

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

На правах рукописи

Полухин Александр Леонидович

МЕТОДЫ ДОСТУПА К ХРОНОЛОГИЧЕСКИМ ДАННЫМ
В РЕЛЯЦИОННЫХ СИСТЕМАХ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ.

05.13.11 – Математическое и программное обеспечение
вычислительных машин, комплексов и компьютерных сетей.

А В Т О Р Е Ф Е Р А Т
диссертации на соискание ученой степени
кандидата физико-математических наук

Санкт-Петербург 2006

Работа выполнена на кафедре информатики математико-механического факультета Санкт-Петербургского государственного университета.

Научный руководитель:

доктор физико-математических наук, профессор Братчиков Игорь Леонидович

Официальные оппоненты:

доктор технических наук, профессор Копыльцов А.В.,

кандидат физико-математических наук, доцент Графеева Н.Г.

Ведущая организация:

Санкт-Петербургский институт информатики и автоматизации РАН.

Защита диссертации состоится “__” _____ 2006 года в __ часов на заседании диссертационного совета Д 212.232.51 по защите диссертаций на соискание учёной степени доктора наук при Санкт-Петербургском государственном университете по адресу: 198504, Санкт-Петербург, Старый Петергоф, Университетский пр., 28, математико-механический факультет, ауд. _____

С диссертацией можно ознакомиться в Научной библиотеке имени М.Горького Санкт-Петербургского государственного университета по адресу: 199034, Санкт-Петербург, Университетская наб., д. 7/9.

Автореферат разослан “__” апреля 2006 года.

Учёный секретарь

диссертационного совета,

доктор физико-математических наук

Мартыненко Б.К.

1. Общая характеристика работы

Актуальность темы диссертационной работы в первую очередь определяется её ориентацией на важную практическую проблему эффективного доступа к хронологическим данным в системах управления базами данных.

В настоящее время наиболее широко используются системы управления реляционными базами данных. Предлагаемые методы ориентированы на решение поставленной проблемы путем создания темпорального расширения реляционной модели данных.

Основой расширения модели является определение его базовых понятий. Такими понятиями для хронологических данных традиционно являются: размерность времени, метка времени, шкала времени и календарь. При этом в настоящий момент нет расширения, вводящего в реляционную модель эти понятия. Существующие модели, содержащие эти понятия, либо узко специализированы, либо существенно отличаются от реляционной, что не позволяет их использовать для работы с существующими базами.

Разработка темпорального расширения реляционной модели позволяет повысить эффективность и удобства работы с хронологическими данными. Построение методологии индексации таких данных должно способствовать полному снятию проблемы работы с хронологическими данными.

Анализ существующих исследований, посвященных решению задачи повышения качества работы с хронологическими данными, выявил крайне незначительное число готовых и апробированных решений, что во многом связано с отсутствием достаточно проработанной теории и практики решения задачи хронологического расширения реляционной модели данных. Эффективное решение описанной задачи и составляет суть диссертационной работы.

Объектом исследования являются размерности времени, промежутки времени, календари, операторы для работы с ними, алгоритмы построения R-деревьев.

Предметом исследования является темпоральное расширение реляционной модели данных и использование R-деревьев для поиска хронологических данных.

Целью работы является повышение скорости и удобства доступа к хронологическим данным в реляционных системах управления базами данных за счет построения расширения реляционной модели данных, учитывающего специфику хронологических данных, а также скорости доступа – за счет использования R-деревьев и разработки алгоритма построения R-деревьев с минимализацией перекрытий на неравномерных данных. Для достижения поставленной цели в диссертационной работе поставлены и решены следующие задачи:

1. Обоснование и выбор подходов для расширения реляционной модели данных хронологическими понятиями.
2. Выбор и реализация операторов для работы с темпоральным расширением.

3. Обоснование возможности использования R-деревьев для поиска хронологических данных.
4. Разработка алгоритма построения R-деревьев с минимализацией перекрытий на неравномерных данных.

Основные методы исследования. В качестве методов исследования использовались статистический анализ, теория множеств, анализ алгоритмов, реляционная алгебра. Компьютерная реализация выбранных операторов и алгоритма производилась на основе объектно-ориентированного подхода.

Научная новизна. Предлагаемая диссертация содержит следующие новые результаты, полученные лично автором:

1. Определены базовые понятия темпорального расширения реляционной модели данных.
2. Разработаны операторы для задания введенных базовых понятий и работы с ними.
3. Использованы R-деревья, для доступа к хронологическим данным, по отрезкам времени, представленным в виде двумерных отрезков.
4. Разработан метод построения R-деревьев с минимализацией перекрытий на неравномерных двумерных данных.

