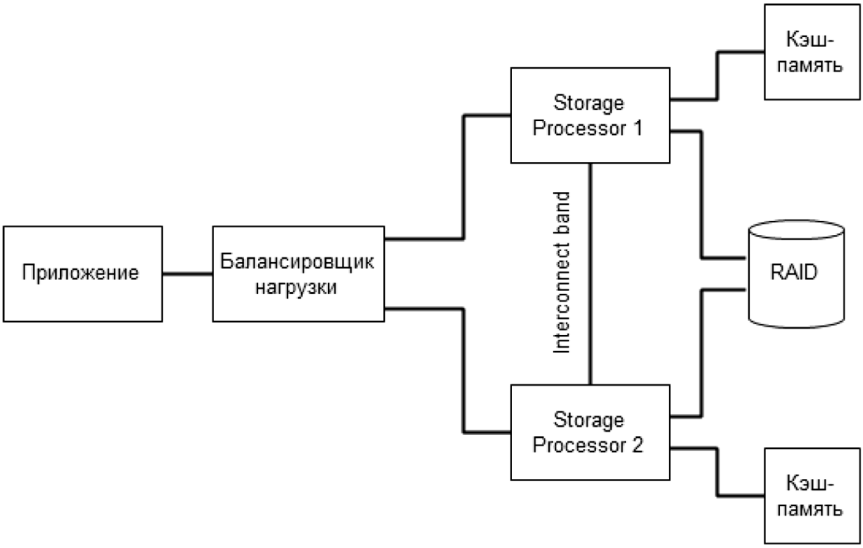


Рис. 1. Общая схема исследуемой системы

- Когерентность — обеспечение когерентности между кэш-памятью обоих элементов «StorageProcessor.»
- Кэш-память представляет собой блок памяти с небольшим значением времени доступа относительно элемента «RAID». Вытеснение данных осуществляется по алгоритму Least Recent Use [2].
- Элемент «RAID» представляет собой массив RAID1 с двумя независимыми интерфейсами доступа, данные в котором разбиты на страйпы размером 128 кб.

Допущения

При построении математической модели использовались следующие допущения:

1. Все запросы, генерируемые элементом «Приложение», представляют собой корректные запросы на чтение.
2. Моделирование осуществляется в дискретном времени.
3. Шины между элементами «StorageProcessor» и между элементами «StorageProcessor» и «RAID» имеют ограниченную пропускную способность (количество запросов которые могут быть переданы за один такт по шине), все прочие каналы передачи данных имеют бесконечную пропускную способность в течение одного такта.
4. Переход запроса между двумя любыми элементами осуществляется с ненулевой задержкой.
5. За один такт запрос может побывать не более чем в двух элементах.

Математическая модель

Функционал производительности

Будем определять производительность системы как функционал следующего вида:

$$\begin{aligned} &P(y) = a_1 M(y) + \frac{a_2}{L(y)}, \\ &a_1, a_2 \in \mathbb{R}^+, \\ &P, M, L \in \mathbb{R}, \\ &y = \{y_i\}_{i=1}^{\infty}, y_i \in \mathbb{R}^3, \\ &y_i = (y_{i1}, y_{i2}, y_{i3}). \end{aligned} \tag{1}$$

где P — функционал производительности системы; M — количество переданных данных в единицу времени; L — задержка обработки запроса; a_1, a_2 — весовые коэффициенты; y — последовательность векторных характеристик обработанных запросов (y_{i1} — количество обработанных запросов

на i-ом шаге, y_{i2} — количество данных переданных на i-ом шаге, y_{i3} — суммарная задержка обработки запросов на i-ом шаге).

Реализация симулятора

Построенная модель представляет собой конечный автомат, с правилами перехода между состояниями, задаваемыми алгоритмически. Однако, так как количество состояний системы велико, то предлагается разбить их на отдельные группы согласно логике обработки запросов.

Общее состояние системы разбивается на состояния отдельных **запросов**, **элементов** и **параметров системы**. Элементы системы — «Приложение», «Балансировщик нагрузки», «StorageProcessor», «RAID».

Состояние запроса представляет собой вектор:

$$\begin{aligned} & r = (a, l, c, e, x) \in R, \\ & a \in \mathbb{N}, \\ & l \in \mathbb{R}^{+}, \\ & c \in \{0, 1\}, \\ & x \in \{0, 1\}, \\ & e \in \{0, 1, 2, 3, 4\}, \end{aligned} \tag{2}$$

где a — запрашиваемый адрес; l — текущая задержка обработки в миллисекундах; e — номер элемента, в котором находится запрос на данном шаге; c — номер элемента «StorageProcessor», которому был первоначально адресован запрос; x — направление движения запроса (1 — от элемента «Приложение», 0 — к элементу). Соответствие номера e конкретному элементу устанавливается таблицей:

Таблица 1

Соответствие номеров конкретным элементам модели

Номер элемента	Название элемента
0	Приложение
1	Балансировщик нагрузки
2	StorageProcessor0
3	StorageProcessor1
4	RAID

Каждый элемент системы можно представить в виде:

$$(e, \alpha, \beta) \in E, \tag{3}$$

где β — текущее состояние элемента с номером e; $\alpha = \alpha(r, \beta, i) = \{r', \beta'\}$ — правило, по которому состояние запроса r и состояние β элемента с номером e

преобразуется на шаге *i*. Далее рассмотрим состояния конкретных элементов системы.

Состояние элемента «Приложение» задается натуральным числом N_i, обозначающим номер текущего шага моделирования.

Вид состояния элемента «Балансировщик нагрузки» зависит от используемого алгоритма балансировки нагрузки. В случае алгоритма Least Queue Depth [1], состояние представляет собой вектор

$$\begin{aligned} &B_i = (b_1^i, b_2^i), \\ &b_1^i, b_2^i \in \mathbb{Z}, \end{aligned} \tag{4}$$

где b_j^i — количество запросов, обрабатываемое *j*-ым элементом «StorageProcessor» на *i*-ом шаге.

В случае использования алгоритма Round Robin, состояние определяется величинами

$$D_i = (d_1^i, d_2^i, d_3^i, d_4^i) \in \mathbb{Z}^4, \tag{5}$$

где d_1^i — номер текущего шага; d_2^i — максимальное число запросов, отсылаемых подряд на один элемент «StorageProcessor»; d_3^i — номер элемента «StorageProcessor», на который отсылаются запросы; d_4^i — число запросов, отосланных подряд на элемент «StorageProcessor» под номером d_3^i.

Состоянием элемента «StorageProcessor» является совокупность состояний собственно StorageProcessor и состояния кэш-памяти. G_i — количество обработанных запросов, ожидающих пересылки по шине Interconnect другому элементу «StorageProcessor».

Состоянием Кэш-памяти является вектор следующего вида:

$$K_i = \left(k_1, ..k_m\right) \in \mathbb{Z}^m \tag{6}$$

где *m* — размер кэш-памяти, K_i — отсортированное по дате последнего доступа множество элементов кэш-памяти на *i*-ом шаге.

Состоянием элемента «RAID» является вектор

$$H_i = (h_1^i, h_2^i) \in \mathbb{Z}^2,$$

где h_j^i — количество обработанных запросов ожидающих пересылки *j*-ому элементу «StorageProcessor».

Параметры системы представляют собой совокупность величин задержек переходов между состояниями, величины кэш-памяти, настроек алгоритмов балансировки нагрузки, величины блока чтения, настройки локальности случайного чтения, количество полос в RAID-массиве. Параметры системы могут быть изменены в процессе моделирования при наличии затрагивающего их **события**. Событие представляет вектор вида

$$p = \left(p_1, p_2, p_3\right) \in P, \tag{7}$$

где P — множество событий модели; p_1 — номер шага моделирования, на котором событие должно произойти; p_2 — номер параметра системы, который будет изменен; p_3 — новое значение параметра p_2.

Шаблоны нагрузки

В предложенной модели рассматриваются два шаблона доступа к данным — последовательный и случайный. Для случайного шаблона доступно задание определенного типа локальности либо его отсутствия.

При последовательном доступе производится генерация запросов к большим блокам данных. На каждом шаге моделирования создается n новых запросов, адреса которых задаются последовательно, начиная с числа a_0 — равномерно распределенной случайной величины в диапазоне $[0, m]$, где m — количество полос в RAID-массиве. Величины n и m являются параметрами модели, поэтому могут быть изменены под действием событий.

При случайном доступе производиться генерация k запросов к отдельным полосам, адреса которых представляют собой равномерно распределенной случайной величины в диапазоне $[0, m]$. Величина k представляет собой параметр модели. При наличии определенного типа локальности, генерация адресов изменяется следующим образом:

1. В случае пространственной локальности с вероятностью u очередной адрес полосы определяется как равномерно распределенная случайная величина из промежутка $[v_1, v_2]$, $0 \le v_i \le m$, где v_i — параметры модели, задающие размер области для выборки адресов пространственной локальности; u — параметр модели, задающий вероятность.
2. В случае временной локальности, с вероятностью y очередной адрес полосы определяется как один из χ последних адресов полос, сгенерированных элементом «Приложение». Величины y и χ являются параметрами модели.

Алгоритм моделирования

Вычисление нового состояния модели на основании предыдущего состояния производиться в три этапа.

На **первом этапе** происходит генерация новых запросов и изменение параметров. Согласно текущему шаблону нагрузки происходит построение N_i — множества новых запросов, сгенерированные элементом «Приложение». Из множества P выбирается подмножество P_i, которое содержит все события для текущего шага:

$$P_i = \{ p \in P \mid p_1 = i \}. \tag{8}$$

Далее, для каждого элемента P_i производится изменение значения параметра p_2 на значение p_3.

На **втором этапе** происходит смена состояний всех незавершенных запросов. Незавершенные запросы на *i*-ом шаге образуют множество Q_i . Оно определяется по следующему рекуррентному соотношению:

$$Q_i = N_i \cup T_{i-1}, \tag{9}$$

где N_i — новые запросы, сгенерированные элементом «Приложение», на *i*-ом шаге, T_{i-1} — обработанные и не завершенные запросы на шаге $i-1$. Таким образом, осуществляется следующая процедура по пересчету состояний запросов:

$$\overline{T_i} = \left\{ r' \in R \mid \{r', \beta'\} = \alpha\ (r, \beta, i),\ r \in Q_i \right\}, \tag{10}$$

где β — состояние элемента с номером *e* запроса *r*.

На **третьем этапе** строиться множество T_i путем удаления завершенных запросов:

$$\begin{aligned} T_i &= \overline{T_i} \setminus F_i, \\ F_i &= \{r \in \overline{T_i} \mid e = 0,\ x = 0\}, \end{aligned} \tag{11}$$

то есть таких, у которых текущий номер элемента *e* соответствует номеру элемента «Приложение» и направление движение *x* соответствует движению к элементу «Приложение».

Алгоритмы повышения производительности

Из трех режимов обработки кэш-промахов в разделе «Исследуемая система» наиболее привлекательным с точки зрения производительности является режим с пересылкой запросов по шине Interconnect, так как он увеличивает доступную обоим элементам «StorageProcessor» емкость кэш-памяти в обмен на дополнительное время задержки, которое затрачивается на пересылку данных по шине Interconnect. Однако данный подход имеет определенные недостатки — так как шина между элементами «StorageProcessor» имеет ограниченную пропускную способность, то в ней возможно возникновение очередей запросов. Так как ожидание в очереди тоже приводит к росту задержки, то возможна ситуация, когда суммарное время отправки и ожидания превысит время отправки запроса напрямую к элементу «RAID». Поэтому предлагается четвертый вариант обработки кэш-промахов — принятие решений о необходимости пересылки запроса по шине Interconnect на основе данных о загруженности этой шины.

Алгоритм принятия решений имеет два параметра, A — максимальный размер очереди, после достижения которого начинается принудительная пересылка запросов напрямую элементу «RAID»; B — количество тактов после обнаружения превышения уровня A, в течение которых запросы будут принудительно отправляться элементу «RAID». Блок-схема алгоритма принятия решений:

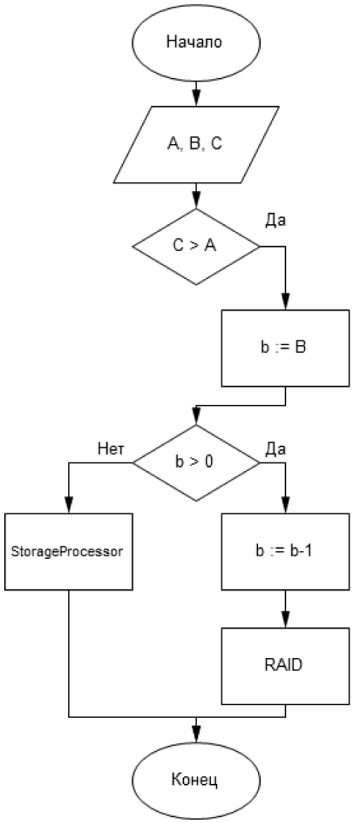


Рис. 2. Схема алгоритма принятия решений
о пересылке запросов по шине Interconnect

Анализ производительности с помощью модели

Для проведения исследования была написана программа на языке C++, позволяющая вычислять состояния модели при задаваемых пользователях параметрах и ограничениях. С помощью данного инструмента было поставлено несколько численных экспериментов по определению функционала производительности для последовательного чтения с разными размерами блоков. Вычислялся как сам функционал производительности $P(y)$, так $M(y)$ и $L(y)$, положив $a_1 = 1$, $a_2 = 1$. Эксперименты проводились на системе с отключенным и включенным алгоритмом управления при различных значениях параметров A, B и различном размере блока чтения. Прочие параметры были выбраны одинаковыми для всех экспериментов:

Таблица 2

Основные параметры модели

Параметр	Значение
Алгоритм балансировки нагрузки	Round Robin
Нагрузка на один узел, страйпы	1
Размер RAID-массива, страйпы	1000
Задержка ожидания в течение одного такта для шины Interconnect, мкс	200
Задержка ожидания в течение одного такта для шины «RAID» — «StorageProcessor», мкс	200
Размер кэш-памяти, страйпы	50
Максимальная пропускная способность шины Interconnect в течение одного такта, страйпы	3
Максимальная пропускная способность шины «RAID» — «StorageProcessor» в течение одного такта, страйпы	10
Задержка чтения данных в элементе RAID, мкс	4000
Задержка чтения данных из кэш-памяти, мкс	250
Задержка передачи данных по шине Interconnect, мкс	200

Результаты моделирования поведения систем с различными настройками приведены в Таблице 2 и Таблице 3.

Таблица 3

Результаты эксперимента для размера блока, равного 10 страйпам

Алгоритм принятия решений	B, такты	A, запросы	P	M, страйпов за такт	L, сек.
Выкл.	–	–	173,03	172,51	0,19
Вкл.	10	10	183,21	182,82	0,26
Вкл.	10	20	182,34	181,95	0,26
Вкл.	20	10	182,48	182,10	0,26
Вкл.	20	20	182,36	181,96	0,25

Таблица 4

Результаты эксперимента для размера блока, равного 20 страйпам

Алгоритм принятия решений	***B*, такты**	***A*, запросы**	***P***	***M*, страйпов за такт**	***L*, сек.**
Выкл.	–	–	368,09	367,68	0,24
Вкл.	10	10	377,59	377,27	0,31
Вкл.	10	20	377,16	376,83	0,31
Вкл.	20	10	377,71	377,40	0,32
Вкл.	20	20	377,58	377,26	0,31

По результатам моделирования можно сделать вывод о том, что общая производительность системы увеличивается в результате использования алгоритма принятия решений о пересылке запросов. Однако подобное улучшение сопряжено с ростом средней задержки обработки запросов, что может быть неприемлемо для ряда задач.

Заключение

В рамках данной работы была построена имитационная модель модульной СХД Active-Active, позволяющая исследовать производительность системы при различных конфигурациях элементов, оценить влияние резкого или постепенного изменения ее параметров в ходе численного эксперимента и эффективность алгоритма повышения производительности.

Литература

1. TN MPIO Policies // URL: http://technet.microsoft.com/en-us/library/dd851699.aspx
2. Cache Algorithms // URL: https://www.usenix.org/legacy/events/usenix01/full_papers/zhou/zhou_html/node3.html

МУЛЬТИАГЕНТНЫЕ ТЕХНОЛОГИИ ДЛЯ ПОСТРОЕНИЯ RAID-ПОДОБНЫХ РАСПРЕДЕЛЕННЫХ СИСТЕМ ХРАНЕНИЯ ДАННЫХ

К. И. Тюшев

студент кафедры системного программирования СПБГУ

E-mail: Kirill.Tyushev8@gmail.com

Научный руководитель:

О. Н. Граничин

E-mail: Oleg_granichin@mail.ru

Аннотация. В статье представлен способ для построения RAID-подобных распределенных систем хранения (СХД) данных с помощью мультиагентных технологий. Приведен пример моделирования СХД на Java Agent Development Framework (JADE) и показано преимущество использования мультиагентных технологий для подсчета контрольных сумм в СХД.

Введение

RAID (*redundant array of independent disks*) — это массив из нескольких устройств хранения данных, связанных между собой скоростными каналами передачи информации и воспринимаемых внешней системой как единое целое. Такой массив должен сохранять целостность данных при выходе из строя одного или нескольких устройств, а также обеспечивать высокую скорость чтения или записи данных. Каждое устройство хранения данных можно воспринимать как полностью независимый объект, находящийся сколь угодно далеко от других устройств. В этом контексте RAID-массив можно представить как мультиагентную систему, где в качестве одного агента выступает одно устройство хранения данных. Причем некоторые агенты либо вообще не могут общаться друг с другом, либо общаются по медленным и не надежным каналам передачи данных. В настоящей статье мы рассмотрим построение RAID-подобной распределенной системы хранения данных на произвольной мультиагентной системе.

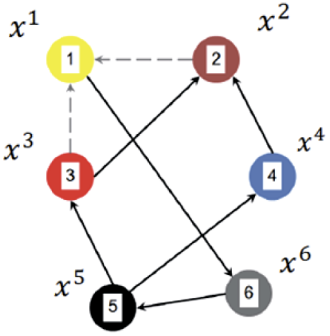


Рис. 1. Топология связей шести агентов с переменными ребрами 1-3 и 2-1

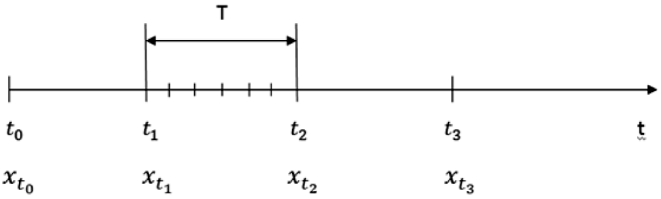
Используя алгоритм локального голосования в мультиагентных системах можно посчитать контрольные суммы на устройствах хранения данных с произвольной топологией

связей между устройствами, в условиях передачи данных с помехами и задержками [2–4]. На Рис. 1 показан возможный пример топологии связей с переменными ребрами 1-3 и 2-1.

Математические основания и алгоритм

Рассмотрим мультиагентную систему из n агентов. Пусть i, $i = 1, \ldots, n$ — номер агента, t — время, $N = \{1, ..., n\}$ — множество агентов. Будем вычислять и хранить m видов контрольных сумм от данных на агентах. В таком случае минимальное кол-во агентов, при котором сохраняется целостность данных, будет равно $(n - m)$.

Для простоты будем считать, что данные передаются без помех, без задержек, и каждый агент хранит только одно число. Расчет контрольных сумм проходит в заданные интервалы времени. В случае успешного завершения всех вычислений на текущем интервале, в дальнейшем возможно восстановление тех данных, которые были записаны до его начала. В начале интервала мы считаем целочисленную функцию f от числа на агенте и сохраняем полученное значение в заранее зарезервированной области памяти:



$$y_0^i = f(x_{t_k}^i),\ i \in N.$$

Далее по алгоритму локального голосования мы пересчитываем значения y_t^i:

$$y_t^i = y_{t-1}^i + u_t^i,\ \ i \in N,$$

$$u_t^i = \alpha \sum_{j \in N_t^i} b_t^{i,j} (y_t^{i,j} - y_t^{i,i}),\ \ i \in N.$$

Согласно [2]:

$$\lim_{t \to \infty} y_t^i = \frac{\Sigma_{j \in N} y_0^j}{|N|} = y^*,\ \ i \in N.$$

По определению предела получаем:

$$\varepsilon > 0,$$

$$\left| y_t^i - \frac{\Sigma_{j \in N} y_0^j}{|N|} \right| < \varepsilon,\quad i \in N,\ t > t',$$

$$\left| y_t^i |N| - \sum_{j \in N} y_0^j \right| < \varepsilon |N|, \quad i \in N, \, t > t',$$

$$\varepsilon |N| < 0{,}5.$$

Так как все y_0^j целые, то при округлении $y_t^i |N|$ к ближайшему целому получим точное значение $\sum_{j \in N} y_0^j$. При изменении функции f мы меняем вид контрольной суммы:

$$f(x_{t_k}^i) = \pm x_{t_k}^i, \quad i \in N,$$

$$m \le |N| \quad x_{t_k} = (x_{t_k}^1, x_{t_k}^2, ..., x_{t_k}^n);$$

$$h_1 = (\pm 1, \pm 1, ..., \pm 1,),$$

$$...$$

$$h_m = (\pm 1, \pm 1, ..., \pm 1,);$$

$$S_1 = <h_1, x_{t_k}>,$$

$$...$$

$$S_m = <h_m, x_{t_k}>;$$

Если $h_1, \ldots, h_m$ линейно независимые, то мы можем по контрольным суммам восстановить до m чисел на различных агентах.

После вычисления контрольной суммы на каждом агенте, только один агент сохраняет её у себя, а остальные удаляют, чтобы не занимать лишнюю память. Для простоты рассмотрим случай контрольной суммы только одного вида.

В общем случае каждый агент хранит массив чисел. Таким образом, все числа можно представить в виде матрицы, у которой столбцы это агенты, а строки это ряды чисел по которым мы считаем контрольные суммы. На диагонали этой матрицы мы можем изначально хранить нули, а в дальнейшем хранить там контрольные суммы. Тогда мы сможем на каждом агенте хранить как данные, так и контрольные суммы:

$$x_1^1, x_1^2, 0 \to x_1^1, x_1^2, S_1;$$

$$x_2^1, 0, x_2^3 \to x_2^1, S_2, x_2^3;$$

$$0, x_3^2, x_3^3 \to S_3, x_3^2, x_3^3.$$

Для восстановления потерянных данных достаточно пересчитать все контрольные суммы, положив неизвестные числа равные нулям и решить систему линейных уравнений относительно неизвестных чисел.

Моделирование

Работоспособность описанного алгоритмы была проверена с помощью имитационного моделирования. Параметр размера шага α в алгоритме локального голосования было выбрано равным 0.1. Числа на агентах выбирались случайно в диапазоне от 1 до 16 000. В качестве модели использовалась мультиагентная система, построенная с помощью Java Agent Development Framework (JADE). На Рис. 2 приведен типичный график зависимости времени подсчета контрольных сумм от кол-ва агентов при фиксированной топологии связей вида кольцо и когда данные передаются без помех и без задержек.

В случае подсчета контрольных сумм без использования алгоритма локального голосования нужно будет передавать данные от каждого к каждому. При произвольной топологии связей это потребует n^3 тактов, где n — кол-во агентов.

Из графика видно, что время подсчета контрольных при использовании мультиагентных технологий гораздо меньше чем n^3 и зависимость близка к линейной.

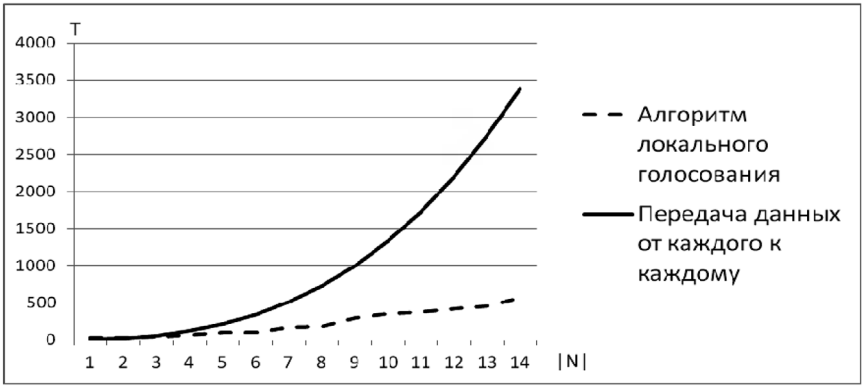


Рис. 2. График зависимости времени подсчета контрольных сумм от количества агентов

Дальнейшие планы

В дальнейшем планируется использовать результаты моделирования для распределенного хранения данных на беспилотных летательных аппаратах (БПЛА). Это позволит не терять данные при выходе из строя одного или нескольких БПЛА при выполнении групповых заданий.

Литература

1. *H. Peter Anvin* (2006–2011), The mathematics of RAID-6, http://kernel.org/pub/linux/kernel/people/hpa/raid6.pdf

2. *Амелина Н. О., Фрадков А. Л.* Приближенный консенсус в стохастической динамической сети с неполной информацией и задержками в измерениях // Автоматика и телемеханика. 2012. № 11. С. 6–29.
3. *Амелина Н. О.* Применение протокола локального голосования для децентрализованной балансировки загрузки сети с переменной топологией и помехами в измерениях // Вестник Санкт-Петербургского университета. Серия 1: Математика. Механика. Астрономия. 2013. № 3. С. 12–20.
4. *Amelina N., Granichin O., Kornivetc A.* Local voting protocol in decentralized load balancing problem with switched topology, noise, and delays // Proc. of the 52nd IEEE Conference on Decision and Control, December 10–13, 2013, Firenze, Italy. P. 4613–4618.

МОДЕЛИРОВАНИЕ ПОТОКА ЗАПРОСОВ НА ЧТЕНИЕ И ЗАПИСЬ ДАННЫХ

С. В. Морозов

СПбГУ, математико-механический факультет, кафедра системного программирования

E-mail: usd.icq@gmail.com

В. Н. Калитеевский

СПбГУ, математико-механический факультет, кафедра системного программирования

E-mail: vkalit@gmail.com

Аннотация. Работа посвящена моделированию нагрузки на устройство хранения. Моделируется поток запросов ввода-вывода, отвечающий заданным параметрам. Рассматриваются как поток одиночных транзакций, так и т.н. мета-транзакции, представляющие собой обращения к достаточно длинным непрерывным областям памяти. Возможна также комбинация двух типов потоков.

Введение

Вращающиеся диски — самый медленный компонент системы хранения данных. Доступ к данным на жестких дисках занимает как правило несколько миллисекунд, тогда как доступ к данным внутри кэш занимает менее миллисекунды.

Почему нельзя полностью заменить жесткие диски на твердотельные накопители? У организаций возникает потребность в хранении чрезвычайно больших объемов данных. Хранение данных на твердотельных накопителях значительно бы ускорило доступ к данным, по сравнению с хранением на жестких дисках, но стоимость оборудования на твердотельных накопителях значительно превышает стоимость хранения на магнитных дисках. В современных системах хранения используются гибридные накопители, укомплектованные магнитными жесткими дисками и флэш накопителями, а также используется высокоэффективная кэш-память, дающая возможность оптимизировать операции ввода-вывода. Такое решение проблемы позволяет значительно уменьшить стоимость за гигабайт ($/GB) и при этом обеспечивает вполне приемлемое время доступа к данным.

Моделирование рабочей нагрузки на систему хранения прежде всего нужно для оценки производительности системы. Смоделированная нагрузка должна быть обобщенной моделью реальной нагрузки. Тестирование оборудования на нагрузку дает заказчику уверенность в том, что система будет работать стабильно в реальных условиях, и упраздняет или уменьшает не-

обходимость испытания оборудования на реальном объекте, что приводит к существенной экономии ресурсов.

Компаниям, где надежность оборудования имеет высокий приоритет, полезно иметь инструмент, который позволяет тестировать оборудование на нагрузку и «на лету» менять интенсивность загрузки системы путем изменения широкого спектра параметров.

Ниже мы рассмотрим вопрос о построении тестового пакета, моделирующего нагрузку на кэш-память. Эта программа должна обеспечить простой способ генерирования нагрузки с возможностью гибкого изменения параметров, влияющих на интенсивность запросов на чтение и запись данных и разброс адресов.

Элементарные транзакции

Представление данных

Нагрузку системы будем представлять в виде последовательности элементарных транзакций (запросов). Запрос — это четверка <Время транзакции, Адрес в кэш-памяти, Размер транзакции, Тип транзакции> (см. Рисунок 1).

Предусмотрена возможность изменения интенсивности нагрузки во времени по специальному алгоритму, моделирующему типичное поведение приложений, нагружающих реальные системы. Компоненты запроса подчиняются определенным законам, к примеру, адреса распределены посредством моделирования кэш памяти с распределением стековых расстояний в соответствие с распределением Парето, а время между последовательными запросами в соответствие с экспоненциальным распределением.

Компоненты запроса:

- Время (Time): Время транзакции (точнее говоря, время между двумя последовательными транзакциями) распределено в соответствие с экспоненциальным распределением. Экспоненциальное распределение имеет единственный параметр $\lambda > 0$, называемый параметром скорости (rate parameter).
- Адрес (Address): распределен при помощи моделирования работы кэш-памяти (LRU алгоритм) с распределением стековых расстояний в соответствие с распределением Парето. Распределение Парето имеет 2 параметра: k и a. k — параметр положения (location parameter), определяет минимально возможное значение $x \geq k$, a — параметр формы (shape parameter), определяет «хвост» распределения.

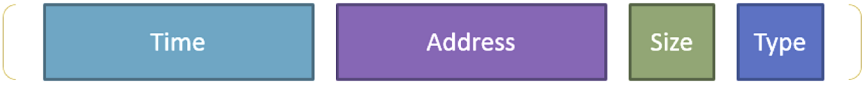


Рис. 1. Элементарная транзакция (request)

- Размер транзакции (Size): размер элементарной транзакции — это определенный размер считываемого или записываемого блока данных, к примеру 512 B, 1 KB, 2 KB, 8 KB, 64 KB и т.д. Задается вероятностями появления в последовательности запросов.
- Тип транзакции (Type): чтение или запись (Read/Write). Задается вероятностями появления в последовательности запросов.

Интенсивность нагрузки

Интенсивность нагрузки системы изменяется во времени. Рассмотрим алгоритм, моделирующий типичное поведение приложений, нагружающих систему.

Будем считать, что наша система может находиться в нескольких состояниях (states); ограничим количество возможных состояний шестью (states ≤ 6). В каждый момент времени система может находиться в определенном состоянии. Для каждого из состояний определены свои значения параметров распределений и вероятностей для каждого компонента элементарной транзакции.