Теоретическая значимость работы заключается в создании расширения реляционной модели данных для работы со временем, которое может послужить платформой для усовершенствования существующих реляционных систем управления базами данных. Разработанный метод построения индекса с учетом времени позволяет эффективно обрабатывать хронологические данные, что дает возможность широко применять предложенное расширение при работе с ними. Кроме того, предложенный подход позволяет ввести прямо в базу используемые методы обработки хронологических данных.

Практическая значимость работы заключается в создании тестовой системы, реализующей предложенный алгоритм построения R-дерева и поиск по тестовым данным с его использованием. Предложенный подход может быть легко применен в реляционной системе управления базами данных, полная же разработка такой системы достаточно трудоемка и выходит за рамки данной работы.

Личный вклад автора. Все основные результаты диссертации получены автором самостоятельно.

Апробация работы. Научные результаты и основные положения работы представлялись на конференциях:

1. IX Санкт-Петербургская международная конференция «Региональная информатика-2004».
2. XXXVI межвузовская научная конференция аспирантов и студентов «Процессы управления и устойчивость».

Реализация. Полученные результаты реализованы в виде тестовой программной системы на языке программирования C#.

Публикации. Автором опубликовано по теме диссертации 3 печатные работы.

Структура и объем диссертационной работы. Диссертация состоит из введения, трех глав, заключения, она излагается на 100 страницах, включая перечень используемой литературы из 86 наименований. Кроме того, в диссертации имеется приложение на 10 листах, содержащее в себе примеры разработанных программ, реализующих алгоритмы, описанные в диссертации.

2. Содержание работы

Во введении содержится обоснование актуальности темы диссертации, сформулированы основные научные результаты, выносимые автором на защиту, а также практическая ценность полученных результатов.

В первой главе «Подходы к формализации модели данных. Опыт использования различных моделей данных в СУБД» обсуждается современное состояние подходов к решению проблемы организации и доступа к данным; проанализированы направления развития моделей данных, рассмотрены основные существующие модели и системы их реализующие.

Рассмотрены различные методы организации упорядоченных данных (последовательностей) и хронологических данных в частности.

Упорядоченные данные (последовательности) встречаются во многих финансовых, промышленных, научных и др. приложениях (например, наборы измерений, снимаемые с датчиков, временные ряды экономических показателей). В связи с этим возникает задача хранения и предварительной обработки последовательных данных в рамках СУБД. Реляционная модель данных оперирует понятием отношения - множества кортежей. В основе модели упорядоченных данных должна лежать последовательность, а не множество. Для упорядоченных данных требуется иметь специфический набор операторов, отличных от реляционных.

Реализация этих операторов с помощью реляционной модели неэффективна. Последовательность является способом описания некоторой функции. Поэтому требуется знать точность описания этой функции и область определения функции. Операторы манипулирования последовательностями должны учитывать эту информацию.

В связи с указанными сложностями, за последние 15 лет было предложено довольно много специализированных моделей данных для работы с последовательностями. Однако, как показано далее, все они не лишены недостатков. Одним из индикаторов актуальности создания универсальной СУБД для работы с последовательными данными является крайне высокая стоимость специализированных исторических СУБД (которые решают лишь часть проблем хранения и обработки последовательностей), таких как IndustrialSQL Server, PI Data Storage и др.

В главе определен ряд структурных и функциональных требований к модели, предназначенной для представления последовательностей.

В связи с тем, что в большинстве приложений, работающих с упорядоченными данными, наряду с последовательностями используются обычные реляционные данные, возникает задача интеграции модели

последовательных данных с реляционной моделью. Существуют следующие подходы к решению указанной проблемы, приводимые ниже.