	1	2	3	4	5	6
1		3	2	3	1	0,78
2	7		3	10	0,6	6
▶ 3	6	9		5	8,57	9
4	1	5,5	4		0,153	3,0938
5	0,8	0,018	0,74	3		7
6	4	2	6	11,9384	4,832	

Рис. 2. матрица состояний, задающая параметры λ_{ij} для перехода из состояния в состояние

Времена нахождения в состояниях, распределены в соответствие с экспоненциальным распределением. Переход из состояния в состояние задается с помощью матрицы состояний (см. Рисунок 2), в ячейках которой располагаются параметры λ_{ij}, задающие параметр скорости (rate parameter), используемого для расчета времени пребывания в состоянии *i* перед переходом в состояние *j*. Выбор следующего состояния осуществляется при помощи выбора минимального

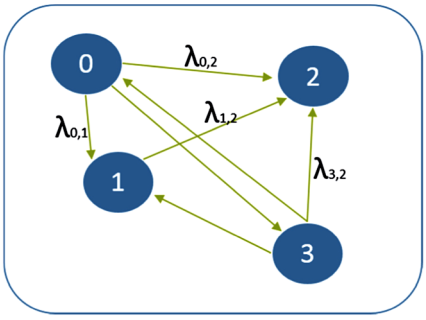


Рис. 3. Схема работы механизма, моделирующего изменяющуюся во времени нагрузку на систему (показаны не все связи между состояниями, states = 4)

промежутка времени, необходимого для нахождения в текущем состоянии перед переходом в следующее. Схема работы этого механизма представлена на Рисунке 3.

Групповые (мета) транзакции

Представление данных

Однако помимо одиночных запросов к устройству хранения может возникнуть необходимость обратиться для чтения или записи к большому, непрерывному участку памяти. Этот случай описывается групповыми транзакциями. И, так как подобная необходимость возникает часто, нужно уметь их моделировать.

Мета-запрос представляет собой последовательность элементарных запросов, описываемую следующими параметрами:

<Начальное время, Интенсивность, Начальный адрес, Размер мета-транзакции, Размер транзакции, Тип > (см. Рисунок 4).

Рис. 4. Мета-запрос (meta-request)

Компоненты мета-запроса:

- Время (Time): Начальное время мета-запроса распределено в соответствие с экспоненциальным распределением.
- Интенсивность (Frequency): Среднее количество элементарных транзакций в единицу времени. Задается через временные промежутки (Δt). Δt также распределено в соответствии с экспоненциальным распределением.
- Начальный адрес (Address): Адрес мета-запроса распределен равномерно по всей доступной области выделенной памяти.
- Размер мета-транзакции (Size): Распределен в соответствии с нормальным распределением. Нормальное распределение имеет два параметра: математическое ожидание и стандартное отклонение.
- Размер транзакции (Size): размер элементарной транзакции мета-запроса также зависит от характеристики системы хранения {512 B, 1 KB, 2 KB, 8 KB, 64 KB}. По умолчанию выбирается равновероятно.
- Тип мета-транзакции (Type): чтение или запись (Read/Write).

Интенсивность нагрузки

Количество генерируемых мета-запросов также зависит от текущего состояния системы и изменяется во времени. В случае сильной нагрузки на систему (пиковой нагрузки) мета-запросы могут пересекаться по времени. В этом случае происходит «наложение» одного мета-запроса на другой

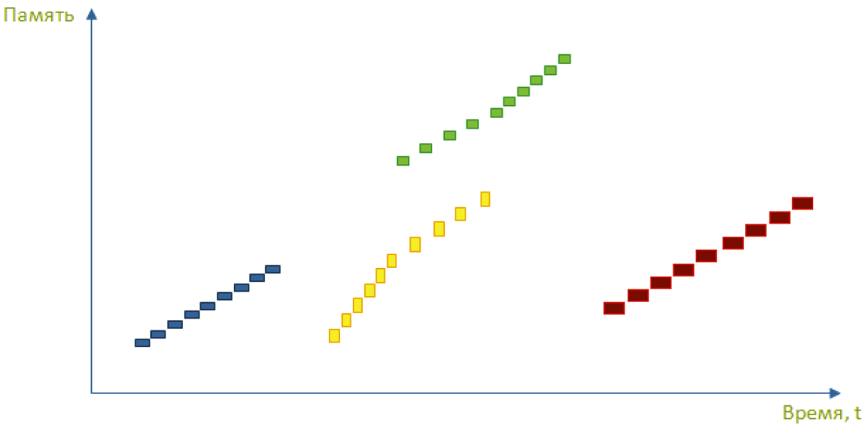


Рис. 5. Пересечение метазапросов

(см. Рисунок 5). Решается путем чередования элементарных транзакций каждого мета-запроса, причем очередная элементарная транзакция выбирается по признаку ближайшей по времени.

Объединение элементарных и мета транзакций

В реальных системах хранения данных происходят как одиночные запросы на чтение и запись, так и запросы на чтение и запись больших участков памяти. Поэтому имеет смысл объединить данные, полученные при моделировании элементарных транзакций, и мета-транзакций. Механизм объединения такой же, как и в случае, когда один мета-запрос «накладывается» на другой. Объединенную последовательность запросов мы получаем при помощи чередования запросов из результата моделирования нагрузки элементарными транзакциями и мета-транзакциями, причем очередной запрос выбирается по признаку ближайшего по времени.

Тестовый пакет для моделирования нагрузки на кэш-память

Программа для генерации нагрузки должна быть удобной в использовании и иметь возможность быстрой калибровки параметров под конкретное устройство. Была предпринята попытка создания такой программы.

Вышеописанный алгоритм моделирования нагрузки на систему хранения был реализован на языке C++ в виде динамически подключаемой библиотеки DLL.

Графический интерфейс написан на языке C# с использованием Windows Forms (см. Рисунок 6). Когда заданы все необходимые параметры вызывается вычислительная часть (динамически подключаемая библиотека на C++) и ге-

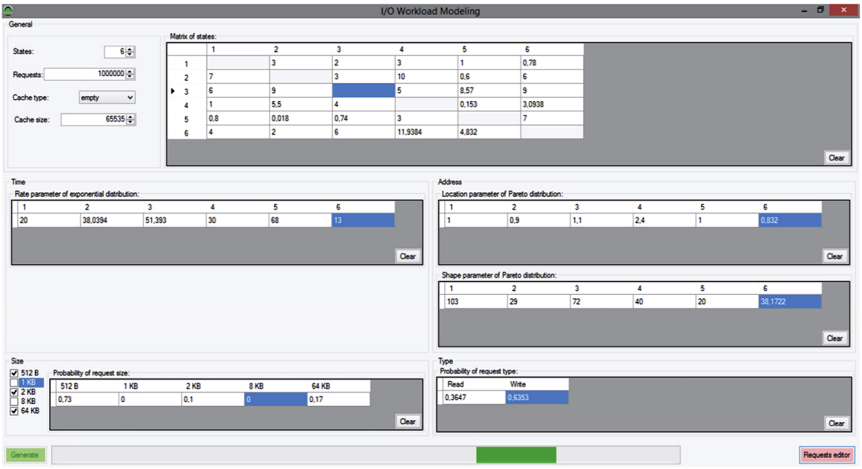


Рис. 6. GUI для генератора нагрузки на кэш-память

нерирует бинарный файл с последовательностью запросов. Имеется возможность открыть встроенный редактор запросов, чтобы посмотреть полученные данные, а также изменить значения отдельных компонент каждого запроса.

Заключение

В этой статье рассмотрена модель, в определенной степени приближающая результат моделирования нагрузки на систему хранения данных к реальной нагрузке, создаваемой приложениями. Для моделирования нагрузки, хорошо приближающей результат моделирования к реальности для конкретного устройства, необходимо подобрать подходящие параметры для компонентов запроса и для состояний системы.

Алгоритм может быть усовершенствован путем добавления дополнительных распределений к параметрам запроса. После анализа данных, полученных при моделировании нагрузки под конкретные условия, можно подобрать оптимальные параметры системы хранения, например выбрать размер кэш памяти, при котором система будет обрабатывать запросы с приемлемой скоростью.

Литература

1. *G. Feitelson*. Workload Modeling for Computer Systems Performance Evaluation.
2. *George Almasi, David A. Padua*. Calculating Stack Distances Efficiently.
3. *А. Н. Бородин*. Элементарный курс теории вероятностей и математической статистики.
4. Information Storage and Management. Second Edition, EMC Education Services.

Информационная безопасность

Молдовян
Александр Андреевич

д.т.н., профессор
зам. директора СПИИРАН по информационной безопасности

Фахрутдинов
Роман Шафкатович

к.т.н., заведующий лабораторией
безопасности информационных систем СПИИРАН

ПОДХОДЫ ДЛЯ РЕАЛИЗАЦИИ ПРОЦЕССА ПОДДЕРЖКИ ПРИНЯТИЯ РЕШЕНИЯ ДЛЯ РАЗВИТИЯ СОВРЕМЕННОЙ ОРГАНИЗАЦИИ НА ОСНОВАНИИ СТАТИСТИКИ СЕРТИФИКАЦИИ ISO

А. А. Молдовян, И. И. Лившиц, Танатарова А. Т.

Процесс проектирования, создания и внедрения современных систем менеджмента является важным и актуальным, при этом вопросы формального соответствия конкретным требования какого-либо выбранного международного стандарта (ISO) отходят на второй план, уступая место вопросам экономического порядка — какую выгоду для данной конкретной организации принесет проект и в какой перспективе? Какие риски могут быть на пути реализации проекта? Какие требования по безопасности приняты на рынках, на которые нацелены стратегические интересы?

Очевидно, что реализация проекта без серьезной проработки, точного расчета и ясного представления рисков, оценки необходимых ресурсов (бюджета, персонала, лицензий и пр.) невозможна для современной организации, работающей в более чем жестких конкурентных условиях. Для исследования выбраны основные стандарты ISO: серии 9001 (СМК), серии 27001 (СМИБ) и серии 14001 (СЭМ), т.к. достаточно полная и достоверная статистика по ним официально опубликована на портале ISO.

Для организации, которая планирует обеспечить достаточный экономический рост, могут быть предложены варианты внедрения определенного «набора» систем менеджмента, которые позволят, объективно, достигать поставленных высшим менеджментом целей — на уровне лучшей мировой практики. Важно, что эти варианты могут быть предложены заблаговременно до выполнения внешней оценки (аудитов), что позволит учесть не только особенности применения «целевых» стандартов, но и оценить «лучшую практику», имеющуюся в данной конкретной отрасли.

В предлагаемой работе предложены некоторые подходы для реализации процесса поддержки принятия решения в части выбора модели (системы менеджмента) для развития современной организации на фазе проектирования и оценки приемлемости выбора: по составу систем менеджмента, по применимым стандартам, по необходимости сертификации в функции обеспечения стабильного роста, безопасности бизнес-процессов, защиты ценных активов (в т.ч. нематериальных) на основании статистики сертификации ISO.

Определенные результаты оценок данного случайного процесса (при известных дискретных значениях времени и состояния) могут быть востребо-

ваны при решении задач реализации проектов систем менеджмента для защиты бизнес-активов как отдельно, так и в составе ИСМ. В частности, оценки СМИБ, представленные в статистических данных ISO, позволяют проследить прямую взаимосвязь с общим успехом по конкретной отрасли (в частности, IT), причем, численные значения отражают такую взаимосвязь для «лидеров» разных рангов.

ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ ИНВЕСТИЦИОННЫХ ПРОЕКТОВ В СФЕРЕ НЕДВИЖИМОСТИ

В. И. Емелин

ведущий научный сотрудник ОАО «НИИ «Вектор», д.т.н., с.н.с.

E-mail: emelin41@mail.ru

А. В. Григорова

преподаватель экономических дисциплин колледжа «Радиополитехникум» СПбГПУ

E-mail: gav.2508@mail.ru

Аннотация. Актуальность решения задач обеспечения информационной безопасности инвестиционных проектов в сфере недвижимости определяется, с одной стороны, усилением зависимости рыночных процессов от глобальных информационных технологий, что связано с существующей спецификой позиционирования недвижимого имущества, как товара, и, с другой стороны, уязвимостью этих технологий от методов и средств конкурентной борьбы. Наиболее рациональным решением этой задачи является создание технологий, направленных на выявление дезинформации, повышение качества информации.

Введение

В современных условиях ведения бизнеса инвесторы начинают активно использовать не три, а четыре основных инвестиционных ресурса: труд, капитал, технологии, а также релевантную информацию (постоянно обновляемые знания и различного рода сведения). Главный достигаемый эффект активного использования информационных технологий заключается в снижении транзакционных издержек при добывании разнообразной информации, а также в получении знаний, которые необходимы для снижения риска инвестиций и получения максимальной прибыли в сфере недвижимости.

Вместе с тем, сетевые технологии стали причиной усиления роли конкурентной разведки, поскольку именно появление Интернета позволило обеспечить многим пользователям дешевый доступ к огромным информационным ресурсам. Информационное оружие, как средство модификации (искажения) информации, обеспечивает достижение поставленной цели двумя возможными способами:

1) путем записи искаженной информации в базу данных после преодоления злоумышленниками систем защиты (несанкционированного доступа);

2) путем ввода в компьютерные системы в режиме санкционированного доступа информации, модифицированной в средствах наблюдения и измерения (искаженной в источниках информации).

В обоих случаях результатом информационного воздействия является наличие в сетевой компьютерной системе модифицированной информации. Отличительным признаком такой информации является нелинейный характер ее негативного влияния на инвестиции в сфере недвижимости.

Постановка задачи

Развитие глобальной информационной инфраструктуры происходит в соответствии с ключевыми принципами создания открытых сетей, что обуславливает необходимость учета влияния следующих их общих свойств на процессы добывания и обработки информации [2,4].

1. Объединение сетей разной архитектуры и топологии формирует сотовую структуру обмена информацией. Взаимообмен информацией, как по вертикали, так и по горизонтали приводит к возникновению эффекта, который можно определить как «недостаток информации при ее избытке». Так, например, в результате поиска ответа на запрос об определении понятия «информационная инфраструктура» Интернет предлагает 11 млн. документов, в которых содержится широкий набор различного, в том числе противоречивого описания этого термина.
2. Источниками информации становятся сети, информационный ресурс которых создается в соответствии с требованиями этих сетей и, прежде всего, в соответствии с их требованиями к полноте, достоверности и актуальности информации. Информационные ресурсы сетей официально подвергаются цензуре в зависимости от проводимой политики и собственных внутренних правил их формирования и распространения. Таким образом, если само понятие «информация» определять через ее свойства, то следует признать, что информационный ресурс сетей представлен глобально распределенной информацией, но свойства информации неизвестны.
3. Сервис World Wide Web (www), как гипертекстовая (гипермедиа) система, позволяет интегрировать различные сетевые ресурсы в единое информационное пространство, однако в сети отсутствует сервис по созданию единого центра управления всей совокупностью информационных ресурсов объединяемых сетей. Создание системы распределенного интеллекта и координации вычислений в сети является перспективным направлением развития глобальной информационной инфраструктуры.

Решение задачи

В этой связи весьма важным является введенное Юсуповым Р. М. [4] понятие информационной безопасности как следующей триады: защиты информации, защиты от информации и информированности (осведомленности) субъекта о намерениях и возможностях противоборствующей стороны в информационной сфере. В представленном определении информированность характеризует способность субъекта использовать имеющуюся у него информацию для формирования правильных суждений и вырабатывать на их основе наиболее обоснованные решения в процессе своей деятельности, которая протекает в условиях конкурентной борьбы. При разработке таких решений качество информации выступает в форме сложного свойства, которое проявляется в сопоставлении реальных измеряемых значений и предъявляемых к ним требованиям по полноте, достоверности и оперативности информации.

Решение задачи выявления дезинформации предлагается проводить для систем, которые в общем случае могут находиться в состоянии детерминированного хаоса. В соответствии с существующими определениями система считается детерминированной во времени, если существует правило в виде дифференциальных или разностных уравнений, определяющее ее будущее, исходя из заданных начальных условий. При исследовании таких процессов будем подразумевать, что система находится в состоянии нерегулярного или хаотического движения, порожденного нелинейными системами, для которых динамические законы однозначно определяют эволюцию во времени состояния системы при известной предыстории. Закономерности, наблюдаемые в поведении целого, есть результат хаотического движения составляющих его частей. Однако, если система находится в состоянии неустойчивого неравновесия, то действие случайных сил может привести к непредсказуемым результатам. Механизм потери информационной устойчивости начинает действовать при попадании параметров системы в область, которую Лоренц назвал странным аттрактором. Переход системы на такой режим означает, что в ней наблюдаются сложные непериодические колебания, весьма чувствительные даже к малому изменению начальных условий. В таких режимах невозможно определить конкретную траекторию изменения состояния и следует перейти к исследованию качественных свойств системы в целом — рассматривать семейство траекторий.

В рамках сформулированной общей проблемы обеспечения безопасности информации рассматриваются методы решения такой сложной задачи как проверка информации на правдоподобие. Под правдоподобием в рамках сформулированной проблемы понимается соответствие информации сформированной системе гипотез, под которыми будем понимать [1, 3] убеждение о существовании таких тенденций, которые действуют в течение прогнозируемого периода с установленной эффективностью. Решение этой задачи основано на описании отличительной особенности недвижимости от других

товаров, которая содержится в его определении: земельные участки, участки недр, здания и другие прочно связанные с землей объекты, перемещение которых без несоразмерного ущерба их назначению невозможно. Все иные характеристики правового статуса, ценности объекта и целый ряд других могут быть неразрывно связаны между собой, если будет сформирован и корректно определен объект недвижимости.

Заключение

Результатом проведенных исследований является установление взаимосвязи между пространством, описывающим возможные структуры состояний объектов, и внутренним пространством этой структуры, как сложного объекта. Информационная устойчивость систем обеспечивается формированием системы знаний, которая определяет задание таких начальных условий, при котором аттрактор расположен в области устойчивого неравновесия с высоким качеством информации. В этом случае созданный уровень информированности будет притягивать все траектории, построенные из значений ее текущих оценок, в область информации более высокого качества. Высокий уровень достоверности и валидности знаний позволяет выявлять поступление модифицированных данных.

Литература

1. *Григорова А. В*. Анализ влияния институциональных факторов на состояние рынка недвижимости мегаполиса (на примере Санкт-Петербурга) / Григорова А. В., Емелин В. И. // Научно-технические ведомости СПбГПУ. Экономические науки. № 2 (168), 2013.
2. *Емелин В. И*. Метод оценки выполнения требований информационной безопасности пользователями автоматизированных систем. Вопросы защиты информации / В. И. Емелин , А. А. Молдовян // Научно-практический журнал № 4 (79). 2007.
3. *Емелин В. И*. Метод оценки устойчивости автоматизированной системы в условиях информационного противоборства / В. И. Емелин // Научно-технические ведомости СПбГПУ. СПб.: Санкт-Петербургское изд-во политехнического университета. 2–1 (53). 2008.
4. *Юсупов Р. М*. Наука и национальная безопасность. 2-е изд. СПб.: Наука. 2011.

СХЕМА ХЭШ-ФУНКЦИИ С ПОТАЙНЫМ ХОДОМ

Д. М. Латышев
научный сотрудник СПИИРАН
E-mail: ldm@cobra.ru

Аннотация. Рассматривается понятие хэш-функции с потайным ходом, ее применение в задаче контроля целостности данных, приводится описание схемы хэш-функции с потайным ходом, стойкость которой основана на вычислительно сложной задачи факторизации.

Введение

Одним из удобных способов контроля целостности информации является использование хэш-функций. Для контролируемых данных с помощью хэш-функции вычисляется защитная контрольная сумма, которая затем используется как эталонная. При необходимости проверки целостности данных, контрольная сумма вычисляется повторно и затем сравнивается с эталонной контрольной суммой. При различии контрольных сумм, делается вывод о модифицировании данных. В большинстве случаев, требуется использовать стойкие хэш-функции, на которые налагаются некоторые требования. Одно из них — стойкость к коллизиям [1]. Это требование выражается в том, что вычислительно неосуществимо нахождение двух различных сообщений, значения хэш-функций которых равны. Однако для решения некоторых задач, полезно использовать хэш-функции, в спецификациях которых возможен обход данного требования, хэш-функции с потайным ходом.

Хэш-функция с потайным ходом

В хэш-функции с потайным ходом возможно осуществление такого модифицирования сообщения, при котором значения хэш-функций от исходного и измененного сообщений будут равны. Данная возможность предоставляется только тому, кто владеет ключом к потайному ходу, представляющим собой некоторые закрытые параметры хэш-функции. Данные параметры не используются в открытом виде, благодаря этому, для базового вычисления хэш-функции нет необходимости знать эти параметры. Для того, кто владеет ключом к потайному ходу, хэш-функция не является стойкой к коллизиям. Принцип формирования коллизии хэш-функции для определенного сообщения следующий. Производится выбор закрытых параметров, значение которых знает только владелец ключа. На основе закрытых параметров формируются открытые параметры, которые вписывают в спецификацию хэш-функции. Владелец ключа произвольно модифицирует исходное сообщение. Затем, используя закрытые параметры, формирует специальный

корректирующий блок данных, который вставляется в модифицированное им сообщение. Модифицируемое сообщение, содержащее корректирующий блок данных, будет иметь то же значение хэш-функции, что и исходное сообщение. Тот, кто не обладает ключом к потайному ходу, может вычислить значение хэш-функции, но не может сформировать коллизию.

Пусть имеется документ M, разбитый на блоки $\{M_1, M_2, \ldots, M_n\}$. Результаты вычисления значения хэш-функций после прохождения каждого блока обозначим как $\{H_1, H_2, \ldots, H_n\}$. Значение хэш-функции для i-го блока вычисляется по формуле:

$$H_i = \mathrm{h}(M_i, H_{i-1}),$$

где $1 \le i \le n$, h — раундовая хэш-функция. Необходимо сформировать измененный документ M′, разбитый на блоки $\{M'_1, M'_2, \ldots, M'_n\}$ и со значениями хэш-функции после прохождения каждого блока $\{H'_1, H'_2, \ldots, H'_n\}$, так, чтобы выходные значения хэш-функций были равны $(H_n = H'_n)$.

Начальное значение H_0 является специфицированным значением и не изменяется. Вставка корректирующего блока M'_i обеспечивает равенство $H_i = H'_i$. Корректирующий блок M'_i может быть вставлен в любое место модифицируемого сообщения, однако для того, чтобы итоговые значения хэш-функций были равны $(H_n = H'_n)$, необходимо чтобы все последующие блоки исходного и модифицируемого сообщений были одинаковы ($\{M_{i+1}, \ldots, M_{n-1}, M_n\} = \{M'_{i+1}, \ldots, M'_{n-1}, M'_n\}$). В связи с этим удобно использовать в качестве корректирующего блока последний блок. Тот, кто не обладает ключом к потайному ходу, может вычислить значения хэш-функции H_n и H'_n, но не может сформировать корректирующий блок.

Таким образом, хэш-функции с потайным ходом актуально использовать для решения задач, связанных с санкционированием изменений информации при контроле целостности данных, когда изменение значения хэш-функции является критичным. В общем случае, хэш-функция с потайным ходом может быть использована для решения следующих задач:

- Формирование двух документов с одинаковым значением хэш-функции;
- Формирование документа в дополнение к уже имеющемуся документу с тем же значением хэш-функции (при условии, что значение хэш-функции от первого документа было вычислено при помощи хэш-функции с потайным ходом);
- Подтверждение факта формирования пары сообщений определенным лицом (обладателем ключа к потайному ходу).

Стойкая хэш-функция с потайным ходом может быть построена на основе вычислительно сложной задачи факторизации. В этом случае, может использоваться следующая схема хэш-функции. Раундовое вычисление значения хэш-функции производится по следующей формуле:

$$H_i = \alpha^{H_{i-1} \cdot M_i^e \bmod n'} \bmod n,$$

где число имеет порядок, равный n' по модулю n; Mi < n'. Модули n и n' являются составными. Модуль n является произведением больших простых чисел p и q. Модуль n' является произведением больших простых чисел p' и q'. При этом, в разложении чисел $p-1$ и $q-1$ присутствуют числа p' и q'. Число α принадлежит показателю n' по модулю n. Число e должно выбираться таким, что НОД (e, $L(n')$) = 1, где НОД — наибольший общий делитель, $L(n')$ — обобщенная функция Эйлера, которая при $N > 1$ по определению равна

$$L(N) = \text{НОК}\,[p_1^{\alpha_1-1}(p_1-1);\ p_2^{\alpha_2-1}(p_2-1)\ldots;\ p_z^{\alpha_z-1}(p_z-1)],$$

где $N = p_1^{\alpha_1} p_2^{\alpha_2} \ldots p_z^{\alpha_z}$ и НОК [*] обозначает наименьшее общее кратное чисел, указанных в квадратных скобках. С учетом используемого значения $n' = p'q'$ имеем $L(n') = \text{НОК}\,[p'-1;\ q'-1]$.

Секретными параметрами являются числа p, q, p', q', d. Число d вычисляется при помощи расширенного алгоритма Евклида и должно удовлетворять $de \equiv 1 \bmod L(n')$. После выбора чисел p, q, p', q' вычисляются модули n и n', после этого числа p, q, p', q′ могут быть уничтожены. Секретные параметры являются ключом к потайному ходу, они генерируются будущим владельцем ключа и не разглашаются. Открытыми параметрами схемы являются числа n, n', e, α. Корректирующий блок данных находится по формуле:

$$M'_i = (H_{i-1} \cdot M_i^e \cdot (H'_{i-1})^{-1})^d \bmod n'.$$

Владелец ключа к потайному ходу может сформировать коллизию для произвольного документа. Вычислительная сложность разложения составного модуля n' гарантирует стойкость хэш-функции для не обладателей ключа. Число n' является произведением больших простых чисел. Исходя из этого, вероятностью того, что $\text{НОД}\,(H'_{i-1}, n') \neq 1$ можно пренебречь. Для того чтобы обеспечить выполнение $\text{НОД}\,(H'_{i-1}, n') = 1$, можно использовать два корректирующих блока данных M'_{i-1} и M'_i. Первый блок M'_{i-1} выбирается такой, чтобы обеспечить равенство $\text{НОД}\,(H'_{i-1}, n') = 1$ для второго блока. Второй корректирующий блок M'_i вычисляется по указанной ранее формуле.

В представленной схеме хэш-функции с потайным ходом, длина блоков данных меньше длины значения хэш-функции. Для того чтобы длины отличались несущественно, множители p и q можно выбирать такими, что $p = 2p' + 1$ и $q = 2q' + 1$. p' и q' должны быть простыми числами, такими что в разложениях $p'-1$ и $q'-1$ содержались простые большие множители.

Заключение

Рассмотренная схема хэш-функции с потайным ходом, основанная на вычислительно сложной задачи факторизации, позволяет осуществлять дозво-

ленное изменение информации без влияния на результаты работы систем контроля целостности.

Литература

1. *Молдовян Н. А., Молдовян А. А*. Введение в криптосистемы с открытым ключом. Спб.: БХВ-Петербург, 2005.

МЕТОДИКА ОЦЕНКИ ЗАЩИЩЕННОСТИ ИНФОРМАЦИОННЫХ АКТИВОВ

С. А. Рудакова
аспирант СПИИРАН
E-mail: kuznechik.88@mail.ru

Аннотация. В статье приведено краткое описание методики оценки защищенности информации, обрабатывающейся на объекте информатизации, в основе которой лежит концепция выбора метрик информационной безопасности.

Введение

Оценка защищенности информационных активов — одна из наиболее пользующихся спросом услуг на рынке информационной безопасности (ИБ). Российским компаниям такая оценка может потребоваться для:

- выполнения требований регуляторов ИБ (ФСТЭК России, ФСБ России, Роскомнадзор, отраслевые регуляторы);
- внутренних целей, таких как составление стратегии развития ИБ в компании, обоснование финансовых расходов на ИБ и др.;
- повышения деловой репутации, если по итогам такой оценки будет декларировано соответствие организации требованиям по ИБ.

Существуют разные подходы к способам оценки защищенности информационных активов. Например, [1] разделяет их на качественные, включающие:

- оценку степени выполнения требований стандарта по ИБ;
- автоматизированные тесты по выявлению уязвимых мест информационной системы,

и количественные, включающие:

- инструментальные средства оценки («АванГард», «ГРИФ» и т. д.);
- дискретные оценки эффективности средств защиты.

Существуют и другие подходы (некоторые из них описаны, например, в [2, 3]).

Таким образом, сегодня нет единого общепринятого подхода к способу оценки защищенности информационных активов, а существующие подходы имеют каждый свои недостатки, такие как:

- игнорирование индивидуальной специфики объекта информатизации;
- узкая направленность;
- высокая степень зависимости от квалификации специалистов, проводящих оценку.

Новая методика оценки защищенности информационных активов, минимизирующая эти недостатки, позволила бы с большей уверенностью гарантировать результаты оценки (т.е. гарантировать, что все необходимые требования учтены и оценены корректно). Поэтому разработка такой методики представляется актуальной задачей.

В статье приведено краткое описание такой методики.

Выбор метрик ИБ

В основе оценки ИБ объекта информатизации лежит оценка некоторых критериев — метрик ИБ.

Как правило, используются либо наборы метрик, предоставленные стандартами по ИБ, либо составленные экспертами. Такие наборы могут быть неполными, или обладать избыточностями.

Предлагается концепция выбора метрик ИБ, основанная на пошаговом разделении свойства «информационная безопасность» [4].

Свойство «информационная безопасность», (свойство верхнего уровня) разделяется на несколько более детализированных свойств (свойств нижнего уровня), с соблюдением правил, описанных ниже.

Каждое свойство нижнего уровня подвергается детализации по тем же правилам (при этом детализируемое свойство становится свойством верхнего уровня) до тех пор, пока оно не будет отвечать требованиям, предъявляемым к метрикам ИБ:

- конкретность (метрика должна иметь непосредственное отношение к ИБ);
- измеримость (должна существовать возможность измерить метрику с помощью булевой алгебры);
- значимость (изменение значения метрики должно означать изменение состояния ИБ объекта информатизации).

Правило необходимости свойств

Должно соблюдаться условие: $L_i \subsetneq H$ при $i = 1, 2, ..., n$, где L_i — множество характеристик свойства нижнего уровня; H — множество характеристик свойства верхнего уровня; n — количество свойств нижнего уровня.

Правило достаточности свойств

Должно соблюдаться условие: $\forall A \cap H = \varnothing$ при $A \cap L_i = \varnothing$; $i = 1, 2, ..., n$, где A — произвольное множество; H — множество характеристик свойства верхнего уровня; L_i — множество характеристик свойства нижнего уровня; n — количество свойств нижнего уровня.

Правило уникальности свойств

Должны соблюдаться условия:

- $L_i \cap L_j = \varnothing$ при $i = 1, 2, ..., n; j = 1, 2, ..., n; i \neq j$;
- $L_i \neq \varnothing$ при $i = 1, 2, ..., n$,

где L_i, L_j — множества характеристик свойства нижнего уровня; n — количество свойств нижнего уровня.

Правило количества свойств

Количество свойств нижнего уровня должно быть минимальным, но не менее двух (предлагается использовать 2–5 свойств).

Оценка метрик ИБ

Каждая метрика ИБ, найденная с помощью концепции выбора метрик ИБ, описанной выше, согласно этой концепции, измерима.

Специалист, осуществляющий оценку (аудитор) должен для каждой метрики ИБ:

- определить ее возможные значения;
- определить ее текущее значение.

Предполагается, что метрика ИБ может быть однозначно измерена, поэтому зависимость результатов измерения от квалификации аудитора на этом шаге не существенна.

Оценка защищенности информационных активов

Для оценки защищенности информационных активов необходимо обработать результаты оценки метрик ИБ таким образом, чтобы можно было говорить о защищенности в целом.

На Рис. 1 схематично представлен процесс выбора метрик ИБ в соответствии с описанной выше концепцией.