1. *Отсутствие интеграции с реляционной моделью и реализация всей требуемой функциональности в рамках специализированной модели.* Основным недостатком такого подхода является необходимость повторения функциональности реляционной модели в рамках специализированной модели данных.
2. *Представление последовательности или группы последовательностей в виде отношения особого типа, для которого действуют все реляционные операторы.* Реализация такого подхода требует создания минимальной надстройки над реляционной моделью (по этому наиболее проста). Однако такой подход имеет ряд недостатков, а именно:
 - 2.1. сложность эффективной реализации операторов над последовательными данными;
 - 2.2. невозможность динамического добавления/удаления последовательностей в БД без использования DDL - операций (операций описания данных, сокращение от Data Definition Language);
 - 2.3. невозможность выборки нескольких последовательностей на основе некоторого критерия (к примеру, выборки последовательных показаний датчиков, установленных в одном помещении).
3. *Представление последовательностей в реляционной модели в виде значений атрибутов абстрактного типа.* Такое представление последовательностей лишено всех недостатков предыдущего подхода и часто используется для реализации моделей последовательных данных на базе объектно-реляционных СУБД. Однако при реализации последовательности в виде абстрактного типа данных все операции над последовательностями реализуются в виде функций, вызываемых из тела запроса на языке SQL. Поскольку оптимизатор SQL ничего не знает о реализации таких функций, оптимизация сложных выражений над последовательными данными невозможна.
4. *Представление последовательностей в реляционной модели в виде значений атрибутов расширенного абстрактного типа.* При таком подходе для абстрактного типа данных, реализующего последовательности, имеется только одна функция, принимающая в качестве аргумента запрос на некотором языке, созданном для последовательностей. Эта функция оптимизирует запрос и возвращает результирующую последовательность, используемую в основном SQL-запросе. Тем не менее, и такой подход не лишен недостатков. Представим себе последовательность A , содержащую коды некоторых ошибок, соответствующие последовательным меткам времени. При этом подробная информация о каждом коде ошибки может содержаться в обычном отношении B . Для получения последовательности, содержащей информацию об ошибках, необходимо выполнить реляционный оператор соединения последовательности A и отношения B . Если

последовательность моделируется с помощью атрибута абстрактного типа, то такая операция невыполнима.

5. *Расширение понятия «отношения» за счет добавления к нему последовательностей в качестве третьего измерения и расширение всех реляционных операторов для манипулирования отношениями нового типа.* Такой подход достаточно труден в реализации, однако лишен всех недостатков, свойственных предыдущим подходам. Используем именно такой подход. В полноценных многомерных БД именно такой подход естественен для работы с последовательностями, реляционную же модель потребуется расширить.

Таким образом, делается вывод о целесообразности применения последнего из рассмотренных подходов.

Во второй главе «Описание используемого способа расширения реляционной модели данных» описано хронологическое расширение реляционной модели данных и примеры его использования. Ниже приводится его сокращенное описание.

Время рассматривается как интервал целых чисел, на котором задан линейный порядок – отношение «<» на множестве чисел. Для поддержки различных размерностей времени рассматривается не один такой интервал, а набор интервалов, элементами каждого из которых являются значения времени какой-либо одной размерности (например, «минуты», «дни» и др.). Для того чтобы иметь возможность сравнивать значения времени разной размерности, пары значений времени разных размерностей должны быть связаны отношениями «является частью». Было дано формальное определение размерности времени.

Размерность времени μ – это либо интервал целых чисел T , называемый *интервалом размерности* (обозначается $T(\mu)$), либо тройка $(T, \eta, F_{\eta \rightarrow \mu})$, где T – интервал целых чисел, называемый *интервалом размерности*, η – некоторая другая размерность, $F_{\eta \rightarrow \mu}$ – отношение «является частью» из интервала размерности η в интервал T , обладающее следующими свойствами:

- $F_{\eta \rightarrow \mu}$ – это сюръективная функция;
- $\forall i \in T(\eta), \forall j \in T(\mu) : (i < j) \& (i, j \in \text{dom}(F_{\eta \rightarrow \mu})) \Rightarrow F_{\eta \rightarrow \mu}(i) \leq F_{\eta \rightarrow \mu}(j)$,

где $\text{dom}(F_{\eta \rightarrow \mu})$ – область определения $F_{\eta \rightarrow \mu}$.

В случае если размерность μ описывается тройкой $(T, \eta, F_{\eta \rightarrow \mu})$, будем говорить, что размерность μ задана через размерность η . Было также определено понятие метки времени.

Метка времени – это пара (τ, μ) , где μ – это размерность времени, а τ – целое число из интервала $T(\mu)$, называемое *значением времени*.

Отношение "является частью" предназначено для задания новых размерностей времени на базе существующих. Кроме того, оно служит для преобразования меток времени одной размерности в метки времени другой размерности.

Пусть задана размерность $\mu = (T, \eta, F_{\eta \rightarrow \mu})$. Если для сюръекции $F_{\eta \rightarrow \mu}$ множество значений $\{\tau_1, \dots, \tau_k\}$ является прообразом для значения τ , то будем говорить, что (τ, μ) *состоит из* $\{(\tau_1, \eta), \dots, (\tau_k, \eta)\}$. Будем записывать это следующим образом: $(\tau, \mu) \sim \{(\tau_1, \eta), \dots, (\tau_k, \eta)\}$. Если при этом размерность η задана через размерность ν , и каждая из меток (τ_i, η) состоит из $\{(\tau_{i1}, \nu), \dots, (\tau_{ik_i}, \nu)\}$, то также будем говорить, что (τ, μ) состоит из $\bigcup_{i=1..k} \{(\tau_{i1}, \nu), \dots, (\tau_{ik_i}, \nu)\}$, а размерность μ задана (*транзитивно*) через размерность ν . Факт того, что размерность μ задана через размерность η (напрямую или транзитивно) будем обозначать следующим образом: $\eta \rightarrow \mu$. Будем также считать, что для любой размерности μ имеет место $\mu \rightarrow \mu$, и для любых меток времени (τ, μ) выполняется $(\tau, \mu) \sim \{(\tau, \mu)\}$.

Предполагается, что на основе имеющегося набора стандартных размерностей времени («секунды», «минуты», «часы», «дни», «недели» и т.д.) пользователь или администратор базы данных будет определять новые, специфичные для его задач, размерности, такие как «двухчасовки», «получасовки», «смены», «кварталы» и др. Причем задание новых размерностей времени возможно на базе любых других уже существующих. Пользователю необходимо лишь задать отношение «является частью», либо воспользоваться специальными **операторами для задания размерностей времени**.

Потребовалось ввести понятие сравнимости размерностей времени и различные отношения для сравнения меток времени сравнимых размерностей, а также отношения «агрегируется в» и «точнее чем», адаптированные для нашей модели.

Размерности времени μ и η называются *сравнимыми*, если существует размерность ν (*общая размерность*), что μ и η заданы через ν (напрямую или транзитивно).

Пусть μ и η – сравнимые размерности времени, ν – их общая размерность (произвольная), т.е. $\nu \rightarrow \mu$, $\nu \rightarrow \eta$, и пусть $(\tau_1, \mu) \sim M$, $(\tau_2, \eta) \sim H$, где M и H – некоторые множества меток времени размерности ν . Тогда:

- если для любой общей размерности времени ν выполняется условие $M \subseteq H$, то будем говорить, что (τ_1, μ) *входит в* (τ_2, η) , и записывать это следующим образом: $(\tau_1, \mu) \subseteq (\tau_2, \eta)$;
- если для любой общей размерности времени ν выполняется следующее условие: $\forall (\tau_M, \nu) \in M, \forall (\tau_H, \nu) \in H : \tau_M < \tau_H$, то будем говорить, что (τ_1, μ) *раньше чем* (τ_2, η) , а (τ_2, η) *позже чем* (τ_1, μ) , и записывать это следующим образом: $(\tau_1, \mu) < (\tau_2, \eta)$;
- если для любой общей размерности времени ν выполняется $M \cap H \neq \emptyset$, то будем говорить, что (τ_1, μ) *пересекается с* (τ_2, η) , и записывать это следующим образом: $(\tau_1, \mu) \cap (\tau_2, \eta) \neq \emptyset$.

С практической точки зрения важна задача нахождения всех меток

времени размерности η , которые входят в, или пересекаются с заданной меткой времени (τ, μ) (это называется преобразованием размерностей времени). Например, если есть хронологическая последовательность с метками времени размерности *час*, и на ее базе необходимо сформировать хронологическую последовательность со сводной информацией с метками времени размерности *смена*, то для выполнения такой операции необходимо уметь находить все метки времени размерности *час*, относящиеся к каждой смене.

Будем говорить, что размерность времени η «агрегируется в» размерность μ , ($\mu \leq \eta$), если они сравнимы, и для любых размерностей времени ν таких, что $\nu \rightarrow \mu$ и $\nu \rightarrow \eta$, выполнено следующее условие:

$$\forall (\tau, \eta), \exists (\tau_i, \mu) \sim \{(\tau_{i1}, \nu), \dots, (\tau_{ik}, \nu)\}, i = 1 \dots k:$$

$$(\tau, \eta) \sim \bigcup_{i=1 \dots k} \{(\tau_{i1}, \nu), \dots, (\tau_{ik}, \nu)\} \quad ,$$

причем метки времени $(\tau_i, \mu), i = 1 \dots k$, зависят только от (τ, η) и не зависят от выбора ν .