Листья графа, представленного на Рис. 1 — это метрики ИБ, корень — свойство «информационная безопасность», которое и требуется оценить.

Для оценки предлагается выполнить следующие действия:

1. Выбрать лист, длина пути от которого до корня максимальна. Назовем его вершиной нижнего уровня. Соседнюю вершину назовем вершиной верхнего уровня (обозначим h).
   Все вершины, получившиеся в результате детализации вершины верх-

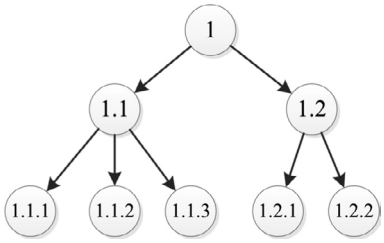


Рис. 1. Концепция выбора метрик ИБ

него уровня при применении концепции выбора метрик ИБ, обозначим как l_i, $i = 1, 2, ..., m$, где m — количество таких вершин.

2. Рассчитать: $h = \frac{\sum_{i=1}^{m} l_i}{m}$.
3. Листы l_i, $i = 1, 2, ..., m$ из графа вычеркнуть.
4. Повторять действия 1–3 до исчезновения в графе листьев.

Оценка корня графа будет являться оценкой свойства «информационная безопасность».

Заключение

В статье предложена методика, позволяющая количественно оценить защищенность информационных активов.

Поскольку методика не опирается ни на стандарты ИБ, ни на наборы требований и метрик ИБ, она применима для широкого круга информационных систем.

Литература

1. *Ерохин С. С.* Методика аудита информационной безопасности объектов электронной коммерции. Диссертация кандидата технических наук. Томский государственный университет систем управления и радиоэлектроники, Томск, 2010.
2. NIST SP 800–115. Technical Guide to Information Security Testing and Assessment.
3. *Просянников Р.* Виды аудита информационной безопасности // http://www.connect.ru/article.asp?id=5293 [дата просмотра: 15.04.2014].
4. *Рудакова С. А.* Концепция выбора метрик информационной безопасности // Вестник государственного университета морского и речного флота имени адмирала С. О. Макарова. 2013. Вып. 3 (22). С. 162–166.

УПРАВЛЕНИЕ ПОЛИТИКАМИ КОНТРОЛЯ ДОСТУПА НА ОСНОВЕ RBAC МОДЕЛИ

Р. С. Одеров

студент кафедры системного программирования СПбГУ,

E-mail: roman.oderov@gmail.com

Введение

Безопасность играет одну из важнейших ролей в современных информационных системах. Обычно под информационной безопасностью (или просто безопасностью) понимают защищенность информации и соответствующего окружения от случайных или умышленных действий, в результате которых может быть нанесен неприемлемый ущерб как системе, так и ее пользователям [14].

Предметная область

Во многих организациях осуществляется работа с различного рода секретной информацией. Системы, работающие с секретными файлами, обязаны удовлетворять строго определенным требованиям, описанным в специальных стандартах, законах и т. п.

Очевидно, чем больше ущерб от возможной «утечки» секретных данных, тем серьезнее принимаемые меры по обеспечению их защиты. Поэтому в нормативных документах, упомянутых выше, требования к безопасности варьируются в зависимости от уровня конфиденциальности информации [5, 14].

Описание политик разграничения (контроля) доступа [ПКД] является неотъемлемой частью любой серьезной системы безопасности. ПКД регламентирует права доступа к данным в рамках информационной системы, определяя разрешенные действия для пользовательских процессов. В основе политик лежат модели контроля доступа [МКД]. МКД — математически формализованная система, в общем случае оперирующая такими элементами, как объект (файл и т.п.) и субъект (процесс, пользователь…). В терминах модели определяются правила доступа, по которым субъектам разрешается или запрещается выполнять определенные операции над объектами. Существует две основных МКД: дискреционная (DAC, Discretionary access control) и мандатная (MAC, Mandatory Access Control) [1, 2, 7, 13, 14].

Система, предоставляющая пользователям возможность одновременно работать с информацией различных уровней секретности, должна реализовывать и различные механизмы для ее защиты. В частности, не обойтись без поддержки ПКД для разных уровней конфиденциальности (согласно законодательству и стандартам в области безопасности).

В крупных организациях, серьезно заботящихся о сохранности своих данных, (оборонные предприятия, службы безопасности и т.п.) сотрудники вынуждены использовать физически разные компьютеры при одновременной работе с файлами различных уровней секретности. Очевидно, этот подход очень неудобен.

На кафедре Системного Программирования Санкт-Петербургского Государственного Университета стартовал проект Multi-Cloud Desktop, направленный на решение поставленной выше проблемы. Цель проекта — позволить пользователю работать с документами разных уровней конфиденциальности на одной физической машине. В основе проекта лежат технология виртуализация и современные подходы, связанные с удаленной доставкой приложений. Для хранения секретной информации и для работы пользовательских приложений организуются отдельные безопасные «облака» (в зависимости от степеней секретности). Таким образом, в рамках системы Multi-Cloud Desktop необходимо реализовать несколько политик контроля доступа, в основе которых лежат концептуально разные модели.

При реализации такой системы безопасности придется столкнуться с задачей выражения основных МКД через некоторую «базисную» модель. Также не стоит забывать о том, что при подобном моделировании существенно возрастет сложность управления рассматриваемой системой безопасности (визуализация, конфигурирование…).

В работе исследованы основные МКД; показан способ построения системы контроля доступа [СКД], средствами которой реализуемы различные модели; описано построение прототипа СКД, основанного на RBAC модели [3, 10, 11, 12], и инструмента управления.

Задача управления политиками доступа в системе Multi-Cloud Desktop

В качестве единой «базисной» была выбрана ролевая модель, средствами которой можно выразить MAC и DAC [4, 6, 8, 9]. Нужно упомянуть, что подобное моделирование приводит к взрывному росту сложности и размера RBAC. Поэтому требуется уделить должное внимание проектированию архитектуры инструмента управления политиками доступа и продумать схему хранения элементов ролевой модели.

Также одной из важнейших задач является реализация удобных и понятных пользователю инструментов администрирования, которые могут быть основаны на современном подходе визуального программирования. Пользователь (даже не профессиональный программист) должен иметь возможность легко создать требуемую политику с помощью таких визуальных средств, как диаграммы, схемы и пр.

Для создания прототипа системы управления политиками контроля доступа были выделены следующие задачи:

- Проектирование архитектуры системы управления политиками.
- Реализация структуры хранения RBAC.
- Реализация алгоритмов работы с RBAC.
- Пакетная загрузка данных в модель (на основе XML).
- Визуализация политик.
- Визуальное программирование политик.

На данный момент реализовано:

- Структура хранения базовой ролевой модели (RBAC0).
- Пакетное внесение данных с использованием XML.
- Управление RBAC0 [11].
- Визуализация модели.

Будущие направления развития:

- Реализация моделей RBAC1 и RBAC2 [11].
- Дополнительные средства визуализации.
- Визуальное моделирование.
- Шаблоны для моделирования MAC и DAC.
- Отслеживание действий пользователя при создании политик. Предотвращение появления потенциально небезопасных состояний системы.
- Интеграция в MCD.

Литература

1. Access Control Fundamentals, part 3 [В Интернете] / авт. *Sadeghi Cubaleska*. 2009. — http://www.trust.rub.de/media/ei/lehrmaterialien/232/OSS_chap3.pdf.
2. Access Control Models [В Интернете] / авт. *Kantarcioglu Murat*. — UT Dallas, 2009. — http://www.utdallas.edu/~muratk/courses/dbsec09s_files/access2.pdf
3. Access rights administration in Role-Based Security Systems [Журнал] / авт. *Matunda Nyanchama Sylvia Osborn*. — [б.м.] : The dep. of CS, The University of Western Ontario, 1994.
4. Configuring Role-Based Access Control to Enforce Mandatory and Discretionary Access Control Policies [Журнал] / авт. *Sylvia Osborn Ravi Sandhu, Qamar Munawer*. 2000.
5. DoD Standard 5200.28-STD «The orange book». — [б.м.] : United States Department of Defense, 1985.
6. How to do Discretionary Access Control Using Roles [Журнал] / авт. *Ravi Sandhu Qamar Munawer*. –1998.
7. Integrity considerations for secure computer systems [Доклад]. — Hanscom Air Force Base, Bedford, Massachusetts : Deputy for command and management systems, Electronic Systems division, Air Force Systems Command, United States Air Force, 1977.
8. RBAC on MLS Systems without Kernel Changes [Журнал] / авт. *Kuhn D. Richard*. 1998.
9. Role Hierarchies and Constraints for Lattice-Based Access Controls [Журнал] / авт. *Sandhu Ravi*. — [б.м.] : George Mason University & SETA Corporation, 1996.

10. Role-Based Access Control (RBAC): Features and Motivations / авт. *David F. Ferraiolo Janet A. Cugini, D. Richard Kuhn*. — [б.м.] : National Institute of Standards and Technology, 1995.
11. Role-Based Access Control Models [Журнал] / авт. *Ravi S. Sandhu Edward J. Coyne, Hal L. Feinstein, Charles E. Youman*. 1995.
12. Role-Based Access Controls [Журнал] / авт. *David F. Ferraiolo D. Richard Kuhn*. — [б.м.] : National Institute of Standards and Technology, 1992 г..
13. Secure Computer Systems: Mathematical Foundations [Доклад] / авт. *D. Elliott Bell Leonard J. LaPadula*. — [б.м.] : MITRE, 1973.
14. Основы информационной безопасности [В Интернете] / авт. *Галатенко В.* // ИНТУИТ. 01.04.2003. — http://www.intuit.ru/studies/courses/10/10/info

МЕТОДЫ УСИЛЕНИЯ ЗАЩИТЫ ПРИ ИСПОЛЬЗОВАНИИ СУЩЕСТВУЮЩЕЙ ИНФРАСТРУКТУРЫ СВЯЗИ

Р. Ш. Фахрутдинов

заведующий лабораторией безопасности информационных систем СПИИРАН, к.т.н.

E-mail: fahr@cobra.ru

Аннотация. В настоящее время сложилась ситуация, при которой существующая развитая инфраструктура связи (ИС) обеспечивает большие возможности для использования в различных областях человеческой деятельности, однако вопросы безопасности информации всё ещё остаются не решёнными в достаточной степени. Рассматриваются различные аспекты повышения защищённости информации, которая передаётся с использованием существующей ИС.

Введение

Как показывает практика, существующая ИС имеет ряд проблем с безопасностью, которые успешно эксплуатируются [1] с целью получения доступа к данным пользователей (независимо от характера использования — промышленная группа, офис, домохозяйка — все данные представляют интерес).

Кроме эксплуатации существующих уязвимостей, специальные службы некоторых стран осуществили (судя по всему, с ведома производителей) внедрение ряда дополнительных «возможностей» в существующее телекоммуникационное оборудование с целью ещё более полного доступа к передаваемой и хранимой информации [2].

Всё это ставит под сомнение желание владельцев ИС в полной мере обеспечить безопасность передаваемой информации. С одной стороны, этому мешает монополия на стандарты в области телекоммуникационного оборудования (с алгоритмами шифрования, вскрываемыми в реальном масштабе времени [3], как это обстоит с алгоритмом A5/1 стандарта GSM). С другой стороны возможности владельцев ИС по использованию полученной ими информации в своих целях (собственная аналитика, контекстная реклама, передача этих данных третьим лицам, сотрудничество со спецслужбами и т. д.).

Особо отметим, что большинство «шпионского оборудования» [2] собирает информацию с инфраструктурных объектов — маршрутизаторов, роутеров, серверов и т. д. Конечно, есть и «клавиатурные» кейлоггеры, подслушивающие устройства, но именно магистральным каналам связи отдают предпочтение заказчики «спецтехники».

Инфраструктура связи и абонентское оборудование

Кратко остановимся на понятии ИС. В соответствии с [4], *«инфраструктура связи» может быть определена как система взаимосвязанных объектов, сооружений, предприятий связи, видов деятельности, персонала, образующих организационно-техническое единство комплекса, обеспечивающего прием, хранение, передачу, доставку информации, сообщений от отправителя до адресата, соответствующего международным стандартам.*

Инфраструктуру связи составляет материально-техническая база: объекты, сети, сооружения, средства связи, работники, способы организации деятельности, работ — все это в своей целостности и создает возможность приема, хранения, передачи, доставки информации в виде сообщений от отправителя до адресата...»

Понятие ИС неразрывно связано с понятием информационной инфраструктуры. В контексте данной статьи, предлагается немного упростить и обобщить понятие ИС до «совокупность программных, программно-аппаратных, аппаратных и других технических средств и пользовательских интерфейсов, позволяющих пользователю осуществлять передачу и приём информации». В рамках данной статьи, ИС являются социальные сети, мобильная связь, IP-службы, Интернет и т. д., в общем всё то, с помощью чего конечные пользователи могут отправлять и принимать информацию.

Абонентское оборудование (АО) даёт возможности подключения пользователя к ИС и имеет набор программно-аппаратных интерфейсов для взаимодействия с человеком (экран, звуковоспроизводящее устройство, клавиатура, манипулятор «мышь» и т. д.). В качестве примера АО можно привести ПК с сетевой картой/модемом, мобильный телефон, VoIP-терминал, факс и т. д. Угрозы и условные нарушители

Основными угрозами являются :

1. перехват;
2. блокирование;
3. искажение;
4. нарушение авторских прав;
5. подмена;
6. незаконное использование.

Условные нарушители — лица и организации, получающие неправомерный доступ к информации без явного согласия её владельцев. «Условность» нарушителя заключается в том, что доступ к информации имеют владельцы ИС, договоры которых с пользователями имеют зачастую слишком широкий характер (разрешая владельцам ИС доступ к пользовательской информации, её репликацию, использование, включая передачу третьим лицам и т. д.) В условиях монополизации рынка ИС, низкой правовой культуры конечных пользователей, слабости правовых механизмов по защите личной информации (как в плане непосредственно законотворчества в этой области,

так и в рамках исполнения существующего законодательства), пользователи не глядя подписывают «типовой» договор, не имеют возможности по отслеживанию реального использования своей информации и т. д. К условным нарушителям можно отнести:

- хакеров;
- ИТР (иностранные технические разведки);
- спецслужбы;
- владельцов сервисов, провайдеров;
- любопытных.

Возможности по усилению защиты

Ранее мы уже отмечали, что усиление защиты инфраструктуры затруднено различными обстоятельствами, среди которых основным является желание владельцев ИС иметь доступ к передаваемой информации.

Основными препятствиями являются :

1. дороговизна;
2. отсутствие необходимости в защите информации всех пользователей (по мнению владельцев ИС);
3. незаинтересованность владельцев;
4. отсутствие возможностей по дополнительному усилению АО (вследствие монополизма в области стандартов и разработки конечных АО).

По совокупности этих причин, усиление защиты в существенной степени зависит от возможности дополнительно защитить АО :

- на уровне канала данных;
- на уровне протоколов и форматов данных (в т.ч. селективным шифрованием) и дополнительных средств ОС/СЗИ;
- на уровне защищённых виртуальных устройств (виртуальный микрофон, виртуальная видеокамера);
- на уровне приложений (расширения программ и т. д.).

Рассмотрим эти возможности подробнее. Защита на уровне канала данных возможна в случае, если ИС предоставляет канал связи для конечного пользователя (например, в случае подключения к сети Интернет). Тогда пользователь имеет возможность организовать защиту канала связи с помощью шифрования с использованием ssl/https, ssh, VPN, криптомаршрутизатора и т. д. Однако защита канала должна быть реализована и на приёмной стороне, что не всегда возможно.

На уровне протоколов, форматов данных и на уровне дополнительных средств ОС/СЗИ, пользователь может включить штатные средства защиты ОС/СЗИ, настроить почтовую систему на приём и отправку шифрованных сообщений, подключить функции шифрования к используемым НЖМД,

сменным носителям, включить шифрование в СУБД, организовать дополнительные средства защиты сетевого трафика (firewall).

При использовании ИС в виде конечных приложений (Skype, VoIP, ICQ, сотовая связь), можно использовать специальные защищённые виртуальные устройства, которые будучи подключенными к реальным устройствам, выполняют некоторое преобразование (условно — шифрование) по секретному ключу, над исходными данными и выдают эти данные в приложение. Так, например, к реальному микрофону подключен модуль преобразования голосового трафика, который выполняет скремблирование по секретному ключу, модуль имитирует ещё один микрофон, подключенный к ПК. К этому микрофону подключается Skype и дополнительно защищённая речь передаётся по протоколу Skype к принимающей стороне. Там, после декодирования с помощью Skype, голосовой трафик поступает на виртуальный динамик, который выполняет обратное преобразование по секретному ключу. Конечно, преобразованное звуковое сообщение хуже сожмётся в речевом кодеке Skype, однако у большинства такого рода ПО (подключаемого по широкополосным сетям доступа) имеет существенный запас (так как подразумевается, что кодек будет сжимать не только речь, но и музыку и разговор нескольких человек, а это по плотности звуковой информации уже близко к скремблированному речевому сообщению). Аналогичным образом возможно защитить и видеоинформацию [5].

На уровне приложений (в основном это социальные сети), которые функционируют в среде интернет-браузеров, возможно использование системы подключаемых модулей (плагинов), которые дают возможность выполнять преобразование вводимого текста, размещаемых изображений, видеофайлов (в том числе с помощью функций селективного шифрования) [5], аудиоданных по секретному ключу.

Отдельным вопросом для пользователя является место хранения ключевой информации. Сам по себе вопрос является достаточно ёмким, самая простая рекомендация — не хранить его на самом АО (ПК, мобильном телефоне и т. д.), лучше использовать отдельное внешнее устройство (можно в виде смарт-карты с интерфейсом USB). Наличие функции секретного хранилища и криптографические возможности, позволяют, при правильном использовании и качественном ПО получить высокую степень безопасности хранения ключевой информации в этом случае.

Заключение

В документе рассмотрены некоторые аспекты усиления защиты информации при использовании существующей ИС.

Литература

1. NSA. PRISM/US-984XN Overview/ NSA // http://s3.documentcloud.org/documents/813847/prism.pdf — 2007.01.08
2. NSA. The NSA»s Spy Catalog (derived from NSA/CSSM 1 52)/ NSA // https://www.aclu.org/files/natsec/nsa/20140130/NSA»s Spy Catalogue.pdf — 2007.01.08
3. *Alex Biryukov, Adi Shamir, David Wagner*. Real Time Cryptanalysis of A5/1 on a PC / Alex Biryukov, Adi Shamir, David Wagner // Fast Software Encryption Workshop 2000. April 10–12, 2000.
4. *Мхитарян Ю. И*. Инфраструктура связи — проблемы соответствия требованиям информационной экономики // Век качества. 2010. № 4. С. 10.
5. *Фахрутдинов Р. Ш*. Метод защиты видеоданных с различной степенью конфиденциальности. Автореф. дис. канд. тех. наук. Санкт-Петербург. 2012. С. 55–57.

НОВОСТНАЯ АВТОРИЗАЦИЯ

М. В. Баклановский
ст. преп. кафедры системного программирования СПбГУ
E-mail: m.baklanovsky@spbu.ru

О. Н. Граничин
проф. кафедры системного программирования СПбГУ
E-mail: oleg_granichin@mail.ru

А. Р. Ханов
аспирант кафедры системного программирования СПбГУ
E-mail: awengar@gmail.com

Аннотация. Новостная авторизация — это новая концепция в области систем контроля доступа. Она основана на выставлении авторизуемому субъекту оценки доверия на основе того, насколько точно он способен описать последовательности событий системы (историю). В качестве математического аппарата, позволяющего реализовать новостную авторизацию, нами предложены алгоритмы рандомизированного восстановления разреженных сигналов.

Введение

Процессы аутентификации и авторизации — это основа систем контроля доступа. Аутентификация — это процесс установления соответствия субъекта легитимному пользователю на основе характеристики — ключа. Авторизация — это предоставление данному аутентифицированному пользователю прав на выполнение в системе определенных действий.

Однако разработка механизмов аутентификации и авторизации связана с техническими проблемами: проблема атаки MITM, генерация и обновление ключа, выбор длины ключа, низкая скорость вычисления криптографических функций, проверка корректности выставления прав на объекты.

В работе предлагается новая концепция, которая предлагает по-новому взглянуть на процессы аутентификации и авторизации.

Динамическая авторизация

В традиционной модели аутентификации субъект предоставляет ключ, в результате он либо проходит эту процедуру, либо получает отказ. В случае успешного захода пользователь авторизуется в системе на выполнение определенных действий.

Динамическая авторизация использует ключ для того, чтобы вычислить коэффициент доверия субъекту. На основе этого коэффициента те или иные операции одобряются или отвергаются. Более того, пользователь не авторизуется в системе единовременно. Его авторизация продолжается непрерывно в процессе его взаимодействия с системой. Каждое действие субъекта корректирует коэффициент доверия.

Новостная авторизация

Новостная авторизация — это частный случай динамической авторизации.

Новость — это событие в системе. Последовательность новостей с временными метками — история новостей. Ключ — это идентификатор, который зависит от истории новостей. Аутентификация — это установление соответствия между ключом и текущей историей новостей. Коэффициент доверия — это оценка соответствия предоставленного ключа текущей истории новостей.

При таком способе аутентификации возникает ряд сложностей. Во-первых, что считать новостью. Событиями в системе могут быть как осмысленные данные, так и псевдослучайные числа. Во-вторых, как хранить историю новостей. Необходимо иметь модель, которая содержит в себе информацию о давних событиях и позволяет быстро вносить данные о вновь произошедших. По модели нужно генерировать ключ — слепок истории. Ключ содержит информацию об истории, которая может быть проверена системой.

Оказывается, что в теории рандомизированного восстановления разреженных сигналов уже есть наработки, позволяющие реализовать новостную авторизацию[1].

Пусть $I=(p_1, \ldots, p_N)$ — последовательность чисел. Она является s-редкой, если не более s её компонент являются ненулевыми. Возьмем m векторов A из R^N с компонентами, выбранными произвольно из множества {0, 1}. Вычисляем m скалярных произведений $b_i = <A_i, I>$. Теперь нужно, зная все A_i и b_i, найти s-редкую последовательность I, причем $m < N$. В общем случае это невозможно, но s-редкий I может быть восстановлен. Для этого необходимо решить задачу оптимизации:

$$\min L_0(I), <A_i, I> = b_i, i = 1 \ldots m.$$

Задача однозначно разрешима при $m >= 2s$ [2]. Однако она решается только полным перебором. Но оказывается, что вместо этой задачи можно решать следующую задачу оптимизации:

$$\min L_1(I), <A_i, I> = b_i, i = 1 \ldots m.$$

Эта задача решается за время $O(n^3)$. Но m должно быть намного больше, чем в предыдущей задаче. По некоторым оценкам $m \sim 4$ s.

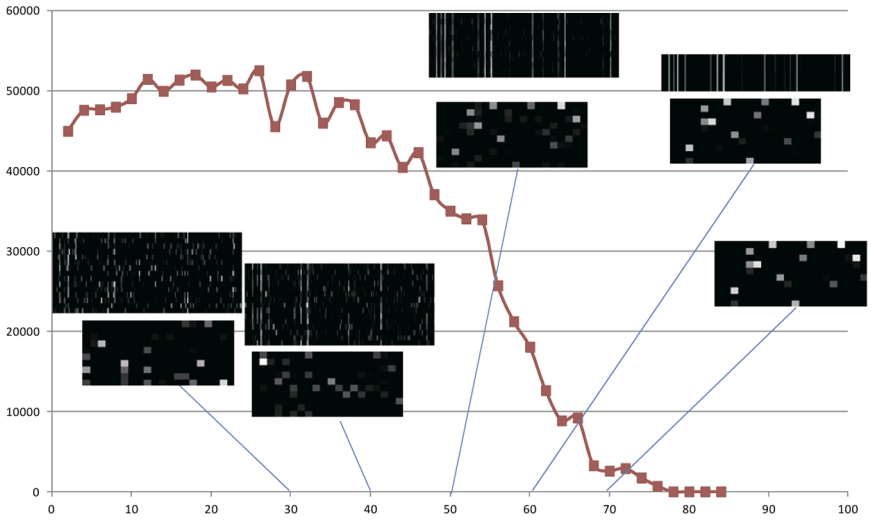


Рис. 1. График зависимости ошибки от m

Был проведен численный эксперимент. Был взят 20-редкий вектор из 200-мерного пространства. Было сгенерировано m векторов A_i и решена задача L_1 оптимизации. Была посчитана ошибка — расстояние между исходными восстановленным векторами по L_1 метрике. Эксперимент был повторен 15 раз, ошибки были просуммированы. На рисунке 1 показана зависимость суммарной ошибки от m.

На рисунках графика показаны восстановленные в каждом эксперименте вектора. По этому графику видно, что с увеличением числа m — количества граничных условий $<A_i, I>=b_i$ — постепенно увеличивается точность восстановления. Причем наибольшие по модулю компоненты восстанавливаются раньше.

Пусть история — это последовательность чисел на шкале времени. Разобьем шкалу на интервалы длины N, если на этом интервале не более s событий, то он является s-редкой последовательностью. Выберем m, сгенерируем m векторов A_i размерности N. Вычислим m скалярных произведений $<A_i, I>=b_i$, где I — интервал на временной шкале. Тогда пары (A_i, b_i), будут описанием истории событий на интервале времени. С течением времени информация о более давней истории должна забываться. Это достигается удалением произвольных пар (A_i, b_i) из истории. Для генерации ключа необходимо предоставить пары (A_i, b_i), которые позволят субъекту восстановить определенный интервал истории событий. Также система может сама восстановить интервал истории и предоставить его в качестве ключа. Аутентификация по данному ключу происходит путем восстановления фрагмента истории решением задачи L_1 оптимизации и сравнения предоставленного субъектом

фрагмента с тем, что известно системе о произошедших в тот период событиях. Система задает субъекту вопросы о произошедших в определенный интервал событиях. Правильные ответы повышают коэффициент доверия субъекту.

Оценка

Для хранения фрагмента истории системе нужно хранить лишь пары (A_i, b_i), которые могут вычисляться налету, без хранения самого фрагмента истории и вектора A_i. Обновление ключа происходит автоматически за счет накопления истории. Даже если злоумышленник будет слушать трафик между субъектом и системой, он не сможет повторить процесс аутентификации, так как будет знать лишь конечное число ответов на вопросы об истории сообщений.

Заключение

Разработка протокола новостной авторизации позволит по-новому взглянуть на процесс взаимодействия субъекта с системой. Аутентификация на основе истории событий применима в мультиагентных системах, при взаимодействии маломощных вычислителей.

Литература

1. *О. Н. Граничин, Д. В. Павленко*. Рандомизация получения данных и $\ell 1$-оптимизация (опознание со сжатием) // Автоматика и телемеханика. 2010. № 11. С. 3–28.
2. *D. Baron, M. B. Wakin, M. F. Duarte, S. Sarvotham*, and *R. G. Baraniuk.* Distributed Compressed Sensing // Technical Report ECE-0612, Electrical and Computer Engineering Department. Rice University, December 2006.

СТОЙКИЕ СХЕМЫ ШИФРОВАНИЯ С МАЛОЙ ДЛИНОЙ КЛЮЧА

Н. А. Молдовян

д-р тех. наук, зав. лабораторией криптологии НИО ПИБ СПИИРАН

E-mail: nmold@mail.ru

А. А. Горячев

канд. тех. наук, научный сотрудник, НИО ПИБ СПИИРАН

E-mail: alex_disp@gmail.com

А. В. Муравьев

аспирант, НИО ПИБ СПИИРАН

E-mail: muravevanton@gmail.com

Аннотация. В данной работе решается задача получения гарантированной стойкости шифрования при использовании секретных ключей малого размера путем объединения процедуры криптографического преобразования по секретному ключу с процедурой бесключевого шифрования. Предложен способ и протокол стойкого шифрования по разделяемому секретному ключу малого размера, который представляет интерес для практического применения в условиях ограниченности ключевого материала.

Введение

Криптографические схемы такие как симметричные шифры, предполагают использование защищенного канала связи, что резко увеличивает издержки, в том числе и временные. Стандартные протоколы симметричного шифрования обеспечивают гарантированную стойкость при использовании секретных ключей достаточно большого размера, например, 128 или 256 бит. Однако, на практике возникают случаи необходимости срочной передачи конфиденциальной информации по открытым каналам при наличии у отправителя и получателя разделяемого секретного ключа достаточно малого размера, например, от 32 до 56 бит. В таких случаях возникает техническая задача обеспечения достаточно высокого уровня стойкости шифрования, например, равного 2^{128} операций шифрования при использовании ключей, которые достаточно легко могут быть определены потенциальным нарушителем методом полного перебора по ключевому пространству.

Для решения поставленной задачи предлагается использовать процедуры коммутативного криптографического преобразования, не требующего использования разделяемых секретных ключей [1,2]. Недостатком криптографических схем, использующих процедуры бесключевого шифрования [3]

является то, что требуется обеспечить возможность аутентификации передаваемых сообщений. Недостаток устраняется за счет использования механизма аутентификации передаваемых сообщений по разделяемому ключу.

Протокол бесключевого шифрования

В рамках решения поставленной задачи, практический интерес представляет использование трехпроходного протокол Шамира (см. с. 516–517 в [1]), который позволяет передать секретное сообщение по открытому каналу без использования отправителем и получателем общих (разделяемых) секретных ключей.

Безопасная передача сообщений в указанном случае представляется возможной при условии выполнения процедур аутентификации шифр-текстов, передаваемых друг другу отправителем и получателем секретного сообщения в процессе выполнения процедуры бесключевого шифрования, по разделяемому секретному ключу.

В случае применения разделяемого секретного ключа в процедурах аутентификации сообщений имеется возможность только однократной попытки угадать секретный ключ и навязать легальному пользователю ложное сообщение. Вероятность того, что злоумышленник с первой попытки сможет угадать ключ и тем самым навязать легальному пользователю ложное сообщение достаточно мала даже в случае использования коротких разделяемых ключей и составляет 2^{-k}, где k — длина ключа в битах. Вероятности 2^{-32}, 2^{-40} и 2^{-56} пренебрежимо малы даже для достаточно критичных применений, поэтому представляется возможным во многих практических случаях выполнение аутентификации сообщений по ключам сравнительно малого размера. Для реализации данной принципиальной возможности требуется обеспечить неразрывность процедуры аутентификации и процедуры бесключевого шифрования для всех практически значимых вариантов потенциальных атак со стороны активного нарушителя.

Обеспечение неразрывности процессов аутентификации и передачи секретного сообщения

Далее будут рассмотрены две схемы, в каждой из которых обеспечивается неразрывность процессов аутентификации и передачи секретного сообщения.