Будем говорить, что размерность времени μ «точнее чем» размерность η ($\mu \leq \eta$), если они сравнимы, и выполнено следующее условие: $\forall (\tau, \mu), \exists (\tau', \eta) : (\tau, \mu) \subseteq (\tau', \eta)$.

Мы определили метку времени как целое число с указанием размерности. Такое представление, несомненно, является удобным для хранения меток времени в базе данных и для выполнения вычислений, но не является ни привычным, ни удобным для пользователя. В реальной жизни люди пользуются метками времени, представленными по структурным (относительным) шкалам, таким как «годы:месяцы:сутки» и др. Поэтому мы определим понятие шкалы времени и способ представления любой метки времени по любой шкале.

Введем оператор, возвращающий первую метку времени размерности η , пересекающуюся с заданной меткой времени (τ, μ) (обозначим этот оператор $first((\tau, \mu), \eta)$), следующим образом:

- если размерности μ и η сравнимы, то $first((\tau, \mu), \eta) = \tau'_{\min}$, где $\tau'_{\min} = \min\{\tau' : (\tau, \mu) \cap (\tau', \eta) \neq \emptyset\}$, т.е. это минимальное значение из таких τ' , что метка времени (τ', η) пересекается с (τ, μ) ; а если при этом таких τ' нет, то оператор $first((\tau, \mu), \eta)$ не определен;

- если μ и η несравнимы, то для них оператор $first((\tau, \mu), \eta)$ не определен.

Аналогично определим оператор $last((\tau, \mu), \eta)$, возвращающий последнюю метку времени размерности η , пересекающуюся с заданной меткой времени (τ, μ) .

Пусть $\mu_i, i = 1 \dots n$, – различные сравнимые размерности времени. Тогда запись вида $\mu_1 : \dots : \mu_n$ будем называть *шкалой времени*.

Пусть размерность времени η сравнима с $\mu_i, i = 1 \dots n$. Будем называть *представлением* метки времени (τ, η) по шкале $\mu_1 : \dots : \mu_n$ запись вида $(\tau_1 : \dots : \tau_n, \eta)$, где:

- $\tau_1 = first((\tau, \eta), \mu_1)$;
- $\tau_{i+1} = first((\tau, \eta), \mu_{i+1}) - first((first((\tau, \eta), \mu_i), \mu_i), \mu_{i+1}), i=1 \dots n-1$.

Если при этом результат хотя бы одного используемого в данном определении оператора *first* не определен, то будем считать, что метка времени (τ, η) не представима по шкале $\mu_1 : \dots : \mu_n$.

Было также сформулировано достаточное условие однозначности представления меток времени по шкале.

Пусть задана шкала времени $\mu_1 : \dots : \mu_n$ и размерность времени η . Пусть при этом выполняется условие $\mu_k \leq \eta$ или $\mu \leq \eta$. Тогда, для любой метки времени (τ, η) , если существует представление по шкале $\mu_1 : \dots : \mu_n$, то оно однозначно определяет эту метку времени.

Данное утверждение служит базой для проверки корректности представления меток времени по шкале.

У шкал имеется два основных назначения:

1. представление пользователю меток времени, хранящихся в базе данных, в удобном для него виде;
2. предоставление пользователю возможности самому задавать интересующие метки времени в удобном для него виде, а также задавать новые размерности времени и календари.

При задании меток времени через представление по шкале важно, чтобы представление однозначно определяло метку времени. Поэтому потребуем, чтобы при задании меток времени было разрешено пользоваться только шкалами, удовлетворяющими достаточному условию однозначности.

При задании метки времени (τ, η) через ее представление $(\tau_1 : \dots : \tau_n, \eta)$ по шкале $\mu_1 : \dots : \mu_n$ кроме явного указания элементов τ_i , будем также использовать обозначение *last*. Данное обозначение, встречающееся на месте τ_{i_p} , обозначает следующее:

$$last((first((\tau, \eta), \mu_{i-1}), \mu_{i-1}), \mu_i) - first((first((\tau, \eta), \mu_{i-1}), \mu_{i-1}), \mu_i).$$

Например, если задана шкала *годы:недели*, то представление $(2003:last, недели)$ определяет последнюю неделю 2003 года.

Последним из базовых хронологических понятий введено понятие календаря.

Множество меток времени одной размерности будем называть *календарем*.

Календари мы будем использовать при выборке данных из хронологической последовательности для указания интересующих нас моментов времени. Календари, требуемые для выборки данных, зачастую поддаются компактному описанию. Для компактного описания календарей вводятся специализированные операторы. Кроме того, на основе существующих календарей можно получать новые с помощью теоретико-множественных операций « \cup », « \cap » и « \setminus », но лишь в том случае, когда операнды содержат метки времени одной размерности. Далее вводятся операторы для задания новых размерностей времени.