Аутентификация шифр-текстов с использованием имитовставок

Наиболее простым способом аутентификации шифр-текстов в протоколе бесключевого шифрования является использование защитных контрольных сумм, в качестве которых можно использовать имитовставки I, вычисляемые по алгоритму ГОСТ 28147–89 [4]. Приемлемый для практического примене-

ния вариант протокола стойкого шифрования по разделяемому секретному ключу K малого размера включает следующие четыре шага:

1. Получатель секретного сообщения M генерирует 256-битовое случайное значение R_n, шифрует его по разделяемому ключу K с использованием алгоритма блочного шифрования ГОСТ 28-147-89 G и направляет отправителю секретного сообщения значение $C_n = G_K(R_n)$.
2. Отправитель генерирует 256-битовое случайное значение R_o, вычисляет значение $R = R_o + R_n$ и шифрует R по разделяемому ключу K с использованием алгоритма G: $C = G_K(R)$. Затем он вычисляет имитовставку I по значению M, используя случайное значение R в качестве ключа имитовставки, генерирует случайный неразделяемый ключ A и зашифровывает пару значений (M, I) в шифр-текст $C_1 = E_A(M, I)$ в соответствии с алгоритмом коммутативного шифрования E. Пару значений (C, C_1) отправитель передает получателю.
3. Получатель генерирует случайный неразделяемый ключ B ключ и зашифровывает значение C_1 в шифр-текст $C_2 = E_B(C_1)$ и вычисляет значение R, используя алгоритм расшифрования G^{-1} в соответствии со стандартом ГОСТ 28-147-89: $R = G^{-1}{}_K(C)$. Затем он зашифровывает значение C_2 по ключу K с использованием алгоритма блочного шифрования G в шифр-текст $C^*_2 = G_K(C_2)$, который передает отправителю.
4. Отправитель вычисляет шифр-текст C_2 по формуле $C_2 = G^{-1}{}_K(C^*_2)$ и преобразует значение C_2 по алгоритму расшифрования D, обратному по отношению к алгоритму коммутативного шифрования E, с использованием ключа A: $C_3 = D_A(C_2) = D_A(E_B(E_A(M, I))) = E_B(M, I)$. Значение C_3 отправитель передает получателю.

По значению шифр-текста C_3 получатель восстанавливает сообщение M и имитовставку I, вычисляет имитовставку по значениям M и сравнивает вычисленное значение имитовставки с ее значением, восстановленным из шифр-текста C_3. Если сравниваемые значения равны, то получатель делает вывод, что секретное сообщение было действительно отправлено подлинным отправителем.

Аутентификация шифр-текстов с использованием их шифрования по разделяемому ключу

Поскольку значения шифр-текстов, возникающих в протоколе бесключевого шифрования являются псевдослучайными (вычислительно неотличимыми от случайных значений), то их зашифрование не даст возможности потенциальному атакующему найти значение разделяемого секретного ключа. Описание шагов, предлагаемой схемы приведено ниже:

1. Отправитель сообщения M шифрует M по неразделяемому ключу A, получает шифр-текст $C_1 = E_A(M)$, зашифровывает значение C_1 по разделяе-

мому секретному ключу K с использованием алгоритма симметричного шифрования ГОСТ 28-147-89 в соответствии с формулой $S_1 = G_K(C_1)$ и направляет получателю секретного сообщения значение S_1 по открытому каналу.

2. Получатель расшифровывает шифр-текст S_1 по разделяемому ключу K, получает шифр-текст $C_1 = G\text{-}1_K(S_1)$, зашифровывает значение C_1 по неразделяемому ключу B по формуле $C_2 = E_B(C_1)$. Затем получатель зашифровывает значение C_2 по разделяемому секретному ключу K с использованием алгоритма G по формуле $S_2 = G_K(C_2)$ и направляет отправителю шифр-текст S_2 по открытому каналу.
3. Отправитель расшифровывает шифр-текст S_2 по разделяемому ключу K, получает шифр-текст $C_2 = G^{-1}{}_K(S_2)$. Затем, используя процедуру расшифрования D, преобразует шифр-текст C_2 по формуле $C_3 = D_A(C_2) = E_B(M)$ и посылает значение C_3 получателю.

Получатель восстанавливает сообщение M из полученного шифр-текста C_3 по формуле $M = D_B(E_B(M))$, также, как и в протоколе бесключевого шифрования. Стойкость трехшагового протокола шифрования по коротким ключам определяется стойкостью используемого алгоритма коммутативного шифрования. Разделяемый секретный ключ K служит для того, чтобы предотвратить атаки активного нарушителя, в которых нарушитель выдает себя за легального отправителя или получателя.

Выбор алгоритма коммутативного шифрования

В качестве симметричного алгоритма шифрования E, обладающего свойством коммутативности и обеспечивающего высокую криптостойкость предложенных протоколов, может быть использован алгоритм шифрования Полинга—Хеллмана [2], который основан на вычислительной трудности задачи дискретного логарифмирования по простому модулю. Стойкость алгоритма Полинга—Хеллмана настолько высока, насколько вычислительно трудна задача дискретного логарифмирования. Для обеспечения 80-битовой (128-битовой) стойкости требуется использовать простое число p размером не менее 1024 (2464) бит, такое, что разложение числа $p - 1$ содержит, по крайней мере, один большой простой множитель q размером не менее 160 (256) бит.

Заключение

Практическое использование предложенного способа относится к сценариям передачи секретного сообщения в условиях ограниченности ключевого материала. Частным случаем способ может быть применен в рамках слабой криптографии, т.е. разрешенной для свободного использования криптографии. Использование слабой криптографии позволяет повысить защищенность общественного информационного ресурса в целом. В качестве способа

повышения стойкости механизмов слабой криптографии можно применить разработанный в данной работе подход, использование которого может способствовать росту доверия массовых пользователей к слабой криптографии и тем самым задействовать на практике ресурсы слабой криптографии при решении практических задач защиты общественного информационного ресурса.

Литература

1. *Schneier B*. Applied Cryptography: Protocols, Algorithms and Source Code (Second Edition). New York: John Wiley & Sons, 1996. 758 p.
2. *Hellman M. E., Pohling S. C*. Exponentiation Cryptographic Apparatus and Method // U. S. Patent. No. 4. 424, 414. 3 Jan. 1984.
3. *Молдовян Н. А*. Введение в криптосистемы с открытым ключом. СПб: БХВ-Петербург, 2007. 286 с.
4. ГОСТ 28147–89. Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. М.: Изд-во стандартов, 1989. 20 с.

BIGHEX: РЕАЛИЗАЦИЯ RSA НА JAVASCRIPT ДЛЯ ВЕБ-ПРИЛОЖЕНИЙ

М. В. Баклановский
ст. преп. кафедры системного программирования СПбГУ
E-mail: m.baklanovsky@spbu.ru

Д. В. Луцив
ст. преп. кафедры системного программирования СПбГУ
E-mail: dluciv@math.spbu.ru

Аннотация. В работе описаны проблемы, с которыми сталкиваются веб-программисты при использовании зашифрованных каналов связи между клиентом и сервером. В качестве решения предложена лёгкая реализация ассиметричного шафрования (алгоритм RSA) на JavaScript для клиентской и серверной сторон веб-приложений. Рассмотрены особенности клиентской и серверной платформ JavaScript. Приведены сравнения производительности библиотек с известными аналогами.

Введение

Существенная часть современных веб приложений хранит и обрабатывает на стороне сервера пользовательские данные, присылаемые с клиентской стороны, и, напротив, представляет клиенту часть его данных, хранимых или обрабатываемых на сервере.

Для защиты от утечек и искажений конфиденциальной информации применяются различные криптографические решения, большинство из которых обладают известными сильными и слабыми сторонами, перечисленными в работе.

Авторами предложено собственное решение, заключающееся в реализации шифрования на языке JavaScript и продемонстрирована достаточная для ряда задач производительность решения при использовании современных веб-браузеров.

Проблематика

Ниже рассмотрены принятые для защиты пользовательских данных подходы.

Отсутствие защиты

Этот подход имеет полное право на существование в тех случаях, когда среда передачи данных полностью подконтрольна разработчикам веб-прило-

жений. Например, это может быть внутриведомственной сетью в организации с высоким уровнем производственной дисциплины. В случае, когда есть уверенность в том, что никакие прослушивающие трафик устройства не будут внедрены в сеть, а существующие узлы сети не будут соответствующим образом запрограммированы, передача пользовательских данных в незашифрованном виде оправдана. Другая допустимая ситуация — когда пользователи системы заведомо готовы мириться с тем, что передаваемые данные будут перехвачены, и, возможно, искажены. Наконец, типичная ситуация — когда небольшая ценность или сам характер открытых данных делают любые усилия по их перехвату и/или искажению нецелесообразными.

Основное, и, пожалуй, единственное достоинство подхода — простота, но в ряде случаев оно оказывается решающим. Недостаток же, тоже оказывающийся решающим — сильно ограниченная область применения.

Защита при аутентификации с применением хэш-функций

Этот способ основан на одном из основных свойств хэш-функций — необратимости (строго говоря, «тяжёлой обратимости»). Для того, чтобы представиться ресурсу, пользователь не передаёт пароль в открытом виде, а передаёт значение хэш-функции на нём. Перехват трафика позволяет злоумышленнику воспроизвести передачу данных от лица пользователя, поэтому для повышения безопасности сервер должен передавать пользователю одноразовую соль, которую пользователь должен использовать вместе с паролем перед вычислением хэш-функции.

Этот способ (и его вариации), как и предыдущий способ, достаточно прост, и в своё время был довольно популярен. Привлекательным его сделали в первую очередь именно простота и элегантность. Однако за последние несколько лет многократно выявлялись дефекты (с криптографической точки зрения) хэш-функций: для популярных хэш-функций создавались алгоритмы поиска коллизий [1], подробно описывались характер и масштаб уязвимостей[1]. Сейчас общедоступные большие радужные таблицы (сама идея которых не нова [2]) часто позволяют искать аргументы по значению функций за разумное время.

Использование ассиметричной криптографии

Универсальный способ, который может быть использован непосредственно при передаче данных небольшого объёма, либо в сочетании с симметричной (для передачи ключа симметричного шифра) при передаче данных значительного объёма.

---

[1] Bruce Schneier. Cryptanalysis of SHA-1. 18.02.2005 https://www.schneier.com/blog/archives/2005/02/cryptanalysis_o.html

На данный момент ассиметричные криптографические алгоритмы считаются надёжными, т.е. при помощи обычных (не квантовых) вычислителей ассиметричные шифры за разумное время не взламываются. Последнее достижение — взлом шифра с ключом длиной 768 бит [3].

Значительная часть практического применения ассиметричной криптографии в компьютерных сетях приходится на использование протоколов SSL (TLS) [4] и основанных на них. Также одной из известных и популярных систем шифрования является система PGP [5] и её клоны, например, GPG.

Ассиметричные криптосистемы завоевали такую популярность, что Тьюринговская премия 2002 года была вручена создателям RSA [6] — Ривесту, Шамиру и Адлеману — за ту практическую ценность, которую благодаря их работе приобрели криптосистемы с отрытым ключом.

Пользователи WWW, в основном, сталкиваются с протоколом HTTPS. HTTPS, как и многие другие основанные на SSL протоколы, подразумевает использование сертификатов, подписанных доверенными центрами сертификации. Получение таких сертификатов платно, а их использование ограничено, и требует дополнительных усилий при веб-программировании.

Как и на любую другую популярную систему шифрования, на SSL регулярно предпринимаются атаки, как на концептуальном уровне[1,2], так и с использованием уязвимости конкретных реализаций[3].

Реализация на языках высокого уровня

Полезность систем ассиметричного шифрования может быть поднята на ещё большую высоту, если вычисления, необходимые для их реализации, станут доступными на языках высокого и сверхвысокого уровня. В плане веб-программирования это первую очередь касается таких популярных и обладающих мощным пропедевтическим потенциалом языков, как JavaScript [7].

Рассмотрим вычислительные задачи, с решением которых сталкивается реализация криптосистемы RSA. Необходимые вычисления можно разделить на 2 группы:

1. Однократные вычисления при построении экземпляра криптосистемы. К ним относятся операции генерации 2-х псевдопростых чисел (pp_1 и pp_2), вычисление их произведения (mod), вычисление фунции Эйлера для полученного произведения ($\varphi\,(\mathrm{mod}) = (pp_1 - 1)\,(pp_2 - 1)$), нахожения мультипликативного обратного (exp) для публичной экспоненты (обычно, $2^{16} + 1$) по модулю mod. Этот этап пользователю не виден и может занимать любое разумное с точки зрения разработки системы время.

---

1 Rohit T. SSL Attacks. 28.10.2013 http://resources.infosecinstitute.com/ssl-attacks/

2 Dan Goodin. Two new attacks on SSL decrypt authentication cookies. 14.03.2013 http://arstechnica.com/security/2013/03/new-attacks-on-ssl-decrypt-authentication-cookies/

3 The Heartbleed Bug http://heartbleed.com/

2. Вычисления, производимые при каждом использовании криптосистемы, например, рукопожатие и авторизация в начале очередной коммуникационной сессии. Большие задержки на этапе крайне нежелательны.

Таким образом, возникает задача реализации вычислений 2-й группы таким образом, чтобы они выполнялись на современных вычислительных устройствах за времена, укладывающиеся в нечёткие понятия «очень быстро» или «мгновенно» с точки зрения конечного пользователя, т. е. за времена порядка 1 секунды или быстрее.

Производительность реализации RSA на JavaScript

Авторами реализована 2-я группа вычислений на языке JavaScript для клиентской стороны (т. е. для веб-браузеров) и для веб-сервера IIS7. При программировании был использован ряд алгоритмов и техник, описанных в [8, 9].

На клиентской стороне времена выполнения основной вычислительной операции (возведение в степень по модулю) на машине с процессором Intel Core i5-2400 3,1 ГГц в браузере Internet Explorer 11 для различных длин модуля и публичной экспоненты = 65 537 выполняются за время, незаметное пользователю (см. таблицу 1). Учитывая то, что модули (публичные ключи RSA) с длиной от 1500 бит считаются сегодня вполне надёжными, результат можно признать соответствующим требованию даже с учётом возможности исполнения на более медленных мобильных устройствах.

На серверной стороне использован JScript.NET[1]. Для того, чтобы поставить содержательный эксперимент, мы не пользовались встроенным классом .NET Framework 4+ `BigInteger` (фактически он позволяет повысить производительность в несколько десятков раз). В тесте, полностью аналогичном (в т.ч. по характеристикам машины) приведённому выше, мы получили времена выполнения, также представленные в таблице 1.

Таблица 1

Время выполнения основной операции шифрования/расшифровки

Длина ключа (бит)	**IE 11 (мс)**	IIS 7 (мс)
1200	1	11
1600	3	25
2000	4	31
2400	6	41
3200	7	93
4000	11	125
6000	22	281

Из сравнения результатов приведённых тестов можно сделать довольно неожиданный вывод о том, на конкретном примере производительность машинного кода, сгенерированного JIT-компилятором современного браузера, превосходит производительность кода, сгенерированного JIT-компилятором плат-

[1] Реализация диалекта ECMA Script для платформы Microsoft .NET, поддерживает статическую типизацию.

формы .NET на порядок, не смотря на то, что для .NET был использован язык Jscript .NET с явным статическим указанием типов переменных. При тестировании разных браузеров лидирует по скорости интерпретатор Chakra браузера Microsoft Internet Explorer 11, достойные результаты показывают также Google V8 и Mozilla IonMonkey.

BigHex

Созданная библиотека длинной арифметики и криптографических вычислений названа BigHex.

Выполненная разработка снабжена тестовым набором из 236 000 примеров для используемых операций длинной арифметики (суммарный объём тестов превышает 400 МБ). Весь тестовый набор успешно выполнен на серверной стороне и во всех популярных веб-браузерах на различных платформах (более 20 полных выполнений).

Все файлы, включая тестовый набор, опубликованы по адресу: http://spisok.math.spbu.ru/util/BigHex/. Тесты клиенской стороны можно запустить прямо с этого адреса. Для скачивания создан архив: http://spisok.math.spbu.ru/util/BigHex/BigHex.zip. Исходный код библиотеки распространяется под лицензией MIT: http://spisok.math.spbu.ru/util/BigHex/BigHex_licence.txt

Практическое применение

Клиентский и серверный код библиотеки BigHex планируется применить для ассиметричного шифрования на сайте конференции СПИСОК. Сайт конференции — хороший пример области применения BigHex: на данный момент единственный требующий шифрования момент — аутентификация, и для её защиты гораздо легче использовать компактную и легко встраиваемую библиотеку на JavaScript, нежели выполнять стандартный набор действий по переводу сайта на использование протокола HTTPS.

Для практического применения на серверной стороне будет использована стандартная библиотека .NET версии 4, содержащая класс `BigInteger`. Это позволит разгрузить сервер, так как у сайта конференции, не смотря на её молодость, уже сейчас довольно много зарегистрированных пользователей.

Отметим, что многие сайты, реализованные с использованием ASP .NET, используют платформу .NET версий 2 и 3.x. Благодаря простоте и открытому исходному коду BigHex, авторы этих сайтов смогут воспользоваться для повышения производительности и другими реализациями длинной арифметики, в частности реализацией `java.math.BigInteger` из стандартной библиотеки J#[1], или же применить предложенную реализацию на Jscript.NET, которая тоже обеспечит производительность на приемлемом уровне.

---

[1] Реализация стандартной библиотеки Java 1.3 и аналогичного Java языка для платформы .NET версии 2.

Заключение

В работе описаны стандартные подходы при реализации систем шифрования для веб-программирования, приведены тесты производительности предложенной авторами лёгкой реализации для веб на языке JavaScript, описана практическая сторона применения библиотеки.

Следует отметить, что, помимо аутентификации, реализация RSA на JavaScript может быть использована для подписывания сообщений. Кроме того, область применения библиотеки не ограничивается заменой SSL. Популярность общественных или создаваемых организациями веб-интерфейсов к ящикам электронной почты делает актуальной задачу реализации на JavaScript систем шифрования почты, совместимых с PGP/GPG.

Наконец, популярность JavaScript в последние годы привела к созданию микроконтроллерных платформ, программируемых на нём[1,2]. Встраиваемые системы, основанные на микроконтроллерах, применяются в умных домах, робототехнике и прочих областях, где уязвимость может позволить злоумышленнику нанести прямой физический ущерб. Очевидно, что безопасность основанных на микроконтроллерах встраиваемых систем также крайне актуальна.

Литература

1. *Xiaoyun Wang, Yiqun Lisa Yin, Hongbo Yu*. Finding Collisions in the Full SHA-1 // Lecture Notes in Computer Science. Vol. 3621. 2005. Pp. 17–36. http://www.infosec.sdu.edu.cn/uploadfile/papers/Finding%20Collisions%20in%20the%20Full%20SHA-1.pdf
2. *Hellman M. E*. A cryptanalytic time-memory trade-off // IEEE Transactions on Information Theory. 26 (4): 401–406. 1980.
3. *Thorsten Kleinjung, Kazumaro Aoki, Jens Franke, Arjen K. Lenstra, Emmanuel Thomé, Joppe W. Bos, Pierrick Gaudry, Alexander Kruppa, Peter L. Montgomery, Dag Arne Osvik, Herman te Riele, Andrey Timofeev, Paul Zimmermann*. Factorization of a 768-Bit RSA Modulus // Lecture Notes in Computer Science. Vol. 6223. 2010. Pp. 333–350.
4. *T. Dierks, E. Rescorla*. The Transport Layer Security (TLS) Protocol. RFC 5246 http://tools.ietf.org/html/rfc5246
5. *Simson Garfinkel*. PGP: Pretty Good Privacy // O»Reilly Media, Inc. 1995. 393 pp.
6. *Rivest R. L., Shamir A., Adleman L*. A method for obtaining digital signatures and public-key cryptosystems // Communications of the ACM. New York, NY, USA: ACM, 1978. Vol. 21. No. 2. Feb. 1978. Pp. 120–126.
7. ECMAScript® Language Specification 5.1 Edition // Ecma International, 2011 http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf
8. *Кнут Д. Э*. Искусство программирования. Т. 2: Получисленные алгоритмы. 3-е изд. М.: Вильямс, 2007. 832 С.
9. *Уоррен Г*. Алгоритмические трюки для програмистов. М.: Вильямс, 2004. 282 с.

[1] Проект Tessel.io https://tessel.io/

[2] Проект Espruino http://www.espruino.com/

ИСПОЛЬЗОВАНИЕ НИЗКОСКОРОСТНЫХ УСТРОЙСТВ ДЛЯ ШИФРОВАНИЯ ВИДЕОДАННЫХ

Р. Ш. Фахрутдинов

заведующий лабораторией безопасности информационных систем СПИИРАН, к.т.н.

E-mail: fahr@cobra.ru

Аннотация. Широкое использование видеоинформации [1] делает актуальной задачу обеспечения её безопасности. Рассматривается возможность применения малопроизводительных аппаратно-программных устройств для шифрования видеоданных.

Введение

Различные видеоприложения прочно вошли в нашу жизнь. Мы совершаем видеозвонки, смотрим цифровое ТВ, просматриваем видеоновости, выкладываем события личной жизни на видеохостинге, нашу личную безопасность охраняют с помощью видеонаблюдения в метро, на вокзалах и т. д. Организации используют видеоконференции, чтобы провести оперативное совещание без выезда сотрудников, врачи проводят видеоконсультации и даже дистанционные операции с помощью видеокамеры.

Возникает вопрос, насколько безопасно для наших личных данных хранить свои видеозаписи на общем видеохостинге (пусть даже и под своим логином и паролем), насколько безопасна видеосвязь, нет ли возможности вмешаться в работу камер видеонаблюдения и т. п.

Угрозы и методы противодействия

Основными угрозами являются :

- Раскрытие конфиденциальности, которое заключается в установлении посторонними лицами существенной информации о видеозаписи (раскрытие всего видеоряда или важной её части, установление персоналий из видеоряда, обстоятельств съёмки и т. д.);
- Искажение, под которым понимается изменение видеоряда, нарушение последовательности действий или иное воздействие, существенно влияющее на восприятие видеоинформации;
- Изменение подлинности даёт возможность нарушителю удалить авторские метки или заменить их. Кроме того, возможна подмена подлинности (например, возможность выполнять видеозвонки от имени другого человека и т. д.).

Для противодействия этим угрозам, возможно использование следующих возможностей:

- Использование по возможности закрытых каналов связи (VPN, ssl, cryptochannel и т. д.).
- Т. н. «накладное» шифрование видеоинформации.
- Аутентификация пользователя.
- Применение селективного шифрования [2].

С точки зрения места реализации конкретных технических мероприятий, возможно усиление защищённости инфраструктуры и абонентского оборудования.

По ряду причин, усиление защищённости инфраструктуры усложнено ввиду ряда причин — дороговизны, неочевидности этих вложений с точки зрения их владельцев (компании сотовой связи, провайдеры услуг, операторы линий связи и пр.), их заинтересованности с точки зрения контроля проходящего через них контента (владельцы социальных сетей, видеохостеры, сервисы видеозвонков).

Однако и со стороны абонентского оборудования существует ряд возможностей, реализуя которые можно повысить защищённость видеокоммуникаций:

- Использование пользовательских закрытых каналов связи (VPN, cryptochannel, SSH, SSL и т. д.).
- Т. н. «накладное» шифрование с помощью программных, программно-аппаратных или аппаратных комплексов.
- Обязательная аутентификация.
- Внедрение селективного шифрования.

Как видим, большинство методов противодействия в том или ином виде включают шифрование. В настоящее время большинство абонентских устройств имеют производительность, достаточную для обеспечения безопасности, в т.ч. и для выполнения шифрования информации (текст, аудио, видео). К сожалению, в большинстве устройств нет специальных возможностей, обеспечивающих недоступность ключей шифрования для остального ПО абонентского устройства, поскольку ключи хранятся в обычной оперативной памяти. В случае заражения ПО на смартфоне компьютерным вирусом, ключи могут быть скомпроментированы, кроме того, может быть произведена модификация ПО с целью оперативной отправки ключей шифрования при их замене.

Использование шифрования

Поэтому более надёжным является использование для шифрования аппаратных устройств, например, смарт-карт. Эти устройства могут быть подключены к большинству оконечных абонентских устройств либо напрямую

(смартфоны, планшеты с 3G/4G), либо через совместимые интерфейсы (USB, PCMCIA).

У смарт-карт имеется энергонезависимая память, доступ к которой закрыт «снаружи». В этой памяти хранятся ключи шифрования, ПО через интерфейс работы со смарт-картой может только выбрать ключ шифрования (по его номеру) и выполнить шифрование/расшифрование, получив результат.

В смарт-картах может быть реализован один или несколько алгоритмов шифрования, в том числе и асимметричные, алгоритмы электронной подписи. Смарт-карты имеют очень низкое энергопотребление и низкую себестоимость [3].

К сожалению, производительность шифрования смарт-карт является невысокой и составляет от нескольких байт в секунду для асимметричных алгоритмов (RSA), до нескольких десятков килобайт в секунду для симметричных алгоритмов (AES, ГОСТ) [4].

Все это затрудняет использование смарт-карт для полного шифрования/расшифрования всех данных, которые используются в современном абонентском оборудовании. Кроме того, смарт-карта выполняет ряд функций по обеспечению связи (в смартфоне, например) и не может выполнять только функции шифрования.

С точки зрения защиты видеоданных, смарт-карта имеет низкую производительность шифрования, что не позволяет использовать её напрямую для этих целей. Однако возможно использование смарт-карты для генерации и шифрования сеансового ключа. При этом реальное шифрование видеопотока стойкими алгоритмами выполняется центральным процессором абонентского устройства по симметричному алгоритму, а смарт-карта в передающем и приёмном устройстве выполняет функцию шифрования и расшифрования сессионного ключа. При этом возможно использование 2-ключевой криптографии, когда сгенерированный на одной из сторон сессионный ключ шифруется на открытом ключе другого устройства, а второе устройство будет расшифровывать его с помощью свтоего закрытого ключа.

При этом можно дискредитировать только сессионный ключ, но не ключ, который хранится в смарт-карте.

К сожалению, у данной схемы есть существенный недостаток, который заключается в использовании смарт-карты только в момент генерации сессионного ключа или при его получении из потока в случае расшифрования. Это позволяет использовать одну и ту же смарт-карту для «раздачи» ключевой информации многим потребителям (т. н. cardsharing). Так, например, в случае использования смарт-карты для получения контента при широковещательной передаче (платное ТВ), возможна покупка одной смарт-карты и организации сетевого протокола для получения многими пользователями «легальной» ключевой информации.

Избежать cardsharing-а можно путём частой смены ключа, что используется в некоторых системах платного ТВ с использованием смарткарт [5].

При использовании селективного шифрования видеоинформации, возможна схема, при которой смарт-карта является источником ключевой информации в течении всего процесса шифрования/расшифрования видеоинформации, что не позволит выполнить cardsharing, а отсутствие сессионного ключа не позволит его скомпроментировать.

При этой схеме, на смарт-карты при её производстве записывается большое число готовых ключей шифрования. При установлении сеанса, выбирается номер используемого ключа (внутренний сессионный ключ), на основе которого формируется ключевая последовательность. Номер сессионного ключа в закрытом или открытом виде (можно также использовать 2-х ключевую криптографию) передаётся приёмной стороне. Само селективное шифрование выполняется центральным процессором абонентского устройства. На приёмной стороне в смарт-карте используется тот же самый ключ и формируется ключевая информация, позволяющая произвести расшифрование центральным процессором абонентского устройства. При этом функции кодирования видеоинформации в сжатый формат и шифрования выполняются одновременно, как и функции декодирования/расшифрования, что не позволяет пользователю изготовить сжатую незащищённую копию видеоинформации (при использовании в системах платного ТВ, например).

Использование одного экземпляра легальной смарт-карты затруднено из-за большого объёма ключевой информации (десятки килобайт в секунду) и может быть ещё более затруднено при переносе существенной части функционала селективного шифрования на смарт-карту.

Заключение

Были рассмотрены некоторые примеры использования малопроизводительных устройств (смарт-карт) для защиты видеоинформации с помощью шифрования, в том числе с использованием селективных методов.

Литература

1. *Roy Germano, Woodward Chair, Sarah Lawrence.* Analytic Filmmaking: A New Approach to Research and Publication in the Social Sciences/ Roy Germano, Woodward Chair, Sarah Lawrence // http://www.roygermano.com/AnalyticFilmmaking.pdf. 2014. С. 1.
2. *A. M. Alattar* and *G. I. Al-Reg*. Evaluation of selective encryption techniques for secure transmission of MPEG video bit-streams // IEEE Symposium on Circuits and Systems. 1999. Pp. 340–343.
3. *Michael Tunstall*. Secure Cryptographic Algorithm Implementation on Embedded Platforms // Royal Holloway, University of London Egham, Surrey TW20 0EX, England. Technical Report RHUL-MA-2007–5. 2007. P. 42–44.
4. Triple DES/AES Encryption Libraries / MICROCHIP Tech Data // http://ww1.microchip.com/downloads/en/DeviceDoc/01033B%2022.pdf. P. 1.
5. *Pallab Dutta*. Java Card For PayTV Application / Pallab Dutta // (IJCSIS) International Journal of Computer Science and Information Security. 2013. Vol. 11. No. 6.

Вероятностные графические модели, нечеткие системы, мягкие вычисления и социокомпьютинг

Юсупов
Рафаэль Мидхатович

д.т.н., профессор, чл.-корр. РАН
директор СПИИРАН

Тулупьев
Александр Львович

д.ф.-м.н., профессор кафедры информатики СПбГУ
заведующий лабораторией ТиМПИ СПИИРАН

РАЗВИТИЕ ЭЛЕКТРОННОЙ ПУБЛИКАЦИИ[1]

А. В. Торопова

аспирант кафедры информатики математико-механического ф-та СПбГУ

E-mail: alexandra.toropova@gmail.com

Аннотация. В работе рассматривается состояние электронной публикации в мире, в том числе и России. Приводятся примеры преимущества электронной публикации перед печатной. Рассказывается о движении «Открытый Доступ», постепенно набирающем все больше сторонников. Прогнозируется дальнейшее развитие электронной публикации.

Введение

С появлением возможности публикации в интернете традиционные печатные издания стали уходить на второй план, все большее количество ученых стали пользоваться электронными изданиями для того, чтобы делиться результатами своих исследований. Это легко поддается объяснению, ведь электронная публикация обладает рядом значимых преимуществ перед печатной таких, как высокая скорость, оперативный доступ в любое время и практически из любого места, увеличение цитируемости авторов, возможность обращаться к более широкой аудитории, расширять и дополнять материалы, экономить средства, затрачиваемые на выпуск печатной продукции, автоматизировать процессы издательства, защита от недобросовестных рецензентов (так как уже известны случаи, когда после отклонения или задержки статьи похожие результаты появлялись у других авторов [1]), даже то, что электронная публикация более экологична, и многие другие. В связи с этим происходит перенос деятельности ученых и научных организаций в онлайн. Традиционные печатные формы опубликования научных работ отходят на второй план, и, возможно, в не таком далеком будущем исчезнут вовсе.

Ученым намного проще воспользоваться электронными источниками, которые можно быстро и не отходя от рабочего места найти с помощью интернета, чем пойти в библиотеку в поисках необходимой для исследования литературы или, например, выписать нужный научный журнал и дождаться, когда им его пришлют, поэтому они осуществляют подбор материалов для своих исследований с помощью традиционных поисковых механизмов в интернете, а также (у кого есть доступ), с помощью таких электронных платформ как Web of Science [16], SCOPUS [12], MathNet.ru [9] и др.

[1] Работа частично поддержана грантом РФФИ № 12-01-00945-а.