Отношение «является частью» $F_{\eta \rightarrow \mu}$ при определении новых размерностей времени можно задавать следующим образом: внутри диапазона значений времени $T(\eta)$ выделить непересекающиеся непустые подмножества, пронумеровать эти подмножества подряд идущими целыми числами, и считать, что $F_{\eta \rightarrow \mu}$ каждому значению времени размерности η , принадлежащему такому подмножеству, ставит в соответствие значение времени размерности μ , являющееся номером этого подмножества.

В этом случае $F_{\eta \rightarrow \mu}$ полностью определяется двумя свойствами:

- набором подмножеств множества значений времени размерности η ;
- нумерацией этих подмножеств.

Задав эти два свойства, мы определяем отношение "является частью". Именно таким образом определяется это отношение для размерностей времени.

Таким образом, в данной главе был предложен новый математический аппарат для описания размерностей времени, шкал и календарей. Предложена хронологическая модель данных, являющаяся расширением реляционной модели. Она учитывает специфику БД, предназначенных для промышленных ИС, позволяя формулировать запросы, используя календари для выборки данных, и взаимосвязи между различными размерностями времени для агрегирования.

В третьей главе «Методы индексирования хронологических данных» исследуется возможность представления хронологических данных в виде пространственных и, как следствие, применение к ним соответствующих методов индексации.

Основное направление исследований в области управления пространственными данными - это структуры данных для хранения и представления пространственных данных. Результаты этих исследований были внедрены в ряде географических информационных систем (GIS - geographical information system). Наиболее популярны среди известных структур данных R-деревья, которые используются для представления неточечных объектов (в частности прямоугольных областей). Серьезная проблема, связанная с R-деревьями, заключается в том, что они допускают перекрытие ограничивающих прямоугольников. Поскольку каждый объект, находящийся в одной из этих перекрывающихся областей, принадлежит только одному ограничивающему прямоугольнику, то в общем случае требуется просмотр множества ветвей дерева индексации. Для R-деревьев были предложены некоторые оптимизирующие методы.

R-дерево представляет собой разновидность сильноветвящегося дерева. Внутренние узлы дерева представляют собой набор записей вида $(cp, Rect)$, где cp является указателем на узел-потомок, а $Rect$ - минимальным прямоугольником, который покрывает все прямоугольники узла-потомка. Лист дерева представляет собой набор записей вида $(Oid, Rect)$, где Oid является ссылкой на запись в базе данных, которая описывает геометрический объект, а $Rect$ является минимальным прямоугольником, который покрывает данный

геометрический объект. Пусть M есть максимальное число записей узла, а m является минимальным числом записей узла ($2 \leq m \leq M/2$). Тогда для R-дерева должны выполняться следующие условия:

- корень дерева имеет не менее двух потомков, если он не является листом;
- внутренний узел имеет не менее m и не более M потомков, если он не является корнем;
- каждый лист имеет не менее m и не более M записей, если он не является корнем;
- все листья находятся на одном и том же уровне.

Практическое сравнение эффективности различных вариантов R-деревьев проведено в ряде работ. Наиболее быстродействующими из них являются R*-деревья, поэтому в дальнейшем в данной работе использована именно эта разновидность R-деревьев.

Индексная структура R-дерева является сбалансированным по высоте деревом с индексными записями в листьях, содержащими указатели на объекты данных. Узлам дерева соответствуют дисковые страницы, если индекс располагается на диске. Структура является полностью динамической, вставка и удаление объектов может свободно перемежаться с запросами на поиск, периодической реорганизации данных при этом не требуется.

Данная индексная структура предназначена для эффективного выполнения регионального поиска, поэтому основная идея дерева заключается в разбиении пространства в каждом узле дерева на минимально пересекающиеся группы, что позволяет при выполнении поиска эффективно отсекал неверные ветви работы алгоритма. Задача разбиения в общем случае, видимо, по меньшей мере является NP-полной, так как в настоящее время даже для двух групп не известно полиномиального алгоритма. Поэтому во всех алгоритмах построения R-деревьев используются эвристические подходы, дающие на реальных данных далеко не оптимальные разбиения.

Высокая степень перекрытия входов в узлах дерева происходит из-за динамичности алгоритмов его построения, когда при вставке очередного элемента его необходимо за минимальное время разместить и при необходимости перестроить дерево. При этом глобальная оптимизация дерева на каждом шаге не производится.

На практике взаимодействие с индексными структурами состоит из двух основных этапов: начального построения дерева и последующей оперативной работы. При этом во многих случаях во время второго этапа производится только поиск объектов без вставки новых и удаления старых.

С учетом специфики начального построения структуры, строится вначале более эффективное R-дерево, а на последующих этапах используются обычные алгоритмы для работы с R-деревьями. В частном случае, когда вначале ничего не строится, получается обычное R-дерево.

Во всех существующих структурах R-деревьев основным параметром, определяющим скорость поиска, считается степень взаимного пересечения потомков в узлах дерева. Именно поэтому в алгоритмах построения R-деревьев

одной из главных целей является минимизация такого пересечения. Это позволяет при выполнении регионального поиска на ранних стадиях эффективно отсекал тупиковые ветви работы алгоритма.

Построенная для заданного набора объектов структура R-дерева предложено называть *оптимальной*, если сумма площадей попарных перекрытий входов во всех нелистовых узлах дерева является минимальной среди всех возможных структур R-деревьев:

$$\min_{R\text{-деревья}} \sum_{i \neq j} S(R_i \cap R_j)$$

, где S – функция площади.

Предлагается следующая стратегия построения R-дерева, близкого к оптимальному.

Алгоритм построения дерева

По заданному множеству из N объектов E_i строим R-дерево с параметрами m, M [1].

Шаг 1. Начало. Построим корень дерева и определим количество уровней дерева по формуле $L(N) = \lceil \log_M N \rceil$.

Шаг 2. Построение дерева. Вызовем *алгоритм построения узла*, передав ему в качестве параметра корень дерева и всё множество объектов.

Алгоритм построения узла

По заданному множеству из N объектов E_i и узлу T строим потомков узла, имеющего уровень L ($L=1$ для листьев).

Шаг 1. Построение листьев. Если $L=1$, то заполняем узел объектами E_i и заканчиваем.

Шаг 2. Выбор количества потомков узла. Вычисляем количество потомков у данного узла по формуле $K = \lceil N^{1/L} \rceil$.

Шаг 3. Построение узлов. Вызываем *алгоритм разделения на группы* для разбиения всего множества объектов на K групп, минимизируя сумму попарных пересечений групп. При этом в алгоритм передаем номер уровня L для правильной оценки минимального и максимального допустимого количества объектов в группах. Для каждой полученной группы строим пустой узел – потомок текущего узла и вызываем для него рекурсивно *алгоритм построения узла* с этой группой в качестве множества.

Наиболее сложной частью предложенной стратегии является *алгоритм разделения на группы*, от трудоёмкости которого зависит общая сложность алгоритма. Предлагается вариант данного алгоритма.

Алгоритм разбиения множества объектов на минимально пересекающиеся группы

Во все существующие алгоритмы построения R-деревьев входят алгоритмы разбиения множества объектов на две части с минимальным пересечением. Рассмотрим предложенный вариант алгоритма разбиения, в котором производится последовательное разбиение всего множества на две части до тех пор, пока всё множество не окажется разбитым на требуемое число частей.

Неравномерный алгоритм деления на группы

Заданное множество из N объектов E_i делим на K групп с соблюдением ограничения на количество объектов в группах снизу m^{L-1} и сверху M^{L-1} , где L – текущий строящийся уровень дерева.

Шаг 1. Разделение на две группы. Вызываем алгоритм деления на две группы, передав в качестве аргументов всё множество объектов E_i и соотношение деления $(\lfloor K/2 \rfloor) : (K - \lfloor K/2 \rfloor)$ для получения групп G_1 и G_2 .

Шаг 2. Рекурсивное разделение. Если $\lfloor K/2 \rfloor > 1$, то вызываем алгоритм деления на группы, передав в качестве аргументов полученную группу G_1 с параметром деления $\lfloor K/2 \rfloor$, иначе выдаем в качестве результата G_1 . Если $K - \lfloor K/2 \rfloor > 1$, то вызываем алгоритм деления на группы, передав в качестве аргументов полученную группу G_2 с параметром деления $K - \lfloor K/2 \rfloor$, иначе выдаем в качестве результата G_2 .

Алгоритм деления на две группы

Заданное множество из N объектов E_i делим на 2 группы в заданном соотношении площадей $k:l$ с соблюдением ограничения на количество объектов снизу $k \cdot m^{L-1}$ и сверху $k \cdot M^{L-1}$ для первой группы, а также снизу $l \cdot m^{L-1}$ и сверху $l \cdot M^{L-1}$ для второй, где L – текущий строящийся уровень дерева.