Открый Доступ

Сейчас все большее распространение получает движение за «Открытый Доступ» [3, 8, 10], смысл этого движения заключается в том, чтобы ученые имели возможность бесплатного доступа к научным публикациям. У открытого доступа есть два пути развития — это «зеленый» и «золотой» (Green Road и Golden Road). Разница состоит в том, что в первом случае это по сути «самоархивирование», то есть авторы публикуют свои работы в интернете в свободном доступе, при этом возможна параллельная публикация в традиционных изданиях. Необходимые для этого средства обычно выделяются из грантов, либо организациями, в которых работают ученые. В рамках второго подхода все затраты на публикацию научных работ несут издатели. Такие модели также финансируются научными организациями и организациями, выдающими гранты [3].

Жители англоязычных странах могут воспользоваться всеми благами открытого доступа, в то время как в остальных же странах нужно что-то предпринять, чтобы люди, даже не владеющие английским языком, могли бы иметь доступ к результатам научных исследований, ведь они также платят налоги, часть которых вкладывается в том числе и в науку.

В странах Азии, например, создаются специальные репозитории, организовываются музеи и библиотеки, но это довольно небольшая часть информации, если сравнивать с тем, что опубликовано на английском языке [13].

В России лучшие результаты в этой области показывает eLIBRARY.RU, крупнейшая в России электронная библиотека научных публикаций, обладающая богатыми возможностями поиска и получения информации. Библиотека интегрирована с Российским индексом научного цитирования (РИНЦ) — созданным по заказу Минобрнауки РФ бесплатным общедоступным инструментом измерения и анализа публикационной активности ученых и организаций. Платформа eLIBRARY.RU была создана в 1999 году по инициативе Российского фонда фундаментальных исследований для обеспечения российским ученым электронного доступа к ведущим иностранным научным изданиям. С 2005 года eLIBRARY.RU начала работу с русскоязычными публикациями и ныне является ведущей электронной библиотекой научной периодики на русском языке в мире. Свыше 1200 российских научных журналов размещены в бесплатном открытом доступе [7].

Будущее развитие электронной публикации

Но сейчас постепенно происходят процессы, при которых формат статьи, даже электронный, не может дать всех преимуществ, которые возможны при использовании современнных веб-технологий. Если статья — это попытка заморозить и отобразить научные процессы и результаты, то Веб позволяет

открыть окна мастерской, где происходят эти процессы, показывая как происходит научная деятельность, стирая различия между процессом и результатами [11]. Ученые уже делятся своими исследовательскими данными в таких хранилищах, как GenBank, Dryad и figshare, используют репозитории такие, как GitHub, чтобы делиться кодом. Некоторые важные научные дискуссии могут проходить на социальных медиа-платформах, например в Twitter [15].

Поэтому создание такой платформы, которая дала бы ученым возможность делиться результатами своих исследований, позволяя представлять их не только в форме обычных электронных статей, но и использовать интерактивную графику, видео и т.п., редактировать их, обмениваться мнениями с коллегами и читателями, а, возможно, также агрегировать нужную информацию из других источников, например из обсуждений каких-либо научных проблем в социальных сетях.

Заключение

В данной работе было рассмотрено состояние электронной публикации в мире, в том числе и России. Были приведены преимущества электронной публикации перед печатной. Рассказано о движении «Открытый Доступ», постепенно набирающем все больше сторонников. Спрогнозировано дальнейшее развитие электронной публикации.

Литература

1. *Богданова И. Ф.* Онлайновое пространство научных коммуникаций // Социология науки и технологий. 2010. № 1.
2. *Веселаго В. Г., Елизаров А. М., Сюнтюренко О. В.* Российские электронные научные журналы — новый этап развития, проблемы интеграции // Электронные библиотеки. 2005. Т. 8. № 1.
3. *Литвинова Н. Н.* Научные публикации в Интернете: соотношение ограниченного (платного) и свободного доступов / [Электронный ресурс]. Режим доступа: http:// http://eqworld.ipmnet.ru/ru/info/sci-edu/Litvinova2005.htm.
4. *Полянин А. Д.* Электронные публикации и основные физико-математические ресурсы Интернета / А. Д. Полянин, А. И. Журов [Электронный ресурс]. Режим доступа: http://eqworld.ipmnet.ru/ru/info/sci-edu/PolyaninZhurov2007.htm/.
5. *Cockerill M.* Make indexing fast and fair // Nature News. 2013.
6. *Cyzyk M., Choudhury S.* A Survey and Evaluation of Open-Source Electronic Publishing Systems. JScholarship. April 2008.
7. eLIBRARY.RU — Электронный ресурс: http://elibrary.ru/.
8. *Johnson R.* Open Access: Unlocking the Value of Scientific Research (2004).
9. MathNet.ru — Электронный ресурс: http://www.mathnet.ru/.
10. *Noorden R.* Open access: The true cost of science publishing // Nature News. 2013.
11. *Priem J.* Beyond the paper // Nature News. 2013.
12. SCOPUS — Электронный ресурс: http://www.scopus.com/home.url.
13. *Sipp D.* Translate local journals // Nature News. 2013.

14. *Smecher A*. The Future of the Electronic Journal // NeuroQuantology. 2008. Vol. 6. No. 1. Pp. 1–6.
15. Twitter — Электронный ресурс: https://twitter.com/.
16. Web of Science — Электронный ресурс: http://thomsonreuters.com/products_services/science/science_products/a-z/web_of_science/.

АНАЛИЗ РАСПРОСТРАНЕНИЯ ВРЕДОНОСНОГО КОНТЕНТА СРЕДИ ПОЛЬЗОВАТЕЛЕЙ СОЦИАЛЬНЫХ МЕДИА[1]

А. А. Азаров

к.т.н., зав. лаб. математического моделирования социальных процессов, МГГУ им. М. А. Шолохова, научный сотрудник лаборатории ТиМПИ, СПИИРАН

E-mail: f2522255@yandex.ru

А. А. Фильченков

к.ф.-м.н., с.н.с. лаборатории математического моделирования социальных процессов, МГГУ им. М. А. Шолохова, научный сотрудник кафедры КТ СПбНИУИТМО

E-mail: aaafil@mail.ru

М. В. Абрамов

аспирант, лаборант-исследователь лаборатории математического моделирования социальных процессов, МГГУ им. М. А. Шолохова

E-mail: mva16@list.ru

Аннотация. В докладе рассматривается проблема анализа распространения вредоносного контента среди пользователей социальных медиа по средству репостов, лайков, ретвитов и д.р. Применение ранее разработанной методики анализа защищенности пользователей информационных систем от социо-инженерных атак злоумышленника позволяет получать вероятностные оценки распространения вредоносного контента среди пользователей социальных медиа.

Введение

Сохранение конфиденциальности корпоративной информации является одной из наиболее важных задач в современной конкурентной борьбе. Абсолютное большинство утечек конфиденциальных данных сопряжено с многомиллионными потерями организаций и громкими скандалами [10]. Одним из самых серьезных вызовов службам безопасности является воздействие злоумышленников на сотрудников организации с целью получения конфиденциальной информации. Эта проблема имеет тесное отношение к вербовке шпионами агентов на территории противника, а потому имеет долгую историю, однако в современности она в синтезе с современными технологиями приобрела специфику, увязывающую ее с безопасностью всей сети.

[1] Исследование поддержано грантом РФФИ на 2014–2016 гг., проект № 14-07-00694-а.

Имеются работы, посвященные проблеме манипулятивного воздействия на людей (социальная инженерия, социо-технические воздействия) [11], а также посвященные проблеме защите информации от атак на информационную систему (взлом корпоративных сетей, DDoS атаки и так далее) [3, 4, 9]. Вместе с тем исследования, которые бы увязывали проблему защиты информации от подобного рода воздействий на ее пользователей, не получили достаточного распространения.

Целью работы является развитие моделей комплекса «информационная система — критичные документы — персонал — злоумышленник» на основе применения вероятностно-реляционных моделей [2, 5–8], а также представление данных моделей для описания распространения вредоносного контента среди пользователей социальных медиа.

Описание моделей

При описании модели информационной системы организации необходимо указать на ряд ограничений. Доклад посвящен, в первую очередь, вопросам подверженности пользователей манипулятивным и суггестивным воздействиями, но не воздействиям на программно-технической составляющей. В докладе рассматривается информационная модель «информационная система — критичные документы — персонал — злоумышленник», которая является развитием модели «информационная система — персонал — критичные документы» элементами которой являются:

1) информационная сеть с со связями между хостами;
2) пользователи с психологическими особенностями и доступом к хостами;
3) критические документы, размещенные на хостах;
4) злоумышленник, который осуществляет социо-атакующее воздействие на пользователей системы на основе одного из известного ему атакующего воздействия, ограниченный в ресурсах и времени атаки.

Такое представление содержит следующие модели: модель пользователя, модель устройства информационной системы, модель критичного документа и модель злоумышленника. Рассмотрим подробнее представленные модели.

Предположим, что в системе есть k устройств, l пользователей, v уязвимостей пользователей.

Модель пользователя включает профиль уязвимостей пользователя, содержащий степени проявления уязвимостей пользователя, доступ пользователя к устройствам информационной системы, а также связи с другими пользователями. Таким образом, модель имеет вид:

$$U_i = (\{V_j^i\}_{j=1}^{v};\ \{Ac_j^i\}_{j=1}^{k};\ \{L_k^i\}_{k=1}^{l}),$$

где $V_j^i(D)$ — степень проявления j-ой уязвимости i-ого пользователя, $\{Ac_j^i\}_{j=1}^{k}$ — наличие доступа у i-ого пользователя к устройству под номером

j, $\{L_l^i\}_{l=1}^n$ — наличие и вес связи у рассматриваемого пользователя с l-ым пользователем.

Необходимо дать несколько комментариев о профиле уязвимостей пользователей. Согласно проведенному пилотному исследованию [1], гипотезой которого была взаимосвязь уровнем проявления психологических особенностей личности и степенью выраженности уязвимостей пользователя, удалось установить связь между психологическими особенностями личности и рядом уязвимостей.

Модель злоумышленника включает информацию об объеме доступных ему ресурсов, известных социо-инженерных атакующих воздействиях, которые он может применить, а также о доступном ему времени. Модель представима в виде:

$$A_i = (R^i;\ \{At_j^i\}_{j=1}^v;\ T^i),$$

где R^i — доступные i-ому злоумышленнику ресурсы, $\{At_j^i\}_{j=1}^v$ — перечень известных i-ому злоумышленнику социо-инженерных атакующих воздействий, а T^i — время доступное i-ому злоумышленнику для совершения социо-инженерных атакующих воздействий. Социо-инженерные атаки злоумышленника бывают двух типов. Первый тип — атаки манипуляционного характера. В такого рода атаках злоумышленник использует свою квалификацию, то есть известные ему социо-инженерные атакующие воздействия. Примером такого воздействия может быть такая ситуация. В 2011 году в г. Москве в дополнительный офис ОАО «Сбербанк России» зашел представительного вида мужчина. На лацкане его пиджака скромно располагался флажок депутата Государственной думы. Он выбрал одну из сотрудниц, обратился к ней и назвался «Милославским Олегом Богдановичем». Посетитель сообщил, что он является VIP-клиентом банка и назвал номер своего личного счета. Кассир-операционист (работавшая в банке второй день) вошла в систему, перепроверила номер счета «Милославского» и потеряла дар речи от того, что общается с очень богатым и влиятельным человеком. Четко фиксируя реакцию собеседницы, «Милославский» сообщил, что хочет перевести 50 000 000 рублей на другой свой счет, недавно открытый в ОАО «ВТБ-24». При этом мошенник не предъявил никаких документов. Операционист доложила заместителю руководителя дополнительного офиса, та также ни в чем не усомнилась и дала добро на оформление операции [10]. Таким образом, злоумышленник путем формирования реакции сотрудников смог успешно выдать себя за владельца счета и совершить противозаконные действия, не встречая препятствия со стороны сотрудников банка, манипулятивным путем заставив их действовать в обход должных инструкций.

Вторым типом атаки выступают атаки с осуществлением какого-либо вида услуг в замен на доступ к конфиденциальным данным, то есть атаки компенсационного характера. Услуги в определенном смысле конвертируемы

в денежном эквиваленте, поэтому ресурсами злоумышленника в данном случае выступают денежные средства. Необходимо отметить, что в рассматриваемой парадигме ресурсы затрачиваются на второй тип социо-инженерных атак, в отличии от первого типа, на который ресурсы не затрачиваются.

Модель программно-технических устройства имеет в своем составе набор программных приложений, установленных на данном устройстве, связи между устройствами информационной системы, а также набор критичных документов, которые хранятся на данном устройстве. Таким образом, модель имеет вид:

$$CM_i = (\{Apps_j^i\}_{j=1}^m;\ \{L_j^i\}_{j=1}^k),$$

где $\{Apps_j^i\}_{j=1}^m$ — набор программных приложений, установленных на данном устройстве, $\{L_j^i\}_{j=1}^k$ — набор связей данного устройства с другими устройствами информационной системы.

Модель критичных документов представима в виде урона, который может быть нанесен компании в случае нарушения конфиденциальности данного критичного документа. Формальное выражение данной модели представимо в виде:

$$CD_i = (Dm^i;\ \{H_j^i\}_{j=1}^k),$$

где Dm^i — урон, который может быть нанесен компании в случае получения несанкционированного доступа к конфиденциальным данным.

Рассматривая социо-инженерные атаки злоумышленника следует подчеркнуть, что целью таких воздействий является достижение злоумышленника критичных документов, хранящихся в информационной системе. Успех атаки, которая заключается в достижении критичных документов, разбивается на несколько шагов. Элементарным шагом такой атаки является социо-инженерное атакующее воздействие злоумышленника на пользователя информационной системы. Поэтому рассмотрим элементарное событие «успех социо-инженерного атакующего воздействия злоумышленника». Вероятность такого события для пользователя i описывается формулой

$$P(R;\ At_i;\ T \mid V_{i1}, ..., V_{ik}),$$

то есть успех социо-инженерного атакующего воздействия злоумышленника на пользователя, обладающего определенными уязвимостями, при наличии у злоумышленника определенного количества ресурсов, запаса времени и знаний о какой-то совокупности элементарных социо-инженерных атакующих воздействий.

В случае если рассматривается ситуация, когда одно социо-инженерное атакующее воздействие злоумышленника влияет на одну уязвимость пользователя, то тогда вероятность успеха социо-инженерного атакующего воздействия злоумышленника представима в виде

$$P_{suc} = \frac{V_i(D)}{V_{i\max}}.$$

При рассмотрении социо-инженерных атак второго вида, то есть компенсационных атак, злоумышленником, необходимо построить показатель ресурсопотребления подобной атаки. Для моделирования такого показателя представляется целесообразным привлечь теорию математического моделирования коррупции.

Стоит также отметить, что существуют различные вид благ, которые может оказывать злоумышленник. То есть существуют социо-инженерные атаки требующие ряд ресурсов еще до начала атаки, и злоумышленник теряет ресурсы вне зависимости от итога проведенной атаки. Также существуют атаки, требующие от злоумышленника передачи ресурсов после завершения успешной социо-инженерной атаки.

При рассмотрении ситуаций, когда одно социо-инженерное атакующее воздействие злоумышленника влияет на совокупность уязвимостей пользователя, вероятность успеха социо-инженерного атакующего воздействия злоумышленника представима в виде

$$P_{suc} = 1 - \prod_{j=1}^{l} \left(1 - \frac{V_{ij}(D)}{V_{ij\max}}\right).$$

В докладе рассмотрены модели комплекса «информационная система—критичные документы—персонал—злоумышленник», представленные с помощью вероятностно-реляционного подхода.

Модель распространения контента

В случае рассмотрения проблемы распространения контента среди пользователей социальных сетей представляется целесообразным использовать представленную выше модели социо-инженерных атк злоумышленника на пользователей информационных систем. В данном случае в качестве социо-инженерного атакующего воздействия злоумышленника рассматривается факт знакомства пользователя социальной сети с контентом. В качестве социо-инженерной атаки злоумышленника рассматривается способ преподнесения вредоностного контента пользователям социальных сетей. В качестве профиля уязвимостей пользователя рассматривается психологический профиль пользователя, с помощью которого можно построить вероятностные оценки возможности пользователя разместить просмотренный контент у себя на странице. В таком случае, при рассмотрении графа социальных связей пользователя, в данный граф необходимо включать не только «друзей» пользователя по социальной сети, но и «подписчиков» данного пользователя,

в силу того, что в новостных лентах пользователей данных двух групп отображается контент, размещенный пользователем на своей странице. Таким образом, пользователи из данных двух групп также попадают в группу риска возможных распространителей вредоносного контента. Злоумышленником считается пользователь, впервые разместивший угрозообразующий контент в социальной сети. Модель злоумышленника, как и модель пользователя, содержит профиль психологических уязвимостей пользователя.

Заключение

В работе предложены модели комплекса «критичные документы — информационная система — персонал — злоумышленник», а также применение данной модели к проблеме распространения контента среди пользователей социальных сетей.

Литература

1. *Ванюшичева О. Ю., Тулупьева Т. В., Пащенко А. Е., Тулупьев А. Л., Азаров А. А*. Количественные измерения поведенческих проявлений уязвимостей пользователя, ассоциированных с социоинженерными атаками // Труды СПИИРАН. 2011. Вып. 19. С. 34–47.
2. *Азаров А.* А. Анализ защищенности информацион-ных систем от социо-инженерных атак рекомпенсационного типа в отношении пользователей // VII Санкт-Петербургская межрегиональная конференция «Информационная безопасность регионов России (ИБРР-2011)» (Санкт-Петербург, 26–28 октября 2011 г.) Материалы конференции. СПб.: СПОИСУ, 2011. С. 160.
3. *Котенко И. В., Юсупов Р.* М. Перспективные направления исследований в области компьютерной безопасности. Защита информации. Инсайд. 2006. № 2. С. 46.
4. *Степашкин М. В*. Модели и методика анализа защищенности компьютерных сетей на основе построения деревьев атак : Дис. канд. техн. наук: СПб.: СПИИРАН, 2002. 196 с.
5. *Тулупьев А. Л., Азаров А. А., Тулупьева Т. В., Пащенко А. Е., Степашкин М. В*. Социально-психологические факторы, влияющие на степень уязвимости пользователей автоматизированных информационных систем с точки зрения социоинженерных атак // Труды СПИИРАН. 2010. Вып. 1 (12). С. 200–214.
6. *Тулупьев А. Л., Азаров А. А., Пащенко А.* Е. Информационные модели компонент комплекса «Информационная система — персонал», находящегося под угрозой социоинженерных атак // Труды СПИИРАН. 2010. Вып. 3 (14). С. 50–57.
7. *Тулупьев А. Л., Азаров А. А., Тулупьева Т. В., Пащенко А. Е*. Визуальный инструментарий для построения информационных моделей комплекса «информационная система — персонал», использующихся в имитации социоинженерных атак // Труды СПИИРАН. 2010. Вып. 4 (15). С. 231–245.
8. *Тулупьева Т. В., Тулупьев А. Л., Азаров А. А., Пащенко А. Е*. Психологическая защита как фактор уязвимости пользователя в контексте социоинженерных атак // Труды СПИИРАН. 2011. Вып. 18. С. 74–92.

9. *Юсупов Р., Пальчун Б.* П. Безопасность компьютерной инфосферы систем критических приложений. Вооружение. Политика. Конверсия. 2003. № 2. С. 52.
10. Украдено 50 000 000 рублей. URL: http://lenta.ru/news/2013/04/07/dzirkaln/ (Дата обращения: 03.03.2014)
11. *Шейнов В. П.* Искусство управлять людьми // Харвест. Минск, 2004. 512 с.

КОПУЛЬНЫЙ ПОДХОД К ОЦЕНКЕ ОТНОСИТЕЛЬНЫХ ПОКАЗАТЕЛЕЙ РИСКА[1]

Д. В. Степанов
СПИИРАН
E-mail: denis_v_stepanov@hotmail.com

Аннотация. В настоящей работе рассматриваются способы применения аппарата копул для описания зависимости между процессами риска в модели расчета относительных оценок частот событий, базирующейся на использовании байесовских сетей доверия.

Введение

Оценка численных показателей динамики риска передачи инфекционных заболеваний относится к числу практически важных и интенсивно исследуемых задач эпидемиологии [1]. Примером такого численного показателя является отношение рисков — RR (relative risk).

Алгоритмический аппарат теории байесовских сетей доверия является мощным инструментом для решения указанной задачи, требующим на предварительном этапе получить оценку условных распределений для случайных процессов, ассоциированных с риском [2].

Обычно для упрощения вычислений предполагают независимость интенсивностей рискового поведения в исследуемых группах [1]. В случае отклонения гипотезы о независимости подобный подход может привести к получению смещенных оценок относительного риска.

Удобным инструментом для описания и моделирования зависимых случайных величин являются копулы, позволяющие исследовать структуру зависимости отдельно от граничных распределений самих случайных величин [3]. Использование копул в моделях расчета относительных оценок риска, базирующихся на использовании байесовских сетей доверия, требует учета ряда ограничений.

Прежде всего, для описания байесовской сети доверия рассматриваемые случайные величины должны быть дискретизованы. Необходимость дискретизации вызвана тем, что информация в задачах социального компьютинга часто носит гранулярный характер а также соображениями удобства при работе с таблицами условных вероятностей [4]. Таким образом, при описании байесовской сети копулы также должны быть дискретизованы.

Кроме того возникает задача оценки копул по выборке при отсутствии информации о совместном распределении рассматриваемых величин.

Настоящая работа посвящена описанию подходов к использованию копул в задачах оценки риска на основе байесовских сетей доверия.

[1] Работа частично поддержана грантами РФФИ № 12-01-00945-а и 14-01-00580.

Распределение величины RR

Величина RR представляет собой отношение вероятностей некоторого события в двух различных популяциях (или в одной популяции до и после внедрения превентивных мер, направленных на снижение риска). Числитель и знаменатель RR также можно интерпретировать как некоторые случайные величины, распределения которых зависят от интенсивностей процессов риска, также описываемых вероятностными распределениями [1, 2, 4]. Например, рассмотрим кумулятивный риск r_n заражения за n эпизодов рискового поведения при вероятности p заразиться за один эпизод:

$$r_n = 1 - (1 - p)^n.$$

Предполагая пуассоновскую модель процесса риска, получим выражение для кумулятивного риска на временном интервале длительности T

$$r = 1 - \sum_{n=0}^{\infty} (1 - p)^n \cdot e^{-\lambda T} \frac{(\lambda T)^n}{n!} = 1 - e^{-\lambda T p}.$$

Будем рассматривать числитель и знаменатель RR как абсолютно непрерывные неотрицательные случайные величины X и Y, имеющие функции распределения F и G (а также плотности f и g) соответственно. Пусть $H(x, y)$ — совместная функция распределения вектора (X, Y) (через h обозначим плотность). Зададимся вопросом о распределении величины $Z = Y/X$. Пусть $D(z) = \{(x, y) : y/x < z\}$, тогда функция распределения случайной величины Z равна:

$$K(z) = \int_0^{+\infty} \int_0^{zx} h(x, y)\, dy dx. \tag{1}$$

Дифференцируя выражение (1) по z, получаем плотность случайной величины Z:

$$k(z) = \int_0^{+\infty} x h(x, zx)\, dx. \tag{2}$$

В предположении о независимости случайных величин X и Y, выражение (2) преобразуется к виду

$$k_0(z) = \int_0^{+\infty} x f(x)\, g(zx)\, dx$$

и может быть использовано для оценки вероятностей, связанных с величиной Z (описывающей поведение RR).

В ряде случаев предположение о независимости рассматриваемых случайных величин вряд ли удастся обосновать (например при исследовании показателей риска в одной популяции до и после превентивных мероприя-

тий). Таким образом, возникает необходимость работы с выражением (2) без предположений о независимости случайных величин X и Y.

Дадим краткое описание инструмента, позволяющего выделить из совместного распределения граничные распределения и структуру зависимости.

Определение 1. Двумерная копула $C(u, v)$ — совместная функция распределения, заданная на $S=[0, 1]\times[0, 1]$, граничные распределения которой равномерные.

Согласно теореме Склара [3], для каждой функции совместного распределения $H(x, y)$ верно представление $H(x, y)=C(F(x), G(y))$ для всех $(x, y)\in\mathbb{R}^2$. Для непрерывных функций распределения такое представление единственно и оно может быть записано в эквивалентном виде:

$$C(u, v)=H\left(F^{-1}(u),\ G^{-1}(v)\right),\ \forall\ (u, v)\in S.$$

Для плотности совместного распределения h верно представление

$$h(x, y)=c\left(F(x), G(y)\right)f(x)g(y),$$

где $c(u, v)=\dfrac{\partial^2}{\partial u\partial v}C(u, v)$ — плотность копулы.

Таким образом, выражение (2) для плотности RR преобразуется к виду:

$$k(z)=\int_0^{+\infty} xf(x)g(zx)c\left(F(x), G(zx)\right)dx. \qquad (3)$$

В следующих разделах рассмотрим, как повлияет на выражение (3) необходимость дискретизации случайных величин при переходе к байесовским сетям доверия и как можно вычислить значения $c(F(x), G(y))$.

Дискретизация случайных величин и аппроксимация копул

Необходимость дискретизации случайных величин при задании байесовской сети доверия подводит к идее использования конечномерных аппроксимаций копул с помощью дважды стохастических матриц [5, 6, 7].

Определение 2. Система функций $\varphi=\{\varphi_i\}_{i=1}^m\in\mathbb{L}_\infty([0, 1])$ называется разбиением единицы, если она удовлетворяет следующим условиям:

$$\varphi_i(x)\geq 0;$$

$$\int_0^1\varphi_i(x)\,dx=\frac{1}{m};$$

$$\sum_{i=1}^m\varphi_i(x)=1,\ \forall x\in[0, 1]\ .$$

Обозначим $\Phi = (\Phi_1, ..., \Phi_m)$, $\Phi_i(u) = \int_0^u \varphi_i(x)\,dx$, $u \in [0, 1]$.

Пример разбиения единицы — набор индикаторных функций $\chi_{i,m}$ равновеликих интервалов $\left[\frac{i-1}{m}, \frac{i}{m}\right]$, $i = 1, ..., m$.

Определение 3. Матрица $\Delta = \{\Delta_{ij}\} \in \mathbb{R}^{m\times m}$ – дважды стохастическая, если $\sum_{i=1}^{m}\Delta_{ij} = \sum_{j=1}^{m}\Delta_{ij} = 1$. Множество дважды стохастических матриц размерности $m \times m$ обозначим $\mathfrak{D}_m$.

Следующее утверждение описывает правила конструирования копул с помощью дважды стохастических матриц [6].

Утверждение 1. Пусть $\Delta \in \mathfrak{D}_m$ и $\varphi = \{\varphi_i\}_{i=1}^m$ – разбиение единицы, тогда $C_\Delta(u, v) = m\Phi^T(u)\Delta\Phi(v)$ — абсолютно непрерывная копула.

Разбиение единицы может быть естественным образом связано с дискретизацией случайных величин в байесовской сети доверия. Действительно, рассмотрим -точечную дискретизацию $\{x_i\}_{i=1}^m$ случайной величины X с функцией распределения F, такую что $F(x_i) - F(x_{i-1}) = \frac{1}{m}$, $\forall i = 1, ..., m$. Аналогично определим дискретизацию для случайной величины $Y \sim G$. Пусть $C(F(x), G(y))$ — копула, отвечающая совместному распределению вектора $(X, Y) \sim H$. Рассмотрим сужение копулы $C(u, v)$, $u, v \in [0, 1]$ на равномерную решетку $\mathfrak{T}_m = \left\{\left(\frac{i}{m}, \frac{j}{m}\right) : i, j = 1, ..., m\right\}$. Введем обозначение для меры множества $\left[\frac{i-1}{n}, \frac{i}{n}\right] \times \left[\frac{j-1}{n}, \frac{j}{n}\right]$, индуцируемой копулой C:

$$\Delta_{i,j}(\tilde{N}) \triangleq C\left(\frac{i}{n}, \frac{j}{n}\right) - C\left(\frac{i-1}{n}, \frac{j}{n}\right) - C\left(\frac{i}{n}, \frac{j-1}{n}\right) + C\left(\frac{i-1}{n}, \frac{j-1}{n}\right).$$

Обозначим матрицу $\Delta(C) = \{\Delta_{i,j}(C)\}_{i,j=1}^m$.

Верно следующее утверждение [5, 6].

Утверждение 2. Матрица $m\Delta(C)$ — дважды стохастическая.

Из Утверждений 1 и 2 получаем следующую аппроксимацию для функции копулы:

$$\mathfrak{C}_m(C)(u, v) = m^2\Phi^T(u)\Delta(C)\Phi(v). \qquad (4)$$

В [6] получены верхние оценки для $\mathfrak{C}_m(C) - C_\infty$. В частности, для случая задания разбиения единицы с помощью индикаторных функций $\chi_{i,m}$ верна оценка $\mathfrak{C}_m(C) - C_\infty \le \frac{2}{m}$.

Таким образом, при наличии экспертно заданных совместной и граничных функций распределения, может быть определена дискретная аппроксимация для копулы, связанная с принятой при описании байесовской сети доверия дискретизацией случайных величин.

Аппроксимация копул по выборке

Рассмотрим ситуацию, в которой распределения случайных величин не задаются экспертом, а оцениваются по выборке.

Определение 3. Пусть $\{(x_k, y_k)\}_{k=1}^n$ — выборка из непрерывного двумерного распределения H, с граничными распределениями F и G соответственно. Эмпирическая копула — функция, задаваемая выражением

$$C_n\left(\frac{i}{n}, \frac{j}{n}\right) = H_n\left(F_n^{-1}\left(\frac{i}{n}\right), G_n^{-1}\left(\frac{j}{n}\right)\right) = \frac{1}{n}\sum_{k=1}^{n} 1_{\{\mathrm{rank}(x_k)\le i,\ \mathrm{rank}(y_k)\le j\}},$$

где H_n, F_n, G_n — эмпирические функции распределения, $1_{\{\bullet\}}$ — индикаторная функция, а $\mathrm{rank}(x_k)$ — номер наблюдения x_k в вариационном ряду. Плотность эмпирической копулы задается выражением:

$$c_n\left(\frac{i}{n}, \frac{j}{n}\right) = \Delta_{i,j}(C_n).$$

Обратим внимание на то, что функция $C_n(u, v)$, вообще говоря, не является копулой [6].

Пусть $m = n$ и разбиение единицы задано с помощью индикаторных функций $\chi_{i,m}$, тогда, согласно Лемме 3 из [6] выборочная оценка копулы задается выражением

$$\mathfrak{C}_n(C_n)(u, v) = n^2 \Phi^T(u) \Delta(C_n) \Phi(v). \qquad (5)$$

Неудобство выражения (5) состоит в том, что число m элементов разбиения единицы (которое должно быть желательно малым) было приравнено к числу n элементов в выборке (которое может быть большим).

Возможное решение данной проблемы состоит в использовании на первом шаге формулы (5) для оценки функции копулы по выборке, с последующей подстановкой $C = \mathfrak{C}_n(C_n)$ в формулу (4).

Заключение

В статье рассмотрены подходы к использованию копул при описании байесовской сети доверия. Описаны способы задания дискретных копул, отвечающих дискретизации случайных величин, принятой при задании байесовской сети доверия в случае экспертно заданных распределений и распределений оцененных по выборке.