Шаг 1. Выбор оси координат для сортировки. Выберем такую ось координат i , на которой достигается максимальное значение выражения

$$\max_{\text{оси } i} \left(\frac{\max_{\text{фигуры } j} R_a^{j,i} - \min_{\text{фигуры } j} R_b^{j,i}}{\max_{\text{фигуры } j} R_b^{j,i} - \min_{\text{фигуры } j} R_a^{j,i}} \right),$$

где $R_a^{j,i}$ и $R_b^{j,i}$ – соответственно меньшее и большее значение i -й координаты минимального объемлющего j -ю фигуру прямоугольника.

Шаг 2. Разделение по группам. Разделим все объекты на три части по значению $P^i(j) = (R_a^{j,i} + R_b^{j,i})/2$ для выбранного i по квантилям уровня α и $1-\alpha$ ($\alpha \in [0;0,5]$ – параметр деления по группам): из наименьшей по значениям части образуем первую группу объектов, а из наибольшей – вторую группу. Оставшиеся объекты распределим по группам в шагах 3 и 4.

Шаг 3. Проверка завершения. Если все объекты распределены, то заканчиваем. Если одна из групп в результате дальнейших операций будет недополнена (количество объектов в группах $p_1 < k \cdot m^{L-1}$ или $p_2 < l \cdot m^{L-1}$), то вставим в неё все оставшиеся объекты и закончим. Если одна из групп в результате дальнейших операций будет переполнена (количество объектов в группах $p_1 > k \cdot M^{L-1}$ или $p_2 > l \cdot M^{L-1}$), то вставим в другую группу все оставшиеся объекты и закончим.

Шаг 4. Распределение объектов. Берем любой нераспределённый объект и помещаем его в группу, размер которой увеличится в минимальной степени при взвешивании по k и l . Переходим на шаг 3.

На втором шаге работы алгоритма деления присутствует параметр α деления по группам. Для установления его влияния на качество получаемого

разбиения было проведено экспериментальное моделирование работы глобального алгоритма построения R-дерева, использующего алгоритм разбиения «Разделяй и властвуй». При этом было установлено, что уменьшение значения α приводит к некоторому улучшению структуры R-дерева на неравномерных распределениях и распределениях со значительным перекрытием объектов. С другой стороны, это приводит к снижению скорости работы алгоритма и уменьшению качества разбиения на точечных и регулярных наборах данных.

Суммируя достоинства и недостатки поведения алгоритма при разных значениях α на различных распределениях объектов, эмпирически было выбрано значение параметра разделения для универсального применения, равное 0,45. При этом если на практике имеются некоторые сведения о реальном распределении объектов, то значение параметра α , вероятно, стоит оценить дополнительно в зависимости от требований конкретной задачи.

Сравнение экспериментальных данных подтверждает правильность выбранной стратегии на минимизацию взаимных перекрытий потомков узлов в R-дереве, так как практически везде меньшему проценту перекрытия потомков соответствует больший процент попадания в нужные узлы при поиске.

В заключении содержится перечень задач, которые были решены в результате диссертационного исследования, а также сведения об апробации и внедрении результатов работы.

3. Основные результаты работы

В представленной работе для достижения поставленных задач решены следующие вопросы.

1. Изучены существующие и перспективные методы организации доступа к данным, в частности хронологическим.
2. Разработано специальное расширение реляционной модели для работы с хронологическими данными.
3. Разработан набор операторов для работы с предложенным расширением.
4. Предложен метод использования для индексации хронологических данных подходов, применяемых для пространственных объектов.
5. Предложен алгоритм построения R-деревьев на неравномерных данных с минимализацией перекрытий.
6. Проведена оценка трудоемкости предложенного алгоритма.

4. Публикации автора по теме диссертации:

Основные результаты диссертации опубликованы в следующих работах:

- [1] Полухин А.Л. Методы доступа к пространственным данным. — IX Санкт-Петербургская международная конференция «Региональная информатика-2004». Материалы конференции., СПб, 2005, стр.75-81.
- [2] Полухин А.Л. Метод темпорального расширения реляционной модели данных. — XXXVI межвузовская научная конференция аспирантов и студентов «Процессы управления и устойчивость». Материалы конференции., СПб, 2005.
- [3] Полухин А.Л. Операторы темпорального расширения реляционной модели данных. // Вестник С.-Петербур. ун-та. Сер.10. 2006. Вып.1. С.123-132.