Литература

1. *Пащенко А. Е.* Применение байесовских сетей доверия для расчета относительных оценок показателей процессов, ассоциированных с риском, в условиях информационного дефицита // Труды СПИИРАН. 2013. Вып. 8 (31). С. 95–120.
2. *Суворова А. В.* Представление пуассоновской модели социально-значимого поведения в виде байесовской сети доверия // Современные проблемы математики. Тезисы Международной (44-я Всероссийская) молодежной школы-конференции. Екатеринбург: Институт математики и механики УрО РАН, 2013. С. 333–335.
3. *Nelsen R. B.* An Introduction to Copulas. New York, Springer, 2006. 270 pp.
4. *Суворова А. В., Мусина В. Ф., Тулупьева Т. В., Тулупьев А. Л., Красносельских Т. В., Фильченков А. А., Азаров А. А., Абдала Н.* Автоматизированный инструментарий для опроса респондентов об эпизодах рискованного поведения: первичный анализ результатов применения // Труды СПИИРАН. 2013. Вып. 3 (26). С. 175–193.
5. *Durrleman V., Nikeghbali A., Roncalli T.* Copulas approximation and new families. Technical report. 2000. [http://dx.doi.org/10.2139/ssrn.1032547].
6. *Guillotte S., Perron F.* Bayesian estimation of bivariate copula using the Jeffreys prior / Bernoulli 18(2). 2012. pp. 496–519.
7. *Amblard C., Girard S., Menneteau L.* Bivariate copulas defined from matrices / arXiv:1310.5560v1 [math.ST] 21 Oct 2013.

ПРИМЕНЕНИЕ ЭКСПЕРТНОЙ СИСТЕМЫ НА ОСНОВЕ НЕЙРОННОЙ СЕТИ ДЛЯ ПРОГНОЗИРОВАНИЯ ПОТРЕБЛЕНИЯ ПРИРОДНОГО ГАЗА

К. Д. Коромыслов

магистрант кафедры «Компьютерные системы автоматизации производства» МГТУ им. Н. Э. Баумана

E-mail: koromyslovkd@gmail.com

Аннотация. Развитие систем искусственного интеллекта (ИИ) и бурное развитие вычислительной техники позволяет внедрять и успешно использовать в промышленности различные виды систем, такие как, например, системы имитационного моделирования, экспертные системы, нечеткие системы, искусственные нейронные сети. В работе рассмотрен вариант использования в газовой промышленности гибридных интеллектуальных систем — экспертных системы на основе нейронной сети для прогнозирования потребления природного газа.

Введение

Природный газ — главный источник энергии с растущей популярностью, главным образом благодаря его благоприятным экологическим свойствам. Для решения проблемы потери давления, вызванной расширением природного газа, потерями на трение, изменением температуры используют компрессорные станции (КС). Внедрение периодических КС вдоль сети трубопровода является одним из решений, используемых для удержания заданного давления. Турбинный привод компрессора, работающий на топливном газе, является одним из наиболее распространенных решений для магистралей природного газа. Для перемещения газа газовые турбины вращают центробежные компрессоры и сжимают газ в сотни раз относительно атмосферного давления.

Магистральная транспортная сеть — это комплексная система, которая может состоять из сотен узлов КС, вследствие чего возникает потребность в контроле и управлении большим числом сопутствующего оборудования. Оптимизация режима работы КС — достаточно сложная задача. Сложность усугубляется нелинейностью профиля потребления газа. Снижение энергопотребления для поддержания работы трубопровода оказывает положительное экономическое и экологическое влияние. Более энергоэффективный режим работы КС приводит к меньшим выбросам, и, как следствие, уменьшению парникового эффекта. КС производят CO2 главным образом из-за необходимости подводить энергию к турбине, за счет которой работает компрессор.

Для каждого мегаватта полученной энергии производится около 5 тонн CO2 каждый год [1].

В данной работе автор стремился спроектировать оптимизированную систему управления газотранспортной сетью (ГТС), позволяющую выбрать такой режим работы компрессорного цеха, который бы использовал минимальное количество энергии (например, топливо, мощность), обеспечивая при этом работу газотранспортного оборудования. Система управления КС состоит из двух частей. Первая часть — прогнозирование потребления газа на основе искусственной нейронной сети, учитывающей основные параметры для сети трубопроводов и профиля потребления газа. Вторая часть, непосредственно оптимизационный алгоритм, который стремится поставить до конечных потребителей достаточный поток газа с минимальными потерями на транспортировку.

В данной работе представлены исследования автора на тему разработки экспертной системы для оптимизации работы КС магистральной сети транспортировки природного газа.

1. Постановка задачи

Для оптимизации управления ГТС необходимо оценить режимы работы компрессорной станции в зависимости от изменяющихся входных параметров. Чтобы достигнуть этой цели, необходимо иметь достаточно знаний о будущих условиях работы сети газопровода. Таким образом, прогноз потребления газа должен быть подан для системы оптимизации как известный входной параметр.

В современной ГТС диспетчер ответственен за оптимизацию планирования работы компрессора и сам решает, какой компрессор должен быть включен или же выключен для того, чтобы изменить давление в трубопроводе, сохраняя оптимальное рабочее давление в системе. Неопытный диспетчер может сделать неправильный выбор из-за недостаточного опыта. Как минимум шесть месяцев требуется для обучения оператора, но и это не гарантирует выбор наиболее энергоэффективного режима работы, потому что существует задержка между показателями сети, выводящимися на пульте управления диспетчера и реальными эксплуатационными условиями на КС. Удаленные сигналы на включение/выключение компрессора являются небезопасными и могут привести к созданию аварийных ситуаций.

Сложность сети трубопроводов и существование многих переменных параметров с нелинейными взаимосвязями существенно осложняют решение задачи. Использование рекуррентных нейронных сетей в качестве системы прогнозирования в нелинейных динамических системах в последнее время активно набирает обороты. В данной статье модели на основе нейронной сети рассматриваются с точки зрения наличия множество взаимосвязанных нелинейных объектов. Акцент сделан на быстроту, стабильность и пространствен-

но-временное поведение рекуррентной архитектуры. Последние достижения в эволюционных алгоритмах позволили начать применять их для решения подобного рода задач. Одним из наиболее известных и популярных генетических алгоритмов (ГА) является оптимизационный алгоритм на основе метаэвристик. Среди преимуществ ГА можно выделить их адаптивность, гибкость и устойчивость. Рассмотрим подробнее архитектуру предлагаемой системы.

2. Предложенное решение

Цели оптимизации работы КС — надежный и достаточный (в необходимых объемах) подвод газа конечным потребителям и минимизация энергии, которую будет потреблять КС для транспортировки газа.

На рис. 1 представлена структура системы, которая включает две главных части: модуль прогнозирования потребления газа и непосредственно сам алгоритм оптимизации работы КС.

Для создания системы прогнозирования была использована информация о потребляемом газе от измерительной станции рядом с г. Калуга за 2010–2011 гг.

В зависимости от особенностей задачи и доступных данных, использование рекуррентной нейронной сети (РНС) может быть наиболее подходящим методом прогноза. РНС существенно отличаются от сетей прямого распространения в том смысле, что они работают, в дополнение ко входным данным, значениям, уже полученных в результате работы сети. Общая структура сети изображена на рис. 2.

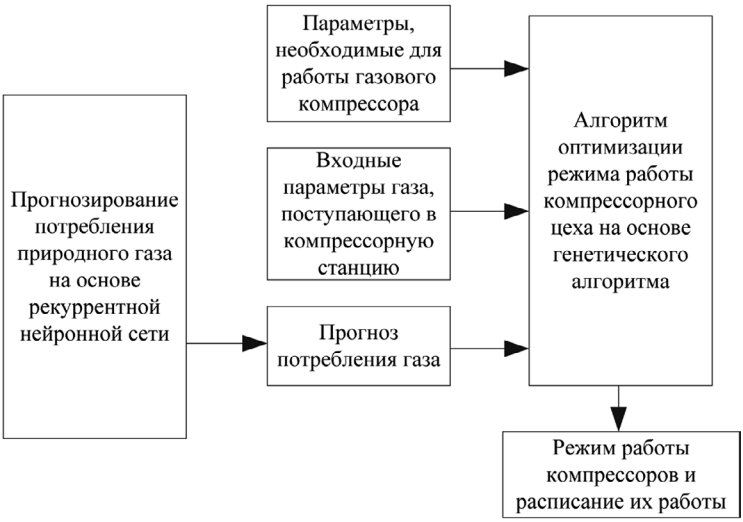


Рис. 1. Структура системы оптимизации управления КС

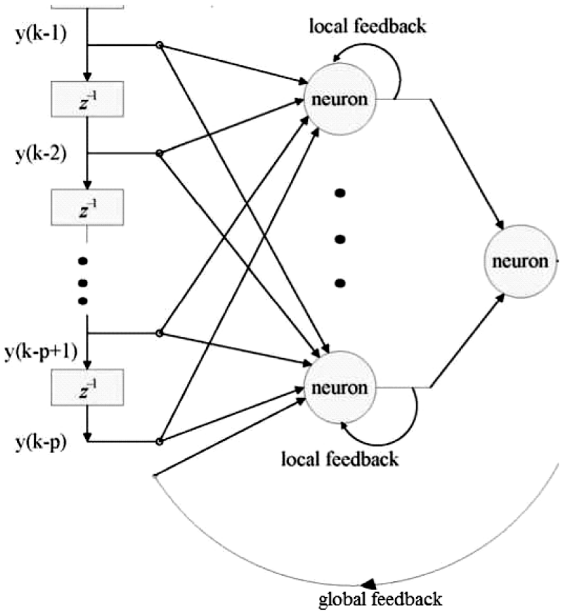


Рис. 2. Структура рекуррентной нейронной сети
с обратным распространением (local and global feedback)

В свое время было предложено много вариантов РНС, в том числе сети Элмана и Джордана. В данной работе была использована сеть Джордана, за счет более высоких показателей производительности. Такая сеть является компромиссом между простотой нейронной сети прямого распространения и традиционной полностью рекуррентной сетью.

Работа нейронов состояния выражается следующим образом:

$$s_i(t) = \lambda s_i(t-1) + y_i(t-1),$$

где λ — положительный коэффициент рекуррентности. Обычно $\lambda < 1$ и его значение определяет насколько будет учтено влияние выходного значения, вновь запускаемого в сеть. После этого, выходные данные из нейронов контекстного уровня поступают на входы нейронов скрытого уровня. Таким образом, нейроны скрытого уровня получают на свои входы вектор значений, состоящий из входных значений и значений, полученных от предыдущего вычисления сети. Затем происходит модификация алгоритма обучения с обратным распространением [2]. Число входных нейронов зависит от числа параметров, которые необходимо определить, в то время как число скрытых нейронов зависит от структурной сложности задачи и должно быть определено экспериментально.

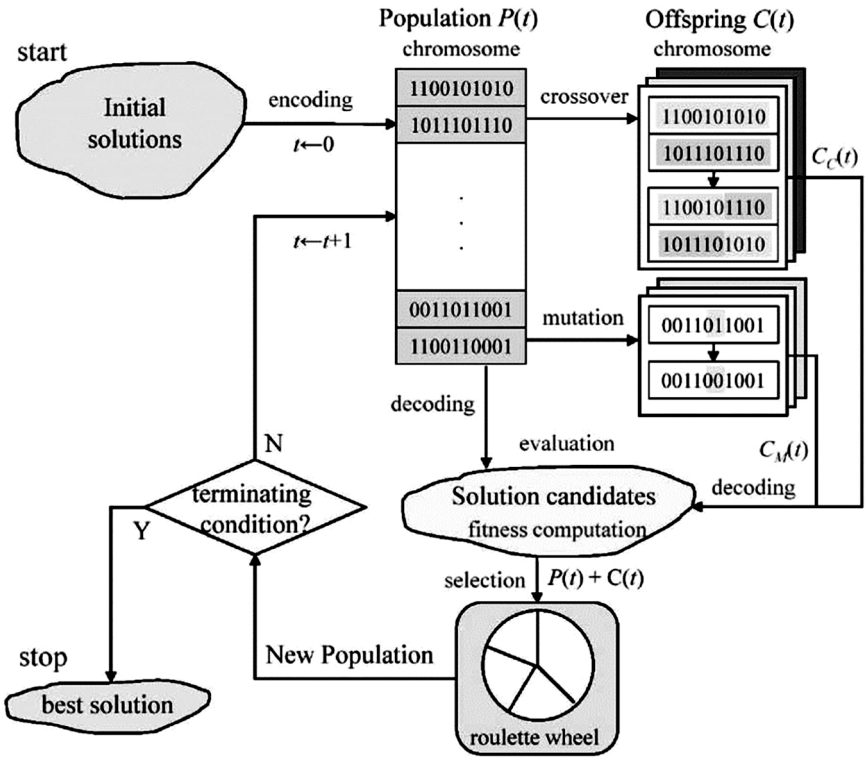


Рис. 3. Общая структура генетического алгоритма

Простой ГА состоит из трех операторов: отбор, кроссовер и мутация. Отбор — процесс, в котором единичные агенты отбираются в отдельный пул согласно их производительности. После отбора кроссовер продолжает создавать новые поколения агентов. Агенты из пула высокопроизводительных особей скрещиваются между собой в произвольном порядке. Затем проводится мутация чтобы предотвратить потерю хороших генов в результате применения кроссовера [3].

В работе процесс отбора осуществлялся с помощью метода рулетки, где агенты с более высокой пригодностью занимают больше ячеек (рис. 3). Были применены методы единичного кроссовера и однобитовой мутации. Для каждой пары агентов положение целого числа k выбиралось равномерно наугад, затем создавалось два новых агента, путем обмена всех отличительных признаков после положения k. Мутация была воплощена в виде случайного изменения величины одного из битов агента. Такие решения были приняты на основе наиболее распространенных вариантов применения ГА.

Обсуждение различного возможного выбора генетических операторов было проведено в работе [4]. В методе турнирного отбора сравниваются значения пригодности, и затем отбираются лучшие агенты. В отличие от метода рулетки, эта схема не отбрасывает хорошие гены, которые скрыты в агентах с плохой пригодностью. Характер области оптимизации обычно является тем фактором, который определяет, какой алгоритм отбора даст лучшие рабочие характеристики.

3. Результаты

Для обучения рекуррентной нейронной сети были использованы данные потребления газа на Калужской КС за 2010–2011 гг. В качестве передаточной функции была взята сигмоидальная функция с начальным шагом в 0.2 и максимумом в 40 со значением степени 0.3. Обучение сети происходило на разных наборах данных для различных временных периодов в рамках 2010–2011 гг. Было отмечено, что параметры, зависящие от температуры, определяют до 80% точности предсказания. Более того, точность предсказания увеличивается при учете температурных данных прошлых дней. Небольшое улучшение точности также возможно при учете дня недели и даты. При расчете показателей для далекого будущего точность данных увеличивается.

ГА использовался при фиксированной стоимости топлива и запуска, остальные показатели задавались переменными. Стоит отметить, что оба компрессора на станции Калужская имеют одинаковые номинальные мощности. Размер популяции варьировался от 35 до 65 потомков, количество поколений — от 500 до 10 000, показатель скрещивания — 0.5, 0.7, 0.9, показатель мутации — 0.0005, 0.004, 0.03.

Было замечено, что при малых показателях объема перекачиваемого газа снижается время расчетов, а при увеличении потока оно растет. Для оптимизации функции приспособляемости следует крайне осторожно определять показатель мутации. Результаты применения ГА для расчета оптимального расписания работы компрессоров были рассмотрены с точки зрения условий работы компрессоров. Было замечено, что при низком потреблении газа функция достигала локального минимума и адиабатический максимум КПД не мог быть достигнут.

Заключение

В данной работе автор рассмотрел непростую задачу оптимизации работы компрессорной станции. Для обучения ИНС и создания ГА были использованы экспериментальные данные о работе ГТС в Калужской области. Использования рекуррентной ИНС для прогнозирования объема спроса на газ позволило получить точные данные для оптимизации работы компрессоров за счет ГА.

По результатам расчетов были определены локальные минимумы эффективности. При подробном рассмотрении результатов можно сделать вывод, что максимизации КПД невозможна при низких объемах поставок газа. Таким образом, для максимизации эффективности работы компрессорных станций следует при низких объемах потребления устанавливать на станции половину компрессоров с номинальной мощностью, в два раза меньшей, чем у основных компрессоров, с возможностью параллельного запуска любого набора компрессоров. Именно так при низком объеме потребления может быть достигнута максимальная эффективность работы компрессора.

Литература

1. *Bott R*. Our Petroleum Challenge: Exploring Canada's Oil and Gas Industry. Petroleum Communication Foundation, 1999. 101 p.
2. *Rumlhart D. E.*, et al. Learning Representations by Back-Propagating Errors // Nature. 1986. Vol. 323. P. 533–536.
3. *Goldberg D. E*. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Publishing Company Inc., 1989. 412 p.
4. *Gen M., Cheng R., Lin L*. Network models and Optimization: Multiobjective Genetic Algorithm Approach. Springer Verlag, 2008. 692 p.

АВТОМАТИЗАЦИЯ АНАЛИЗА ПОПУЛЯРНОСТИ ТЕХНОЛОГИЧЕСКИХ ОБЛАСТЕЙ В КОРПУСЕ ТЕКСТОВ РУССКОЯЗЫЧНЫХ ЭЛЕКТРОННЫХ МЕДИА НА ОСНОВЕ ДАННЫХ ВИКИПЕДИИ

А. М. Алексеев
студент СПбГУ
E-mail: anton.m.alexeyev@math.spbu.ru

Аннотация. В докладе рассматривается подход к автоматизации подсчёта, вместе с какими технологическими областями во влиятельных электронных русскоязычных СМИ и в какой момент упоминались наименования различных организаций.

Дано описание использованных методов извлечения наименований организаций и технологических новостей.

Введение

С формированием Web 2.0 и, следовательно, быстрым ростом объёма данных, публикуемых в WWW увеличивающимся числом пользователей, на свет появилось Web Science — научное направление, предметом которого является WWW, «ее социальная и технологическая сущность» [2]. К нему относят междисциплинарные исследования «виртуального мира», в том числе позволяющие делать выводы о состоянии «реального мира».

Большие объёмы потенциально ценной информации в сети Интернет — это неструктурированные либо слабоструктурированные данные, в том числе тексты на естественном языке. Число примеров огромно: новостные порталы, блоги, архивы публикаций и др.

В последние годы преимуществами таких данных из WWW пользуются маркетологи. Например, известны работы, посвящённые предсказанию тенденций на рынках с использованием данных поисковых сервисов [8] и социальных сетей [3]. Так как объём данных велик, требуются средства автоматизации работы аналитика, поэтому существует значительное число инструментов для автоматического анализа медиаприсутствия той или иной компании.

В докладе предлагается подход к автоматизации подсчёта, вместе с какими технологическими областями во влиятельных электронных русскоязычных СМИ и в какой момент упоминались наименования различных организаций.

Извлечение наименований организаций

Извлечение организаций можно отнести к задачам выделения именованных сущностей (named-entity recognition), то есть объектов заданного

типа, имеющих имя, название или идентификатор [1]. К самым эффективным из современных способов поиска именованных сущностей без словарей можно отнести выделение с помощью вероятностных графических моделей и родственных им: вероятностных контекстно-свободных грамматик [4], скрытых марковских моделей [6] и условных случайных полей [5, 7]. Во всех указанных подходах задача сводится к разметке текста, разбитого на токены, некоторыми из ярлыков: «начало сущности», «часть сущности», «конец сущности», «не относится к сущности». Кроме того, решается и задача классификации: определить, к какой категории (например, «человек» или «организация») относится найденная именованная сущность.

Однако в нашем случае можно ограничиться заранее заданным списком наиболее значимых компаний (с различными вариантами написания каждой), что позволит легко приводить найденные организации к некоторой нормальной форме. Для этой цели можно осуществлять поиск упоминаний организаций в данном тексте с использованием соответствующих структур данных: инвертированных индексов, префиксных деревьев и т. д. Ниже дано описание предлагаемого метода.

Пусть дан список значимых организаций (в качестве источников можно использовать, например, habrahabr.ru, crunchbase.org, wikipedia.org).

1. Наименование каждой организации из списка подвергаем токенизации и стеммингу.
2. Конкатенируем цепочку токенов каждой компании через пробел и добавляем получившуюся строку в префиксное дерево (Inverted Radix Tree).
3. Из анализируемого текста с помощью регулярного выражения стираем все «потенциальные false positives», например, URL.
4. Все запятые заменяем на несуществующее слово (например, «symbolnottobesearched»).
5. Производим токенизацию и стемминг анализируемого текста, затем конкатенацию полученной цепочки через пробел.
6. Осуществляем поиск подстрок в строке, полученной на этапе (5), с помощью построенного на этапе (2) префиксного дерева.

Каждая совпавшая подстрока — наименование организации (с точностью до приведения к нормальной форме с помощью списка альтернативных написаний).

Извлечение технологических областей

Задачу выделения ключевых слов и терминов можно по праву считать классической для компьютерной лингвистики. Число используемых на практике методов велико. В нашем распоряжении нет корпуса текстов выбранной тематики с отмеченными ключевыми словами, и число искомых нами технологических областей должно быть ограничено, поэтому использование методов «без учителя» может оказаться неоправданным. В качестве решения

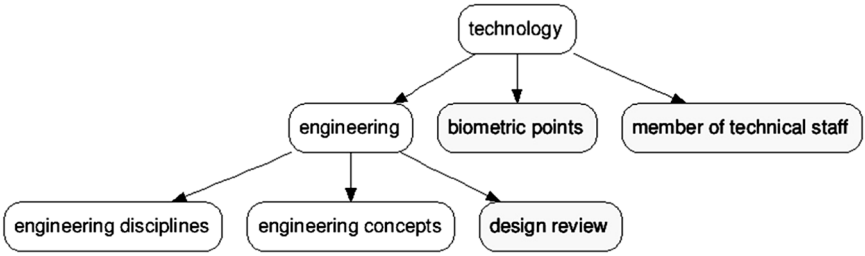


Рис. 1. Подграф «графа категорий»:
светлыми вершинами обозначены категории, тёмными — статьи

можно взять поиск имён объектов из базы знаний. Под базой знаний будем понимать иерархически структурированный набор понятий, который может быть получен из свободно распространяемых материалов Википедии.

Википедия

В Википедии многие статьи отнесены к той или иной «категории». Те и другие образуют «граф категорий» (см. Рис. 1).

С точки зрения теории графов, «граф категорий» — ориентированный граф, в котором вершины делятся на два вышеуказанных типа. Статьи не имеют исходящих рёбер, но, как и категории, могут иметь несколько входящих (см. Рис. 2).

В этом графе есть контуры (см. Рис. 3), в которые, очевидно, входят только вершины-«категории».

Если исключить из орграфа контуры, он будет задавать некоторый частичный порядок на множестве вершин. Разметкой Википедии занимаются люди, поэтому ошибки возможны, но задуманный частичный порядок такой: если есть ребро из одной вершины в другую, то первая вершина описывает более общее понятие, чем вторая (см. Рис. 1). Предками категории или статьи будем

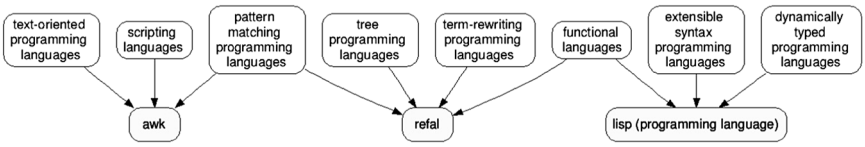


Рис. 2. Пример общих категорий нескольких статей
в «дереве категорий»

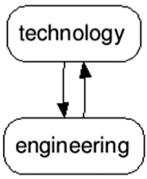


Рис. 3. Пример контура в «дереве категорий» Википедии →

называть категории, у которых есть ребра, ведущие в данную. Потомками категории будем называть категории либо статьи, в которые у данной ведёт ребро.

Метод извлечения технологических областей

В этом разделе дано описание словарного метода извлечения технологических областей с помощью Википедии.

Пусть дан текст на русском языке. Ниже описаны шаги этапа подготовки данных.

1. Строится словарь всех нормализованных с помощью стемминга слов, упоминающихся в заголовках статей и текстовых ссылках Википедии.
2. Наполняется индекс (инвертированный индекс или префиксное дерево) над нормализованными стеммингом заголовками статей русскоязычной Википедии.
3. В индекс добавляются пары вида «заголовок русскоязычной статьи» — «заголовок соответствующей англоязычной статьи» (interwiki links).
4. Осуществляется выборка потомков вики-категории Technology из дерева категорий англоязычной Википедии (с ограничением по «расстоянию» от Technology не более некоторого конечного числа) и построение ассоциативного массива вида «потомок → предок».
5. Целевой русскоязычный текст разбивается на токены, стоп-слова заменяются на заведомо несуществующие слова, все токены, которые, после применения к ним стемминга, не находятся в словаре из (1), заменяются на несуществующее слово.

После этого происходит собственно извлечение технологических областей.

1. Для всех лексем в предобработанном целевом тексте, а также для всех пар, троек и т. д. лексем осуществляется поиск соответствующих русского и английского заголовков статей.
2. Для каждого найденного английского заголовка проверяется его наличие в множестве всех вершин дерева категорий из BFS-дерева категории Technology, ограниченного по высоте; если нет — убираем данный заголовок из рассмотрения, если есть — оставляем и добавляем в список технологических областей некоторое фиксированное число его предков в BFS-дереве вики-категорий, хранящегося в ассоциативном массиве.
3. Из получившегося списка четвёрок вида (phrase,ru_article,en_article) берутся все en_article.

Список английских категорий и будем считать извлечёнными технологическими областями. Пример работы метода приведён на Рис. 4.

Фильтрация по «дереву категорий» позволяет:

1. Разрешить неоднозначность (пример: «Ягуар» как автомобиль и животное «ягуар» могут быть перепутаны; наличие категории «Кошачьи» однозначно определяет целевую статью).

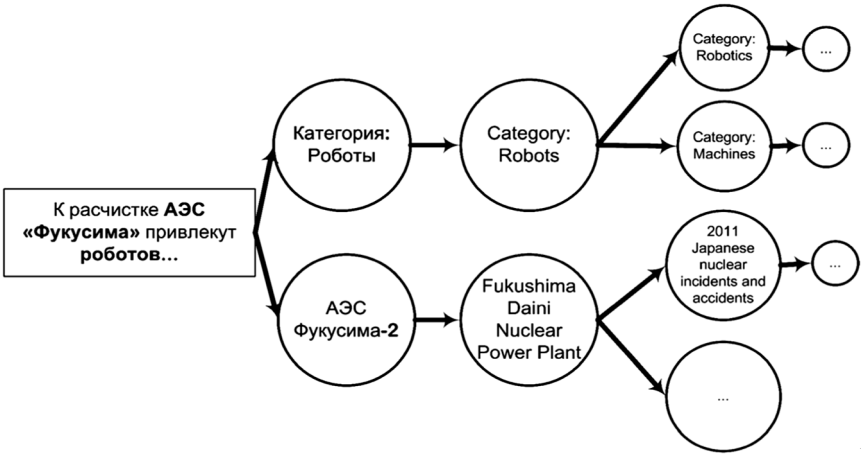


Рис. 4. Метод поиска технологических областей

2. Справиться с ошибками сопоставления набора слов из входного текста с заголовком статьи.
3. Обеспечить, хотя бы отчасти, значимость извлекаемой технологической области, если предположить что более 800 000 переведённых страниц Википедии (из более чем 3 млн. русскоязычных) — наиболее важны как «переведённые в первую очередь».

Дополнение множества найденных трендов с помощью «восхождения по дереву категорий» позволяет при обнаружении некоторого конкретного объекта сопоставить ему технологию (пример: при обнаружении «Шагоход» будет добавлена область «Роботы»), что обеспечивает высокую полноту извлечения технологических областей.

Заключение

Предложен способ автоматизации анализа частот упоминаний тех или иных наименований организаций вместе с различными технологическими областями с использованием русскоязычной Википедии. Использование актуальных данных постоянно расширяющейся энциклопедии позволит не упустить появление новых областей. Кроме того, выбранный подход является достаточно общим, чтобы можно было говорить о возможности применения изложенных методов для анализа не только «технологических» текстов.

Благодарности

За поддержку, критику и ценные рекомендации автор высказывает благодарности (в алфавитном порядке) Д. А. Кану, Ю. В. Каткову, С. В. Серебрякову, А. Л. и Т. В. Тулупьевым, А. В. Уланову и А. А. Фильченкову.

Литература

1. *Ткаченко М. В*. Метод выделения именованных сущностей на основе Википедии. Дипломная работа. Математико-механический факультет СПбГУ. 2011. 36 с.
2. *Ширин С. С*. Всемирная паутина как объект исследования в политической науке // Вестник Санкт-Петербургского университета. Сер. 6: Философия. Культурология. Политология. Право. Международные отношения. 2013. № 2. С. 98–105.
3. *Bollen J., Mao H., Zeng X*. Twitter mood predicts the stock market // Journal of Computational Science. 2011. Vol. 2. No. 1. Pp. 1–8.
4. *Dinarelli M., Rosset S*. Tree representations in probabilistic models for extended named entities detection // Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL «12). 2012. Association for Computational Linguistics, Stroudsburg, PA, USA. Pp. 174–184.
5. *Gareev R., Tkachenko M., Solovyev V., Simanovsky A., Ivanov V*. Introducing baselines for russian named entity recognition // Proceedings of the 14th international conference on Computational Linguistics and Intelligent Text Processing — Volume Part I (CICLing'13). 2013. Springer-Verlag, Berlin, Heidelberg. Vol. Part I. Pp. 329–342.
6. *Malouf R*. Markov models for language-independent named entity recognition // Proceedings of the 6th conference on Natural language learning. 2002. Association for Computational Linguistics, Stroudsburg, PA, USA. Vol. 20. Pp. 1–4.
7. *McCallum A., Li W*. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons // Proceedings of the seventh conference on Natural language learning at HLT-NAACL. Association for Computational Linguistics, Stroudsburg, PA, USA. 2003. Vol. 4. Pp. 188–191.
8. *Preis T., Moat H. S., Stanley H.* E. Quantifying Trading Behavior in Financial Markets Using Google Trends. Scientific Reports. 2013. Vol. 3. Pp. 1684.

ЭКСПРЕСС-АНАЛИЗ РЕПЛИК И МЕТАДАННЫХ СОЦИАЛЬНЫХ СЕТЕЙ С ИСПОЛЬЗОВАНИЕМ ПРОГРАММНЫХ СРЕДСТВ АВТОМАТИЗАЦИИ ПОЛУЧЕНИЯ ДАННЫХ

А. Е. Пащенко

к.т.н., н.с. СПИИРАН

E-mail: AEP@iias.spb.su,

Т. В. Тулупьева

к. пс. н., доц. каф. информатики СПбГУ, с.н.с. СПИИРАН

Аннотация. В докладе рассматривается подход к экспресс-анализу данных, полученных со страниц пользователей социальных сетей. Одним из основных технологических вопросов является сохранение реплик и метаданных социальных сетей, в которых содержится информация, представленная на естественном языке, и которая является основным материалом для анализа.

Введение

В последнее время социальные сети получили огромную популярность. Они используются не только как площадка для общения со знакомыми людьми, но все больше — как инструмент хранения всей электронной информации: помимо текстов и заметок это могут быть видеофайлы, аудиофайлы, картинки, а также ряд других информационных «сущностей».

Кроме того, следует отметить, что все больше социальные сети являются основным инструментом представления и продвижения различных мнений, социальных и политических программ [5,6], и в этой связи социальные сети престают быть просто инструментом межличностного общения двух или более индивидов, а становятся влиятельным средством массовой информации. В этой связи появляется потребность оперативного содержательного анализа информации, хранящейся в социальной сети.

Описание предметной области

Среди социальных медиа [3] (это вид массовой коммуникации, осуществляемый посредством интернета), можно выделить семь разновидностей наиболее используемых форм: социальные сети, блоги и микроблоги, форумы, сайты отзывов, фото и видеохостинги, сайты знакомств, геосоциальные сервисы [4]. Но в данной работе мы станем рассматривать только социальные сети, так как они имеют наибольшую аудиторию, большое количество лич-

ной информации о пользователях, а также различными способами в социальных сетях отражается информация из других социальных медиа.

Классические методы социологического исследования, нацеленного на сбор и последующий анализ данных, следующие [1]:

1. Наблюдение;
2. Опрос;
3. Анализ документов;
4. Методы социометрии;
5. Эксперимент в социологии.

Наблюдение — это метод сбора первичных эмпирических данных, который заключается в направленности, систематическом восприятии и регистрации значимых с точки зрения целей и задач исследования социальных процессов, явлений, ситуаций, фактов, подвергающихся контролю и проверке.

Опрос — метод получения первичной социологической информации, основанный на устном или письменном обращении к исследуемой совокупности людей с вопросами, содержание которых представляет проблему исследования.

Анализ документов — это совокупность методических приемов, применяемых для извлечения из документальных источников социологической информации, необходимой для решения исследовательских задач.

Социометрия — метод сочетания опросной методики и алгоритмов специальной обработки первичных измерений, который сводится к исчислению разнообразных персональных и групповых индексов.

Эксперимент — это метод сбора и анализа эмпирических данных, направленный на проверку гипотез относительно причинных связей между явлениями. Обычно эта проверка производится путем вмешательства экспериментатора в естественный ход событий.

Как мы видим, все классические методы социологического анализа можно использовать, если принять социальные сети, и информацию, содержащуюся в них, как универсальный источник данных. Существенным отличием информации, хранящейся в социальных сетях, а вместе с тем и исследований, базирующихся на анализе данной информации, является изначальное ее наличие в электронном виде, в отличие от классических социологических исследований, где наиболее существенной частью исследования является получение информации от респондентов, с последующим ее занесением и верификацией в базу данных.

Следует отметить, что хотя данные в социальных сетях и хранятся в электронном виде, но структура их хранения и возможность использования существенно ограничены (что зачастую отражается термином «слабоструктурированные данные»), так как если использовать информацию из страницы веб-браузера непосредственно, мы получим ситуацию, сходную с получением информации в классических социологических исследованиях.

В работе мы рассмотрим технические средства извлечения записей с пользовательской стены в социальной сети ВКонтакте. Для начала необходимо определимся, с какими данным мы хотим работать.

Зададимся целью извлечь пользовательские записи с профилей в социальной сети ВКонтакте. А именно: личную информацию о самом пользователе — его имя и фамилию, идентификационный номер его профиля и ссылку на него, а также ряд других. По этому набору данных можно легко идентифицировать пользователя, с целью дальнейшей работы над ним.

Данная социальная сеть имеет, как уже говорилось ранее, различные виды информации из социальных медиа.

При извлечении данных основной акцент сделаем на анализе комментариев к видео-постам, так как такой набор данных содержит большой объем информации, который крайне трудоемко анализировать содержательно, используя классические методы социального анализа.

Технические средства автоматизации

Для создания программного продукта использовался язык программирования C#, а также была использована технология API запросов к социальной сети vk.com.

Для того чтобы вызвать метод API Вконтакте, необходимо осуществить запрос по протоколу HTTPS на следующий URL:

https://api.vk.com/method/«METHOD_NAME»?»PARAMETRS»&access_token=«ACCESS_TOKEN»,

где METHOD_NAME — название метода, PARAMETRS — параметры соответствующего метода API, ACCESS_TOKEN — ключ доступа (для его получения каждому пользователю приложения необходимо авторизоваться в сети Вконтакте). В ответ на такой запрос мы можем получить ответ в формате JSON или XML. В нашем приложении мы получаем ответ в формате XML. Далее, мы извлекаем необходимую нам информацию их XML-документа и выводим ее на форму.

Например, метод video.get возвращает следующую информацию о видеозаписях:

Тэг в XML-документе	Описание	Формат
vid	идентификатор видеозаписи	положительное число
owner_id	идентификатор владельца видеозаписи	int (числовое значение)
title	название видеозаписи	строка
description	текст описания видеозаписи	строка
duration	длительность ролика в секундах	положительное число

Тэг в XML-документе	Описание	Формат
link	строка, состоящая из ключа video+vid	строка
date	дата добавления видеозаписи в формате unixtime (определяется как количество секунд, прошедших с полуночи 1 января 1970 года)	положительное число
views	количество просмотров видеозаписи	положительное число
image	url изображения-обложки ролика с размером 160×120px	строка
Image_medium	url изображения-обложки ролика с размером 320×240px	строка
comments	количество комментариев к видеозаписи	положительное число
player	адрес страницы с плеером, который можно использовать для воспроизведения ролика в браузере. Поддерживается flash и html5, плеер всегда масштабируется по размеру окна	строка

Программная реализация

Возможно два типа загрузки данных: по ID пользователя и по ID группы.

В первом случае в поле «Введите ID пользователя» вводим id пользователя, данные о котором желаем получить, после нажимаем кнопку «Загрузить». Данные будут загружены в таблицы приложения и одновременно сохранятся в базу данных.

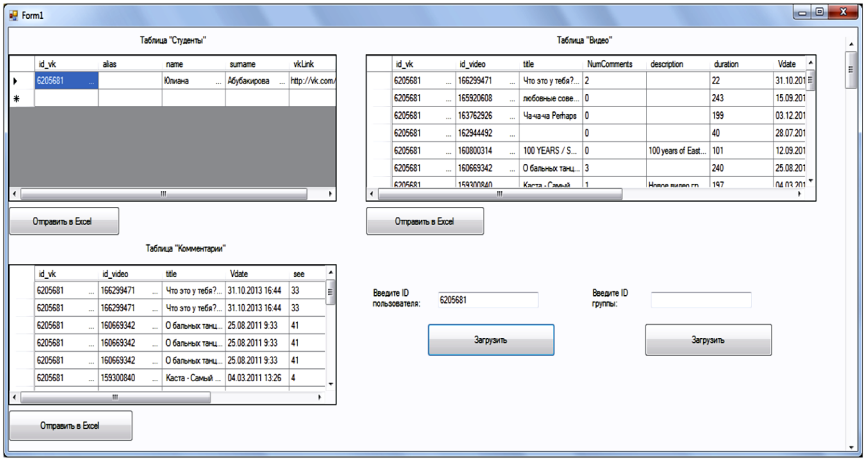
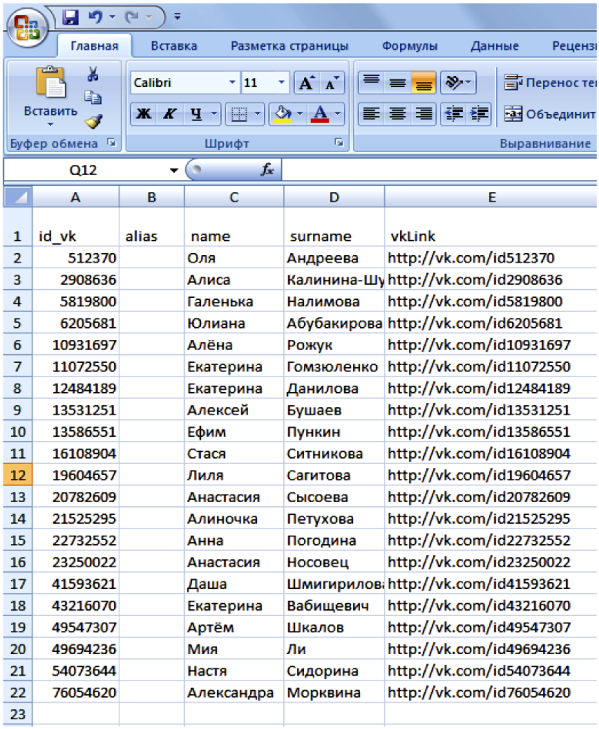


Рис. 1. Основное окно приложения



	A	B	C	D	E
1	id_vk	alias	name	surname	vkLink
2	512370		Оля	Андреева	http://vk.com/id512370
3	2908636		Алиса	Калинина-Шу	http://vk.com/id2908636
4	5819800		Галенька	Налимова	http://vk.com/id5819800
5	6205681		Юлиана	Абубакирова	http://vk.com/id6205681
6	10931697		Алёна	Рожук	http://vk.com/id10931697
7	11072550		Екатерина	Гомзюленко	http://vk.com/id11072550
8	12484189		Екатерина	Данилова	http://vk.com/id12484189
9	13531251		Алексей	Бушаев	http://vk.com/id13531251
10	13586551		Ефим	Пункин	http://vk.com/id13586551
11	16108904		Стася	Ситникова	http://vk.com/id16108904
12	19604657		Лиля	Сагитова	http://vk.com/id19604657
13	20782609		Анастасия	Сысоева	http://vk.com/id20782609
14	21525295		Алиночка	Петухова	http://vk.com/id21525295
15	22732552		Анна	Погодина	http://vk.com/id22732552
16	23250022		Анастасия	Носовец	http://vk.com/id23250022
17	41593621		Даша	Шмигирилова	http://vk.com/id41593621
18	43216070		Екатерина	Вабищевич	http://vk.com/id43216070
19	49547307		Артём	Шкалов	http://vk.com/id49547307
20	49694236		Мия	Ли	http://vk.com/id49694236
21	54073644		Настя	Сидорина	http://vk.com/id54073644
22	76054620		Александра	Морквина	http://vk.com/id76054620
23					

Рис. 2. Экспорт данных в Excel

На Рис. 1 показано окно приложение. Оно включается в себя таблицы с контактной информацией о пользователях, информации о видео контенте доступном на их страницах, а также комментариях к этим видеозаписям.

Кроме того, в окне «Загрузка данных в БД» под каждой таблицей есть кнопки «Отправить в Excel». При нажатии на кнопку данные из выбранной таблицы отправляются в таблицу Excel, что позволяет анализировать данные с использованием классических инструментариев, таких как пакет анализа данных SPSS или «Статистика».

Заключение

Рассматрен подход к анализу данных, полученных со страниц пользователей социальных сетей, в котором необходимо использовать методы автоматической обработки текста. Одним из основных вопросов является сохранение реплик и метаданных в социальных сетях, в которых содержится информация, сформулированная на естественном языке, и которая является основным материалом для анализа.

Литература

1. Политология: методы исследования. [Электронный ресурс]. URL: http://www.read.virmk.ru/s/SANZ_SOC/g-0353.htm
2. *Moser K. D.* SQl SERVER, Apps bounds to tighten // Ziff Davis Media Inc.1993.
3. *Балуев Д. Г.* Политическая роль социальных медиа как поле научного исследования // Образовательные технологии и общество (Educational Technology & Society), 2013. Т. 16. № 2. С. 604–616
4. Развитие интернета в регионах России [Электронный ресурс]. URL: http://company.yandex.ru/researches/reports/2014/ya_internet_regions_2014.xml
5. *Шалимов А. Б.* Диалектика социального и индивидуального // Некоммерческое партнёрство «Проектно-аналитическое агенство «Шаг». 2013. С. 1030–1036.
6. *Варлыгина З. В.* Тенденции развития тематических социальных сетей в российском интернете. 2008. С. 220–227.
7. *Хансен Марк Д., Чоу Ричард Т., Маури Кевин К., Смит Дуайт Р., Уорден Джеймс П.* Система и способ для управления доступом ненадежных приложений к защищённому контенту // Патент на изобретение. 2010.
8. Интерфейс программирования приложений [Электронный ресурс]. URL: http://ru.wikipedia.org/wiki/Api
9. *Э. Фримен.* Изучаем HTML, XHTML и CSS = Head First HTML with CSS & XHTML // СПб.: Питер, 2010. 656 с. 9. Работа с API. Авторизация [Электронный ресурс]. URL: https://vk.com/dev/authentication
10. Список методов секции users: users.get [Электронный ресурс].URL: https://vk.com/dev/users.get

АЛГЕБРАИЧЕСКИЕ БАЙЕСОВСКИЕ СЕТИ: ОТКРЫТЫЕ ВОПРОСЫ ЛОКАЛЬНОГО АВТОМАТИЕСКОГО ОБУЧЕНИЯ

А. Л. Тулупьев
проф., зав. лаб.; СПбГУ, СПИИРАН
E-mail: alexander.tulupyev@gmail.com

Аннотация. Алгебраические байесовские сети являются одним из классов вероятностных графических моделей, предложены В. И. Городецким. Ряд операций в алгебраических байесовских сетях изучены достаточно хорошо: проверка и поддержание непротиворечивости, априорный вывод, апостериорный вывод.

Однако, несмотря на определенные успехи, нуждается в развитии теория машинного обучения алгебраических байесовских сетей как на локальном (параметрическом), так и на глобальном (структурном) уровнях.

Введение

Алгебраические байесовские сети (АБС) — одна из логико-вероятностных графических моделей баз фрагментов знаний с неопределенностью [1, 2, 4, 6, 7]. Успехи теории АБС уже обсуждались на конференции СПИСОК и Лавровских чтениях [5], отражены в ряде других публикаций. Целью доклада является постановка и обсуждение нерешенных задач локального машинного обучения (или автоматического обучения) алгебраических байесовских сетей.

В теории АБС в первую очередь предполагается, что знания о предметной области могут быть представлены в виде системы случайных бинарных (булевских, логических) элементов, допускающей декомпозицию. Отдельные части системы, которые получились в результате декомпозиции, называются фрагментами знаний. Фрагменты знаний могут пересекаться, но такие пересечения должны быть достаточно «редки».

Математической моделью фрагмента знаний в АБС выступает идеал конъюнктов, построенный над небольшим алфавитом атомарных пропозициональных формул (атомов). Каждому конъюнкту в идеале приписывается либо скалярная, либо интервальная оценка его истинности. Для краткости идеал с оценками далее будем называть фрагментом знаний [4–7].

Совокупность фрагментов знаний формирует собой алгебраическую байесовскую сети и является ее первичной структурой. Система связей между фрагментами знаний в АБС является ее вторичной структурой [4–6, 8, 9, 10].

Помимо конъюнктов над заданным алфавитом можно построить пропозициональные формулы-кванты: они представляют собой конъюнкцию

всех атомов из заданного алфавита, причем каждый атом в эту конъюнкцию входит один раз, но либо с отрицанием, либо без такового. Фактически, в вероятностной логике набор квантов является множеством элементарных событий [1, 2, 6].

Для фрагмента знаний, построенного над идеалом конъюнктов, набор оценок вероятностей записывается в виде вектора **P**; для фрагмента знаний над набором квантов [2, 6 — в виде вектора **Q**. Эти два вектора оценок связаны между собой [1, 2, 7]. Доклад в значительной степени ассоциирован с публикациями [3, 7, 8] и следует сложившейся в них системе терминов и обозначений

Постановка задачи

В задачах локального обучения нас не будут интересовать АБС на глобальном уровне. Будем предполагать, что нам известна структура некоторого фрагмента знаний (она однозначно задается указанием алфавита атомов, над которым фрагмент построен). Задачей локального обучения является формирование оценок вероятностей конъюнктов, вошедших в ФЗ [8].

Исходные данные для локального обучения могут поступать из «источников» трех типов:

1. Выборка совместных означиваний сразу всех наблюдающихся случайных элементов, в ней возможны пропуски;
2. Априорное распределение или семейство распределений над элементами фрагмента знаний;
3. Нечисловые сведения о соотношениях вероятностей элементов фрагмента знаний или о соотношениях истинностных означиваний пропозициональных формул, построенных из атомов, вошедших во фрагмент знаний.

Первый источник — выборку — называют обучающей выборкой.

Автоматическое обучение предполагает, что компьютерная программа сама, уже без вмешательства специалистов, формирует оценки вероятностей конъюнктов и/или квантов, обрабатывая доступные исходные данные.

Выборка без пропусков

Элементы обучающей выборки без пропусков, с точки зрения теории АБС, являются квантами.

Пусть задан фрагмент знаний над алфавитом из трех атомов $A = \{x_2, x_1, x_0\}$. Набор квантов

$$S = \left\{ \begin{array}{l} x_2 x_1 x_0,\ x_2 \overline{x}_1 x_0,\ x_2 x_1 x_0, \\ x_2 \overline{x}_1 \overline{x}_0,\ \overline{x}_2 x_1 x_0,\ x_2 x_1 x_0,\ x_2 \overline{x}_1 x_0 \end{array} \right\}$$

служит примером выборки.

Для каждого отдельно взятого фрагмента знаний алфавит фиксирован, а порядок следования атомов в кванте может быть заранее оговорен, поэтому нет необходимости все время использовать буквенно-индексные обозначения для того, чтобы указать означивания того или иного атомарного литерала. Договоримся его отрицательному означиванию сопоставлять 0, а положительному — 1. Тогда та же самая выборка может быть записана как $S = \{111, 101, 111, 100, 011, 111, 101\}$, что заметно короче.

Таким образом, каждому кванту можно сопоставить фрагмент знаний с бинарными оценками вероятности истинности, то есть с оценками вероятности, которые могут принять одно из двух значений: 0 или 1.

Последовательность означиваний, из которых состоит выборка, нелегка для восприятия и труднообозрима, особенно если она большая. Однако выборку можно записать в более компактной форме: кванты упорядочить, повторы среди них исключить, каждому кванту сопоставить число — сколько раз он встречается в обучающей выборке. Если поделить каждое такое число на общее число элементов в выборке, то получим оценку вероятности истинности кванта. Таким образом будет сформирован вектор **Q**, задающий распределение вероятностей на квантах. Его можно преобразовать в вектор **P** — особым образом упорядоченную совокупность вероятностей конъюнктов — с помощью уравнения $\mathbf{P} = \mathbf{J}\mathbf{Q}$ [5, 7, 12].

Любой из векторов **Q** и **P** может служить результатом локального обучения по имеющейся выборке. Как правило, число элементов во фрагменте знаний невелико; много меньше, чем объем выборки; поэтому получившийся результат будет более обозрим, чем исходные данные.

Формализуем описанную процедуру и обсудим ее некоторые особенности, не особенно заметные на первый взгляд. Пусть выборка состоит из набора квантов $S = \{q^i\}_{i=0}^{N-1}$, где N — это размер выборки. Каждому кванту можно сопоставить $\mathbf{Q}^i$ — вектор вероятностей квантов, построенных над соответствующим алфавитом. В векторе $\mathbf{Q}^i$ все элементы нули, за исключением того, который соответствует кванту q^i: этот элемент вектора равен единице.

При введенных обозначениях и соглашениях процедура обучения выглядит совсем просто:

$$\mathbf{Q} = \frac{1}{N}\sum_{i=0}^{N-1}\mathbf{Q}^i. \tag{3}$$

Формулу (3) легко адаптировать для ситуации, когда элементы выборки поступают друг за другом. При этом на каждом шаге доступен «обученный» уже поступившими сведениями вектор **Q**. Пусть у нас совершается i-тый шаг; $\mathbf{Q}[i-1]$ — результат обучения предшествующими элементами выборки, поступает элемент $\mathbf{Q}^i$. Тогда «дообучение» будет выглядеть как

$$\mathbf{Q}[i] = \frac{i\mathbf{Q}[i-1] + \mathbf{Q}^i}{i+1} = \mathbf{Q}[i-1] + \frac{\mathbf{Q}^i - \mathbf{Q}[i-1]}{i+1}. \tag{4}$$

Для простоты примем $\mathbf{Q}[-1]=\mathbf{0}$, поскольку на первой итерации он умножается на ноль и далее роли не играет.

Для вектора вероятностей конъюнктов справедливы аналогичные уравнения:

$$\mathbf{P}=\frac{1}{N}\sum_{i=0}^{N-1}\mathbf{P}^i, \tag{5}$$

$$\mathbf{P}[i]=\frac{i\mathbf{P}[i-1]+\mathbf{P}^i}{i+1}=\mathbf{P}[i-1]+\frac{\mathbf{P}^i-\mathbf{P}[i-1]}{i+1}. \tag{6}$$

Для удобства будем считать, что $\mathbf{P}[-1]=\dot{\mathbf{0}}$, то есть все компоненты этого вектора нулевые, за исключением самой первой: она равна 1. Заметим, что этот вектор непротиворечив как совокупность оценок вероятностей.

Уравнения (4) и (6) можно переписать как линейную комбинацию векторов:

$$\mathbf{Q}[i]=\frac{i\mathbf{Q}[i-1]+\mathbf{Q}^i}{i+1}=\frac{i}{i+1}\mathbf{Q}[i-1]+\frac{1}{i+1}\mathbf{Q}^i, \tag{7}$$

$$\mathbf{P}[i]=\frac{i\,\mathbf{P}[i-1]+\mathbf{P}^i}{i+1}=\frac{i}{i+1}\mathbf{P}[i-1]+\frac{1}{i+1}\mathbf{P}^i. \tag{8}$$

Векторы $\mathbf{Q}[i-1]$, $\mathbf{P}[i-1]$, $\mathbf{Q}^i$, $\mathbf{P}^i$ непротиворечивы. В теории АБС известно, что линейная комбинация непротиворечивых фрагментов знаний одинаковой структуры будет являться непротиворечивым фрагментом знаний; это же справедливо и для соответствующих им векторов оценок. Таким образом, в процессе обучения у нас формируются непротиворечивые фрагменты знаний, то есть оказываются непротиворечивыми наборы оценок $\mathbf{Q}[i]$, $\mathbf{P}[i]$.

Уравнения, записанные в виде (7) и (8), можно рассматривать как один из вариантов адаптации нашей системы знаний $\mathbf{Q}[i-1]$ и $\mathbf{P}[i-1]$ к поступившим новым сведениям $\mathbf{Q}^i$ и $\mathbf{P}^i$. Однако адаптивное обучение может строиться на иных (более общих, но похожих) уравнениях:

$$\mathbf{Q}[i]=\alpha\mathbf{Q}[i-1]+(1-\alpha)\mathbf{Q}^i, \tag{9}$$

$$\mathbf{P}[i]=\alpha\mathbf{P}[i-1]+(1-\alpha)\mathbf{P}^i, \tag{10}$$

где $0\le\alpha\le1$, а векторы $\mathbf{Q}[-1]$ и $\mathbf{P}[-1]$ непротиворечивы и представляют наши априорные знания или предположения о системе. Подход, представленный уравнениями (9) и (10), имеет очень глубокие теоретические основания, с ними можно ознакомиться, например, в [7].

Уравнения адаптации могут быть составлены и иначе:

$$\mathbf{Q}_{\text{result}}=\alpha\mathbf{Q}_{\text{a priori}}+(1-\alpha)\mathbf{Q}[N-1], \tag{11}$$

$$\mathbf{P}_{\text{result}}=\alpha\mathbf{P}_{\text{a priori}}+(1-\alpha)\mathbf{P}[N-1]. \tag{12}$$

Выбор стратегии обучения (уравнений обучения) зависит от доступных источников данных, их типов, предметной области и от стоящей перед интеллектуальной системой задач.

Выборка с пропусками

Обучение по «хорошей» выборке, в которой нет элементов с пропусками в означиваниях, идеологически просто — сводится к подсчету частот появления того или иного означивания. Затем полученные результаты могут быть дополнительно скомбинированы с некоторым объектом, представляющим априорные знания, однако и здесь проблем не возникает. К сожалению, не всегда оказывается так, что обучение проходит по «хорошим» (то есть без пропусков) исходным данным. Общим случаем обработки данных «с дефектами» и восстановления их пропущенных элементов занимаются целые отрасли статистики и искусственного интеллекта [1, 14, 16]. Мы же рассмотрим возникающие проблемы в рамках и на языке логико-вероятностного подхода, которого придерживается теория алгебраических байесовских сетей. Кроме того, ограничимся лишь тем случаем, когда нам не важен порядок поступления элементов (реализаций) выборки, а значит, эти элементы можно переставлять и, что особенно важно, группировать по одинаковым означиваниям — см. табл. 1 и 2.

Таблица 1

Элементы выборок без пропусков

№ 10	№ 2	Число атомов в элементе выборки		
		1	2	3
0	000	0	00	000
1	001	1	01	001
2	010		10	010
3	011		11	011
4	100			100
5	101			101
6	110			110
7	111			111

В теориях байесовских и марковских сетей предполагается, что каждый отдельно взятый ФЗ может быть построен над небольшим набором атомов. Тогда все кванты, сформированные над таким набором можно проиндексировать и перечислить в таблице (или в массиве, если речь идет о коде программы).

Таблица 2

Элементы выборок с пропусками

№ 10	№ 3	Число атомов в элементе выборки		
		1	**2**	**3**
0	000	0	00	000
1	001	*	0*	00*
2	002	1	01	001
3	010		*0	0*0
4	011		**	0**
5	012		*1	0*1
6	020		10	010
7	021		1*	01*
8	022		11	011
9	100			*00
10	101			*0*
11	102			*01
12	110			**0
13	111			***
14	112			**1
13	120			*10
16	121			*1*
17	122			*11
18	200			100
19	201			10*
20	202			101
21	210			1*0
22	211			1**
23	212			1*1
24	220			110
25	221			11*
26	222			111

Примечание: Здесь * указывает на пропущенное означивание.

В табл. 1 перечислены элементы обучающих выборок без пропусков. Их удобно индексировать числами в двоичной системе счисления. В программном коде такую индексацию удобно использовать в массиве, в котором накапливаются частоты встречаемости различных элементов в выборке. При наличии такого массива процесс локального обучения сводится к тем уравнениям, которые были приведены в предшествующем разделе.

В табл. 2 перечислены элементы обучающих выборок с пропусками. Их удобно индексировать числами в троичной системе счисления. Как и раньше, элементы без пропусков можно использовать в локальном обучении ФЗ с помощью уравнений из предшествующего раздела. Однако элементы с пропусками требуют адаптации предложенного подхода.

Наличие или отсутствие закономерности, которой подчиняются места расположения пропущенных означиваний, играет существенную роль. Классификация такого рода закономерностей предлагается в соответствующей литературе.

Чтобы двинуться дальше, требуется исследовать семантику * — пропущенного означивания. На самом деле, на месте знака * либо мог стоять 0, либо могла стоять 1. Таким образом, элементу с пропущенными означиваниями может соответствовать несколько различающихся элементов без пропущенных значений (или возможных выборок, а значит, всей исходной выборке с пропусками — несколько различных выборок, не противоречащих исходной.

Можно говорить о семействе выборок без пропусков, не противоречащих исходной выборке с пропусками. Таким образом, выборка с пропусками задает семейство распределений вероятностей, а значит фрагмент знаний, построенный на ее основе, не может содержать лишь скалярные оценки истинности. Без дополнительных предположений, гипотез или соглашений невозможно оправдать выбор одного распределения вероятностей из многих для того, чтобы обеспечить каждый элемент фрагмента знаний скалярной оценкой вероятности истинности. Следовательно, открывается два выбора:

1) либо допустить интервальные оценки истинности,
2) либо предложить принцип выбора или формирования одного распределения при условиях, что имеются несколько возможных.

Обучение интервальных оценок

Можно строить (обучать) интервальные оценки, если всякий раз, когда встречается знак *, будем рассматривать оба допустимых случая его означивания (0 и 1) и вести подсчет нижней и верхней оценки частоты встречаемости положительного означивания.

Точно такой же подход можно применить и при обучении вероятностей конъюнктов во фрагменте знаний, образованном над алфавитом из двух или более атомов, когда поступающая выборка содержит пропущенные эле-

менты в означиваниях. Следует заметить, что здесь мы вторгаемся в богатую область исследований, связанную с многозначными логиками [2].

Таким образом, мы можем обработать обучающую выборку и получить верхнюю и нижнюю оценку вероятности каждого конъюнкта из фрагмента знаний. Равно мы можем получить верхние и нижние оценки вероятностей квантов. Следует учесть, что если в векторах **Q** и **P** присутствуют интервальные оценки, то уравнения **Q** = **IP** и **P** = **JQ** не будут уже иметь места в общем случае.

Примером дополнительных сведений, которые применимы при формировании интервальных оценок, может послужить соотношение вероятностей вида $p(x_1) \geq p(x_0)$; на его основе можно сузить оценки, полученные в результате обучения.

Формирование скалярной оценки

Чтобы получить скалярные оценки элементов фрагмента знаний, построенного либо над идеалом конъюнктов, либо над набором квантов, необходимо выбрать какое-то одно распределение вероятностей, причем в условиях дефицита информации, то есть в случае, когда обучающая выборка содержит означивания с пропусками элементов. В этом разделе подход, предложенный Н. В. Ховановым [9], адаптируется для применения к фрагментам знаний.

Каждому элементу выборки с пропусками можно сопоставить его возможные означивания без пропусков. По каждому означиванию без пропусков можно построить ФЗ с бинарными оценками, а из них построить линейную комбинацию с равными весами. Получится фрагмент знаний с точечными оценками. Далее такие ФЗ комбинируются с весами, учитывающими частоту встречаемости соответствующих элементов, по всей выборке. Таким образом по выборке с пропусками будет сформирован ФЗ с точечными оценками вероятностей.

Дополнительные сведения могут позволить исключить из рассмотрения некоторые возможные означивания пропусков в элементах выборки. Это изменит результат операции линейной комбинации. Примером таких сведений служит высказывание «как правило, утверждение $x_1 \supset x_0$ справедливо». При принятых обозначениях атомов окажутся возможными только три сочетания: 00, 01 и 11. Сочетание 10 будет противоречить приведенному правилу.

Заключение

Локальное обучение в теории АБС сводится к формированию оценок конъюнктов или квантов из фрагмента знаний. Формирование оценок осуществляется за счет комбинации сведений о частоте встречаемости элементов выборки, предположений об априорных распределениях и иной дополнительной информации. Хотя основные принципы, которые должны лечь

в основу локального машинного обучения АБС, установлены и достаточно ясны, ряд вычислительных и семантических аспектов остаются неизученными, а вопросы комбинирования сведений из различных типов источников находятся только в начале рассмотрения. Задачи локального машинного обучения АБС оказались связаны с задачами глобального машинного обучения этих сетей, то есть машинного обучения структуры АБС [7, 10, 11], причем результаты глобального обучения в определенной ступени формируют «заказ» на локальное обучение за счет выделений тех фрагментов знаний, оценки вероятности в которых требуется сформировать.

Литература

1. *Тулупьев А. Л.* Алгебраические байесовские сети: локальный логико-вероятностный вывод. СПб.: СПбГУ; Анатолия. 2007. 80 с.
2. *Тулупьев А. Л.* Байесовские сети: логико-вероятностный вывод в циклах. СПб: Изд-во С.-Петерб. ун-та, 2008. 140 с.
3. *Тулупьев А. Л.* Обработка дополнительной нечисловой информации в локальном обучении алгебраических байесовских сетей по выборкам с пропусками // Международная конференция по мягким вычислениям и измерениям. Сборник докладов. 2009. Т. 1. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2009. С. 139–142.
4. *Тулупьев А. Л.* Алгебраические байесовские сети: система операций глобального логико-вероятностного вывода // Информационно-измерительные и управляющие системы. 2010. Т. 8. № 11. С. 65–71.
5. *Тулупьев А. Л.* Алгебраические байесовские сети и родственные модели знаний с неопределенностью // Лавровские чтения-2013: Материалы всероссийской научной конференции по проблемам информатики. Пленарные заседания (23–26 апреля 2013 г., С.-Петербург). СПб.: ИПК «Береста», 2013. С. 67–77.
6. *Тулупьев А. Л., Николенко С. И., Сироткин А. В.* Байесовские сети: логико-вероятностный подход. СПб.: Наука, 2006. 607 с.
7. *Тулупьев А. Л., Сироткин А. В., Николенко С. И.* Байесовские сети доверия: логико-вероятностный вывод в ациклических направленных графах. СПб.: Изд-во С.-Петерб. ун-та, 2009. 400 с.
8. *Тулупьев А. Л., Фильченков А. А., Вальтман Н. А.* Алгебраические байесовские сети: задачи автоматического обучения // Информационно-измерительные и управляющие системы. 2011. Т. 9. № 11. С. 57–61.
9. *Хованов Н. В.* Анализ и синтез показателей при информационном дефиците. СПб.: Изд-во С.-Петерб. ун-та, 1996. 196 с.
10. *Фильченков А. А., Тулупьев А. Л.* Связность и ацикличность первичной структуры алгебраической байесовской сети // Вестник Санкт-Петербургского университета. Серия 1: Математика. Механика. Астрономия. 2013. № 1. С. 110–119.
11. *Фильченков А. А., Фроленков К. В., Сироткин А. В., Тулупьев А. Л.* Система алгоритмов синтеза подмножеств минимальных графов смежности // Труды СПИИРАН. 2013. № 4 (27). С. 200–244.

АВТОМАТИЗИРОВАННОЕ ИЗВЛЕЧЕНИЕ ДАННЫХ С ПОЛЬЗОВАТЕЛЬСКОЙ СТРАНИЦЫ В СОЦИАЛЬНОЙ СЕТИ, ЭКСПОРТ ИХ В БАЗУ ДАННЫХ И В ЭЛЕКТРОННЫЕ ТАБЛИЦЫ

Е. В. Иванова
студентка СПбГУ
E-mail: lena.iwfyl@mail.ru

А. Е. Пащенко
к.т.н., н.с. СПИИРАН
E-mail: AEP@iias.spb.su

А. Л. Тулупьев
д.ф.-м.н., проф. СПбГУ
E-mail: alexander.tulupyev@gmail.com

Аннотация. В докладе рассматривается подход к автоматизированному извлечению данных из социальной сети ВКонтакте, экспорт их в базу данных Microsoft SQL Server и в электронные таблицы Microsoft Excel.

Введение

Поскольку социальные сети стали очень популярны, в связи с этим появилась потребность в обработке социального контента, появляющегося в сети интернет, с целью анализа желаний и настроений пользовательских аудиторий. Извлекать обрабатываемые данные вручную очень долгая и сложная работа. Поэтому появилась необходимость ускорить этот процесс [2].

В докладе рассматриваются программные средства, с помощью которых можно автоматизировать работу по извлечению необходимых данных, на примере записей, комментариев и иного контента, полученного из социального сервиса ВКонтакте.

Первой задачей в этой области является извлечение данных с помощью api-запросов, что удобно реализовано в приложении Visual Studio. Для проекта следующей задачей является представление и хранение данных в Microsoft SQL Server, эта **система управления реляционными базами данных** удобно взаимодействует с инструментами Visual Studio и помогает представить данные в приложении в табличном виде. Заключительной задачей является обработка извлечённой информации, а именно представление в удобном табличном виде, с помощью Microsoft Excel, с целью упрощения дальнейшей работы над ней и возможностью её изменения.

Описание предметной области

Тема развития и значимости информационных технологий стала довольно актуальной за последние несколько лет. В отличие от ситуации десятилетней давности, сейчас уже трудно найти человека, который не умеет пользоваться сетью интернет [1]. Вместе с этим набирает популярности и другой аспект — социальные медиа [2, 3]. Это вид массовой коммуникации, осуществляемый посредством интернета. Можно выделить семь разновидностей наиболее используемых форм соцмедия: социальные сети, блоги и микроблоги, форумы, сайты отзывов, фото и видеохостинги, сайты знакомств, геосоциальные сервисы [4].

Есть несколько причин их стремительного развития, одной из них является появившаяся, благодаря социальным сетям, возможность общаться с новыми людьми или находить старых знакомых на расстоянии нескольких тысяч километров. Это намного удобнее, чем писать письма, которые доходят до адресата неделями.

Ещё одним фактором успеха социальных медиа является желание человека хорошо и весело проводить время, не выходя из помещения, ведь с помощью них можно не только общаться, но и посмотреть видео, и послушать музыку. К ресурсом, предоставляющим такую возможность можно отнести блог-платформы, видеохостинги, развлекательные и информационные ресурсы [2, 3].

Довольно большую популярность всегда имели социальные сервисы, где можно было вести блоги, интернет-дневники или участвовать в обсуждениях на форуме. С развитием этих форм общения стали образовываться социальные сети — совокупности участников, объединенных не только средой общения, но и с явно установленными связями между собой. В связи с этим, социологи отмечают потребность в исследованиях настроений аудитории, с помощью анализа контента социальных сетей [5, 6].

Поскольку это необходима небольшая, но оперативно сформированная выборка данных, то в докладе мы рассмотрим технические средства извлечения записей с пользовательской стены в социальной сети ВКонтакте. Для начала необходимо определимся, с какими данным мы хотим работать.

Используемые данные

Извлекать мы будем пользовательские записи с профилей в социальной сети ВКонтакте. А именно, личную информацию о самом пользователе — его имя и фамилию, идентификационный номер его профиля и ссылку на него. По этому набору данных можно легко идентифицировать пользователя, с целью дальнейшей работы над ним.

Для записи со стены мы выбрали следующие атрибуты: текст записи, и ссылки на содержащий там контент, типа видео или фото, а также коли-

чество комментариев и «лайков», идентификационный номер пользователя и самой записи. Выбранные нами атрибуты хранят довольно много содержательной информации о записи пользователя для последующего анализа. Эти атрибуты записей позволяют нам, на данном этапе, провести анализ активности пользователя, его читаемости и иных рейтинговых характеристик.

Наиболее интересным блоком базы данных является блок с комментариями к записям на стене. Туда входит: идентификационный номер записи и человека, оставившего комментарий, и текст самого комментария. Этот минимальный набор данных удобен для исследования содержательной части записей и настроения аудитории, комментирующей тот или иной пост.

Технические средства автоматизации

Основным средством извлечения данных из контакта служит технология API-запросов [7].

API (англ. *application programming interface, API*) — набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением (библиотекой, сервисом) для использования во внешних программных продуктах. Используется программистами для написания всевозможных приложений [8].

API для сайта — это, как правило, скрипт, который принимает запросы (по методам GET (site.ru/api.php?a=b), POST) и отдаёт не обычный HTML для браузеров, а результат запроса в определённом формате (XML, JSON). Соответственно API предназначен скрипту со стороннего сайта или сервиса, который посылает эти GET/POST запросы, получает результат и каким-то образом использует данные. Пользователям-программистам он нужен для интеграции с другими сайтами, сервисами, или автоматизации некоторых действий, создав программу для сайта.

Определим некоторые названные ранее понятия. GET — это метод передачи данных, который используется для запроса содержимого указанного ресурса. С помощью метод GET можно также начать какой-либо процесс. HTML — стандартный язык разметки документов в сети интернет [9], а XML — расширяемый язык разметки.

На сайте разработчиков социального сервиса ВКонтакте есть раздел «Работа с API», который включает себя информацию о том, как об авторизации, так и о самих запросах и правах доступа [9].

В результате прохождения авторизации мы получаем ключ доступа access_token, с помощью которого можем выполнять любые запросы к API ВКонтакте от имени пользователя или от имени приложения. В зависимости от набора полученных **прав доступа** некоторые методы API могут быть недоступны [7, 9].

Для вызова метода API ВКонтакте, мы осуществляем GET запрос по протоколу HTTPS на индивидуальный для каждого запроса URL-адрес, общая схема которого:

https://api.vk.com/method/'"METHOD_NAME"'?'"PARAMETERS"' &access_token='"ACCESS_TOKEN"'

В запросе:

METHOD_NAME — название метода из **списка функций API**,
PARAMETERS — параметры соответствующего метода API,
ACCESS_TOKEN — ключ доступа, полученный в результате успешной **авторизации приложения**.

Ответ на запрос получаем в формате XML, но есть возможность получение ответа в формате JSON.

Каждый метод имеет собственный набор поддерживаемых параметров, однако существуют параметры, которые принимают все методы, например, язык и версию API.

Все наборы параметров для запросов можно найти в пункте «Документация» на сайте разработчиков ВКонтакте. Так, например, для метода users.get предложен следующий список параметров [10]:

user_ids перечисленные через запятую идентификаторы пользователей или их короткие имена (screen_name). По умолчанию — идентификатор текущего пользователя.

fields список дополнительных полей, которые необходимо вернуть. Доступные значения: *sex, bdate, city, country, photo_50, photo_100, photo_200_orig, photo_200, photo_400_orig, photo_max, photo_max_orig, online, online_mobile, lists, domain, has_mobile, contacts, connections, site, education, universities, schools, can_post, can_see_all_posts, can_see_audio, can_write_private_message, status, last_seen, common_count, relation, relatives, counters, screen_name, timezone*

name_case падеж для склонения имени и фамилии пользователя. Возможные значения: именительный — nom, родительный — *gen*, дательный — *dat*, винительный — *acc*, творительный — *ins*, предложный — *abl*. По умолчанию *nom*.

Поскольку этот метод не требует ключа access_token, то запрос URL можно преобразовать до следующего вида:

https://api.vk.com/method/users.get.xml?user_ids=''«USER_IDS»''

В реализованном приложении для метода users.get был только параметр user_ids.

После получения информации в формате XML, с помощью свойства XmlNode.InnerText пространства имен System.Xml получаем текст, содержащийся внутри XML.

Выполненное приложение является desktop-приложением, поскольку оно несложно реализовывается на Visual Studio и отлично связывается с Microsoft SQL Server. Извлечённая информация по нажатию клавиши выводится в окно приложения и добавляется в таблицы базы данных.

На данном этапе схема базы данных выглядит следующим образом:

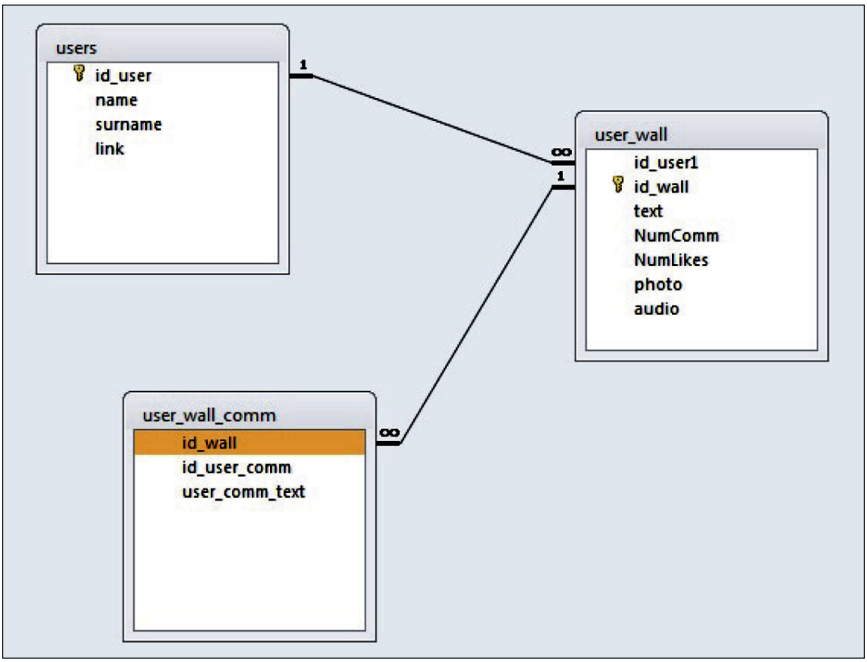


Таблица с описанием блока users:

№ п.п.	Имя поля	Тип данных	Описание
1	id_user	nchar(30), not null	Идентификационный номер пользователя
2	name	nchar(30), not null	Имя пользователя
3	surname	nchar(30), not null	Фамилия пользователя
4	link	nchar(30), null	Ссылка на профиль ВКонтакте пользователя

Таблица с описанием блока user_wall:

№ п.п.	Имя поля	Тип данных	Описание
1	id_user1	nchar(30), not null	Идентификационный номер пользователя.
2	id_wall		Идентификационный номер записи на стене пользователя.
3	text	nvarchar(max), null	Текст записи пользователя.
4	NumComm	int, null	Количество комментариев к записи пользователя.
5	NumLikes	int, null	Количество «лайков» к записи пользователя.
6	photo	varchar(max), null	Ссылка на графический объект в записи пользователя.
7	audio	nvarchar(max), null	Ссылка на ауди в записи пользователя.

Таблица с описанием блока user_wall_comm:

№ п.п.	Имя поля	Тип данных	Описание
1	id_wall	nchar(30), not null	Идентификационный номер записи на стене пользователя.
2	Id_user_comm	nchar(30), null	Идентификационный номер пользователя, оставившего комментарий.
3	user_comm_text	nvarchar(max), null	Текст комментария.

Программная реализация

В предыдущих пунктах были описаны технические средства реализации приложения, а в этой части представлен прототип программного комплекса.

На Рис.1 показано окно приложение. Оно включается в себя меню, с помощью которого, пройдя по пути: Файл — Соединение с БД, можно подключится к базе данных. На самой форме приложения есть возможность вводы идентификационного номера пользователя и добавление данных о нём в первую таблицу. Аналогично с добавлением записей со стены и комментариев к любой из них.

Отдельное внимание можно уделить тому, что вывода записей каждого пользователя возможны два фильтра: количество записей, которое необходимо вывести, и дата, начиная с которой необходимо выводить записи.

Кнопки «Очистить таблицу» позволяют удалить все данные из предложенных таблиц.

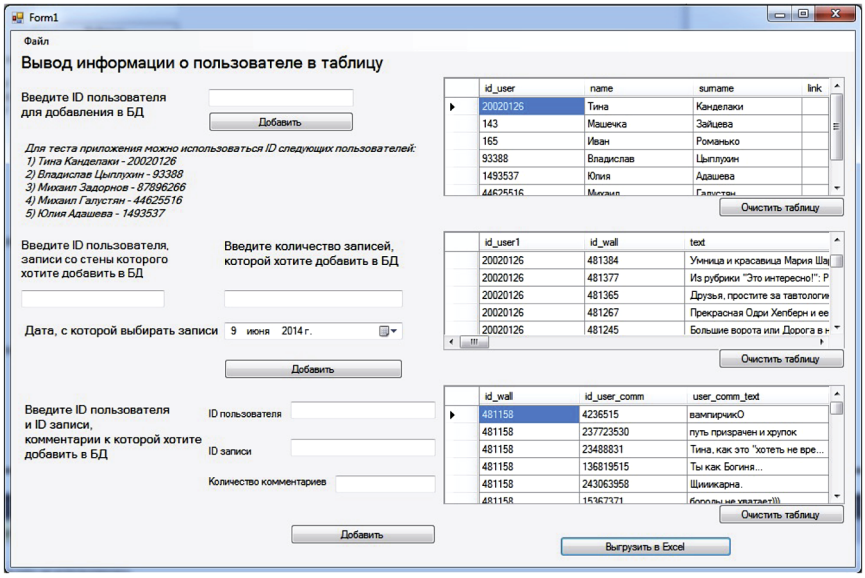


Рис. 1. Окно приложения

Отдельного рассмотрения заслуживает вывод полученных данных в таблицу Microsoft Excel, которая служит удобным способом представления данных (см. Рис. 2).

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	id_user	Имя	Фамилия	id записи	Текст записи	Комм	"Лайки"	Фото	Аудио	id_users_com	Текст комментария		
2	20020126	Тина	Канделаки	481546	Сердце не обманет)	9	71	http://cs		82325574	Наоборот сердца полюбит ,		
3				481535	Лицо Парижа.	8	215	http://cs		20020126	[id8010840|Илья], да) Но		
4	44625516	Михаил	Галустян	54033	И пэсня;) @chistyakova	14	172			118694946	Вообще Красавцы!!!		
5										118694946	И мне так привет передать		
6				54032	@chistyakova_ionova Наташка,	3	3						
7													
8													

Рис. 2. Таблица Microsoft Excel

Заключение

Описанный в предыдущих пунктах способ извлечения данных с помощью api-запросов позволяет нам быстро получать данные и выбирать те, которые мы хотим видеть в базе дынных. Реализация его в Microsoft Visual Studio позволяет создать удобное приложение, которое хорошо сочетается с базами данных Microsoft SQL Server.

Используемый метод извлечения данных направлен в первую очередь на автоматизацию работы социологов по получению информации для дальнейшей обработки и анализа.

Литература

1. Развитие интернета в регионах России [Электронный ресурс]. URL: http://company.yandex.ru/researches/reports/2014/ya_internet_regions_2014.xml
2. Исследование TNS Web Index [Электронный ресурс]. URL: http://habrahabr.ru/company/palitrumlab/blog/186422/
3. *Балуев Д. Г.* Политическая роль социальных медиа как поле научного исследования // Образовательные технологии и общество (Educational Technology & Society). 2013. Т. 16. № 2. С. 604–616.
4. *Benkler Yochai*. The Wealth of Networks. New Haven: Yale University Press, 2006.
5. *Шалимов А. Б.* Диалектика социального и индивидуального // Некоммерческое партнёрство «Проектно-аналитическое агенство «Шаг». 2013. С. 1030–1036.
6. *Варлыгина З. В.* Тенденции развития тематических социальных сетей в российском интернете. 2008. С. 220–227.
7. *Хансен Марк Д., Чоу Ричард Т.,Маури Кевин К., Смит Дуайт Р., Уорден Джеймс П.* Система и способ для управления доступом ненадежных приложений к защищённому контенту // Патент на изобретение. 2010.
8. Интерфейс программирования приложений [Электронный ресурс]. URL: http://ru.wikipedia.org/wiki/Api
9. *Э. Фримен*. Изучаем HTML, XHTML и CSS = Head First HTML with CSS & XHTML // СПб.: Питер, 2010. 656 с. 9. Работа с API. Авторизация [Электронный ресурс]. мURL: https://vk.com/dev/authentication
10. Список методов секции users: users.get [Электронный ресурс]. URL: https://vk.com/dev/users.get
11. *Moser K. D.* SQl SERVER, Apps bounds to tighten // Ziff Davis Media Inc. 1993.

СОДЕРЖАНИЕ

Нейроинформатика и мультиагентное управление 7

Памяти Адиля Васильевича Тимофеева 5

А. В. Тимофеев, Е. О. Путин. Классификатор для статического обнаружения компьютерных вирусов, основанный на машинном обучении 9

С. Э. Чернакова, А. И. Нечаев, А. А. Иванов. Интеллектуальные роботы-ассистенты на Земле и в Космосе 22

А. М. Бакурадзе. Принципы мультифрактального проектирования глобальных сетей нового поколения 26

А. А. Алексеев. Предварительная обработка видео данных для распознавания в системах управления движением 32

А. И. Кукушкин. Сравнительный анализ методов коллективного интеллекта в задачах кластеризации и оптимизации 34

Т. М. Косовская. Модель формирования логико-предикатной нейронной сети. 44

В. А. Перхуров. Разработка базовых программных модулей анализа рабочих движений оператора при обучении робота методом показа. 48

Системное программирование 53

Р. С. Одеров. Управление политиками контроля доступа на основе ролевой модели 55

С. А. Серко. Организация эффективного хранения образов виртуальных машин с возможностью их модификации для автозапуска приложений 59

А. И. Птахина. Эволюция языков при метамоделировании «на лету» в DSM-платформе QReal 62

Д. Е. Дерюгин. Профилирование операционных систем реального времени 72

А. А. Гудошникова. Обзор методов анализа предметной области . . . 79

Т. Ю. Агапова. Автоматическая миграция моделей 84

А. Ю. Кирсанов. Функциональное реактивное программирование роботов на базе платформы Трик 92

А. Ю. Коровянский, К. С. Амелин. Аппаратная возможность организации децентрализованной сети мобильных датчиков . . . 97

Фундаментальная информатика 103

Т. М. Косовская. Понятие неполной выводимости предикатной формулы и ее применение к решению задач искусственного интеллекта 105
В. А. Гошев. Особенности языка программирования рефал-5е 110
М. А. Герасимов. Квадратичный по времени алгоритм приближенного разбиения множества натуральных чисел с гарантированной оценкой точности 113
Д. М. Августинов. Исследование новостного потока методами частотного анализа. 117
Е. Д. Заболотский. Компьютерная реализация тематической классификации текстов методами частотного анализа 123
М. А. Старицын, С. В. Яхонтов. Эффективное по времени и памяти вычисление W-функции Ламберта 130
С. В. Яхонтов. Алгоритмическая вещественная функция, заданная на отрезке [0, 1], которая не является полиномиально вычислимой по времени. 132

Технологии и инструменты разработки программ и облачные вычисления 137

А. И. Михайлова. Аспектно-ориентированный рефакторинг CMS Orchard с помощью Aspect.NET 139
Д. А. Григорьев, А. В. Григорьева, В. О. Сафонов. Реализация механизма доступа к динамическому контексту в точках применения аспектов для системы Aspect.NET 142
А. К. Рагозина. Сравнение алгоритмов обобщенного восходящего и нисходящего синтаксического анализа. 148
М. Е. Стрельцова. Примеры бесшовной интеграции функциональных блоков MS Enterprise Library с использованием Aspect.NET 151
М. А. Зотов. Реализация надстройки MS VS 2012 для поддержки системы аспектно-ориентированного программирования Aspect.NET 157

Адаптивное управление и распознавание образов в условиях неопределенностей 165

В. А. Ерофеева. Поисковой алгоритм стохастической аппроксимации в задаче балансировки загрузки при неизвестных, но ограниченных возмущениях на входе. 167
В. К. Филатов. Алгоритм ориентирования сверхлёгкого БПЛА по данным бортового фото-видеорегистратора 175
Д. Г. Найданов, Р. Е. Шеин. Применение онтологий в МАС на примере алгоритма муравьиной колонии 181

В. Н. Калитеевский, И. Н. Калитеевский. Использование скрытых марковских моделей при разработке интерфейса взаимодействия с ПК при помощи движений головы 188
Е. В. Короткова. Метод дистанционного мониторинга санитарного состояния леса 194
В. Н. Шац. Рандомизация в задачах машинного обучения 200

Параллельные алгоритмы и вэйвлетная обработка числовых потоков 205

Б. Д. Воробьев. Моделирование гребенчатых структур вэйвлетов и их распараллеливание 207
Ю. К. Демьянович. Об архитектуре параллельной системы 214
В. С. Богданов, М. Ю. Быц. Компьютерная реализация вычислений некоторых элементарных функций 219
И. Д. Мирошниченко. О калибровочных соотношениях для $B\varphi$-сплайнов четвертого порядка 226
М. В. Парахин. О вэйвлетном разложении 229

Теория и практика кодирования информации 237

А. И. Веселов, В. А. Ястребов. Анализ применения разреженного кодирования в задаче восстановления регионов изображений . . . 239
Н. Д. Егоров, М. Р. Гилмутдинов. О задаче универсального кодирования источников без памяти 245
Д. В. Новиков. Метод регуляризации в задаче восстановления искаженных изображений 252
А. И. Акмалходжаев. Новый пассивный метод оценки качества голоса в сети 3GPP LTE 259
А. В. Борисовская, И. А. Пастушок. Алгоритм планирования ресурсов на базовой станции с учетом требований к качеству обслуживания пользователей 268
Е. С. Востокова. Криптографические системы основанные на спаривании 278

Распараллеливание в OPEN MP и сплайновые аппроксимации 279

И. Г. Бурова, Н. С. Домнин. О построении полиномиальных и тригонометрических аппроксимаций третьего порядка 281
И. Г. Бурова, Т. О. Евдокимова. Об аппроксимации разрывными интегро-дифференциальными сплайнами третьего порядка 285
О. В. Безрукавая. О сжатии и восстановлении изображения с помощью параметрически заданных сплайнов 291

Методы хранения, поиска и анализа информации 295

P. V. Fedotovsky, K. E. Cherednik, G. A. Chernishev. Supporting additional tree data structures in GiST 297
A. D. Dubatovka, B. A. Novikov. Creating sentiment dictionaries and analysis of goods reviews in Russian 302
M. A. Zarechensky. Text detection in natural scenes with multilingual text 308
О. Ю. Янюк. Управление данными об отходах потребления на примере города Петрозаводска 314

Кибернетика и робототехника 321

А. А. Лосенков, С. В. Арановский. Управление промышленным гидравлическим приводом на примере робототехнического крана-манипулятора, применяемого в лесозаготовительной промышленности 323
А. А. Лосенков, С. В. Арановский. Компенсация неизвестного мультисинусоидального возмущения прямым адаптивным методом на основе декомпозиции возмущения 330
А. С. Кавокин. Распознавание пола человека по фотографии 339
Т. В. Чугаева, С. Ю. Сартасов. Локальная оценка качества отпечатков пальцев 343
С. Г. Сарманова. Проектирование и реализация библиотеки времени исполнения виртуальной JAVA машины CLDC HI для кибернетического контроллера ТРИК 349
Е. С. Егорова. Моделирование одежды в реальном времени 355

Математическое и компьютерное моделирование 361

И. П. Манакова. Визуализатор моделей мультимедийных CDN 363
А. С. Лебедев, И. Н. Обабков, К. А. Яковлев. Моделирование процессов электролиза меди 370
Е. В. Шадрина, М. А. Назаров, С. Э. Грегер. Моделирование структуры и создание визуальных отображений системы управления учебным процессом 373
И. Б. Литус, А. А. Кукченко. Моделирование процесса патронирования унитарных выстрелов 378

Автоматное программирование, машинное обучение и биоинформатика 383

И. А. Петрова, А. С. Буздалова, М. В. Буздалов. Повышение эффективности эволюционных алгоритмов при помощи обучения с подкреплением в нестационарной среде 385

М. В. Буздалов, А. С. Буздалова. Сравнительный анализ метода выбора вспомогательных критериев и метода спуска со случайными мутациями 393
М. В. Буздалов, А. С. Буздалова. Асимптотически оптимальные алгоритмы для выбора вспомогательных критериев оптимизации 400
Д. С. Чивилихин, В. И. Ульянцев, А. А. Шалыто. Применение метода нарушения симметрии в алгоритмах построения управляющих конечных автоматов 405
Н. В. Ведерников, В. Ю. Демьянюк, П. В. Кротков, В. И. Ульянцев, А. А. Шалыто. Автоматизированное построение управляющих автоматов в среде STATEFLOW при помощи методов машинного обучения 409
И. Б. Сметанников, М. В. Буздалов. Разработка эффективного метода определения самопересечений белковой цепи. 416
В. О. Долганов. Разработка метода сборки транскриптома на основе анализа компонент связности графа де Брёйна 423
А. С. Крамар. Генетическое картирование по неполным и зашумленным данным. 429

Синтез элементов компьютерной архитектуры 433

Г. А. Леонов, Н. В. Кузнецов, К. Д. Александров. Двухфазная схема Костаса и гипотеза Беста 435

Математические методы и алгоритмы в системах хранения данных высокой производительности 441

Г. Крайчик. Программа восстановления параметров потока обращений к устройству хранения. 443
В. С. Зайберт, А. В. Маров. Разработка модуля восстановления утраченных дисков в RAID N+M в поле $GF(2^8)$ 450
И. И. Демьяненко. Распознавание мультимедиа-приложений по SCSI-запросам на стороне СХД 458
М. М. Заславский. Разработка имитационной модели производительности модульных систем хранения данных Active–Active 465
К. И. Тюшев. Мультиагентные технологии для построения RAID-подобных распределенных систем хранения данных 475
С. В. Морозов, В. Н. Калитеевский. Моделирование потока запросов на чтение и запись данных 480

Информационная безопасность. 487

А. А. Молдовян, И. И. Лившиц, А. Т. Танатарова. Подходы для реализации процесса поддержки принятия решения

для развития современной организации на основании статистики сертификации ISO 489
В. И. Емелин, А. В. Григорова. Информационная безопасность инвестиционных проектов в сфере недвижимости 491
Д. М. Латышев. Схема хэш-функции с потайным ходом 495
С. А. Рудакова. Методика оценки защищенности информационных активов 499
Р. С. Одеров. Управление политиками контроля доступа на основе RBAC модели 503
Р. Ш. Фахрутдинов. Методы усиления защиты при использовании существующей инфраструктуры связи 507
М. В. Баклановский, О. Н. Граничин, А. Р. Ханов. Новостная авторизация 512
Н. А. Молдовян, А. В. Муравьев, А. А. Горячев. Стойкие схемы шифрования с малой длиной ключа 516
М. В. Баклановский, Д. В. Луцив. BigHex: реализация RSA на JavaScript для веб-приложений 521
Р. Ш. Фахрутдинов. Использование низкоскоростных устройств для шифрования видеоданных 527

Вероятностные графические модели, нечеткие системы, мягкие вычисления и социокомпьютинг 531

А. В. Торопова. Развитие электронной публикации 533
А. А. Азаров, А. А. Фильченков, М. В. Абрамов. Анализ распространения вредоносного контента среди пользователей социальных медиа 537
Д. В. Степанов. Копульный подход к оценке относительных показателей риска 544
К. Д. Коромыслов. Применение экспертной системы на основе нейронной сети для прогнозирования потребления природного газа 550
А. М. Алексеев. Автоматизация анализа популярности технологических областей в корпусе текстов русскоязычных электронных медиа на основе данных Википедии 557
А. Е. Пащенко, Т. В. Тулупьева. Экспресс-анализ реплик и метаданных социальных сетей с использованием программных средств автоматизации получения данных 563
А. Л. Тулупьев. Алгебраические байесовские сети: открытые вопросы локального автоматиеского обучения 569
Е. В. Иванова, А. Е. Пащенко, А. Л. Тулупьев. Автоматизированное извлечение данных с пользовательской страницы в социальной сети, экспорт их в базу данных и в электронные таблицы 578

Научное издание

СПИСОК-2014

Материалы
всероссийской научной конференции
по проблемам информатики

23–25 апр. 2014 г.,
Санкт-Петербург

Компьютерная верстка: *О. В. Шакиров*

Издательство «ВВМ»

190000, Санкт-Петербург,
ул. Декабристов, 6, лит. А, пом. 10н
E-mail: vvmpub@yandex.ru

Подписано к печати 31.10.14. Формат $60 \times 90\ ^1/_{16}$. Бумага офсетная.
Гарнитура Таймс. Печать цифровая. Печ. л. 37,00. Тираж 300 экз.
Заказ 6106

Отпечатано в Отделе оперативной полиграфии
химического факультета СПбГУ

198504, Санкт-Петербург, Старый Петергоф, Университетский пр., 26
Тел.: (812) 428-4043, 428-6919