САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

На правах рукописи

Куликов Александр Сергеевич

ПОСТРОЕНИЕ АЛГОРИТМОВ ДЛЯ ЗАДАЧ БУЛЕВОЙ ЛОГИКИ ПРИ ПОМОЩИ АВТОМАТИЗАЦИИ, КОМБИНИРОВАННЫХ МЕР СЛОЖНОСТИ И ЗАПОМИНАНИЯ ДИЗЪЮНКТОВ

05.13.17 — теоретические основы информатики

А В Т О Р Е Ф Е Р А Т

диссертации на соискание ученой степени
кандидата физико-математических наук

Санкт-Петербург — 2009

Работа выполнена в Учреждении Российской академии наук
Санкт-Петербургском отделении Математического института
им. В. А. Стеклова РАН

Научный руководитель:	кандидат физико-математических наук, доцент Гирш Эдуард Алексеевич
Официальные оппоненты:	доктор физико-математических наук, профессор Гимади Эдуард Хайрутдинович (Институт математики им. С. Л. Соболева Сибирского отделения РАН)
	кандидат физико-математических наук, доцент Соловьев Игорь Павлович (Санкт-Петербургский государственный университет)
Ведущая организация:	Уральский государственный университет

Защита диссертации состоится “ ” 2009 г. в час. на заседании совета Д 212.232.51 по защите докторских и кандидатских диссертаций при Санкт-Петербургском государственном университете по адресу: 198504, Санкт-Петербург, Старый Петергоф, Университетский пр., 28.

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

Автореферат разослан “ ” 2009 г.

Ученый секретарь
диссертационного совета,
доктор физ.-мат. наук,
профессор Даугавет И. К.

Общая характеристика работы

Актуальность темы. Интерес к доказательству экспоненциальных верхних оценок для NP-трудных задач в последние несколько десятилетий остается на стабильно высоком уровне. Одним из наиболее хорошо изученных подходов к доказательству таких оценок является метод расщепления. Впервые данный метод был предложен в 1960-м году Дэвисом и Патнемом и сформулирован в более современном виде Дэвисом, Лоджеманном и Лавлэндом в 1962-м году. Его основная идея заключается в расщеплении входного примера задачи на несколько более простых примеров (упрощаемых в дальнейшем некоторыми правилами упрощения), таких что, построив решение для каждого из них, возможно за полиномиальное время построить решение для исходного примера. В списке ниже мы приводим некоторые верхние оценки на время решения NP-трудных задач в наихудшем случае, доказанные методом расщепления и являющиеся наилучшими из известных (здесь и на протяжении всей работы мы опускаем полиномиальные от размера входа множители в оценках, указывая только экспоненциальную составляющую):

1. 1.074^L для задачи пропозициональной выполнимости формул в КНФ, где L — длина (то есть общее количество литералов) входной формулы (Гирш, 2000);

2. 1.341294^K для задачи максимальной выполнимости, где K — количество дизъюнктов входной формулы (Чен и Канж, 2002);

3. 1.122463^m для задачи о максимальном разрезе, где m — количество ребер входного графа (Скотт и Соркин, 2003);

4. 1.3289^n для задачи о 3-раскрашиваемости графа, где n — количество вершин входного графа (Бейгель и Эпштейн, 2005).

В диссертации рассматриваются алгоритмы расщепления в применении к задачам выполнимости и максимальной выполнимости. Обе зада-

чи формулируются на языке булевых формул, являющемся очень удобным для кодирования многих алгоритмических задач (таких, например, как автоматическое доказательство теорем, составление расписаний, оптимизационные задачи на графах). Задача пропозициональной выполнимости (satisfiability problem, SAT) является одной из наиболее известных NP-полных задач. Данной задаче посвящена международная ежегодная конференция (The International Conference on Theory and Applications of Satisfiability Testing), проводящаяся уже более десяти лет, а также научный журнал (Journal on Satisfiability, Boolean Modeling and Computation). Поскольку NP-трудные задачи часто возникают в практических приложениях (например, при разработке микросхем, в распознавании образов), важное место в исследовании задачи выполнимости занимает разработка программ, решающих SAT (такие программы называются SAT-солверами). Ежегодно проводятся соревнования таких программ. Современные SAT-солверы способны быстро решать многие задачи, считавшиеся нерешаемыми несколько лет назад.

Важным оптимизационным обобщением задачи выполнимости является задача максимальной выполнимости (maximum satisfiability problem, MAX-SAT). В терминах задачи MAX-SAT могут быть естественным образом переформулированы многие оптимизационные NP-трудные задачи, к примеру, такие оптимизационные задачи на графах, как задача о максимальном разрезе (maximum cut problem, MAX-CUT), задача о минимальном вершинном покрытии (minumum vertex cover problem), задача о максимальном независимом множестве (maximum independent set problem). Задача MAX-SAT возникает также в задачах искусственного интеллекта и комбинаторной оптимизации.

Для задачи MAX-SAT существуют приближенные алгоритмы, находящие за полиномиальное время решение с некоторой гарантированной точностью. Например, алгоритм Асано и др. выдает набор, выполняющий хотя бы долю 0.77 количества дизъюнктов, выполненных оптимальным набором. В то же время известно, что в предположении P$\neq$NP не существует

полиномиального по времени алгоритма, находящего приближенное решение, сколь угодно близкое к оптимальному. Алгоритм Данцина и др. находит такое приближение, но за экспоненциальное время. Известно также много алгоритмов, решающих практические примеры задачи максимальной выполнимости. Для данных алгоритмов, однако, неизвестно никаких оценок (кроме тривиальных) на время работы в наихудшем случае.

Наряду с общей задачей максимальной выполнимости мы рассматриваем следующие ее частные случаи:

- $(n, 3)$-MAX-SAT — вариант задачи MAX-SAT для формул, в которых каждая переменная содержится не более, чем в трех дизъюнктах;
- MAX-2-SAT — вариант задачи MAX-SAT, где каждый дизъюнкт входной формулы содержит не более двух литералов;
- $(n, 3)$-MAX-2-SAT — вариант задачи MAX-SAT, где каждый дизъюнкт входной формулы содержит не более двух литералов и каждая переменная входит не более, чем в три дизъюнкта.

Известно, что все перечисленные выше задачи являются NP-трудными. Таким образом, в предположении P$\neq$NP не существует полиномиальных по времени алгоритмов для данных задач. Тем не менее, поскольку подобные задачи часто возникают в практических приложениях, важно понять, какое время требуется для их решения, даже если это время экспоненциально.

Как сказано выше, лучшие известные оценки для многих NP-трудных задач получены именно методом расщепления. Простейший алгоритм расщепления для задачи выполнимости на формуле с N переменными имеет время работы порядка 2^N. Первые улучшения данной оценки были получены в начале 1980-х годов Мониеном и Шпекенмейером и Данциным для формул, длины дизъюнктов которых ограничены некоторой константой. Позднее было получено много оценок, улучшающих тривиальную для некоторых NP-полных подклассов задач выполнимости и максимальной

выполнимости. Вопрос о том, могут ли данные две задачи быть решены за время c^N, где $c < 2$ — константа, до сих пор остается открытым и является одним из самых знаменитых вопросов современной теоретической информатики. Однако недавно были разработаны алгоритмы, решающие задачи выполнимости и максимальной выполнимости быстрее, чем за 2^N, для формул константной плотности, то есть формул, у которых отношение количества дизъюнктов к количеству переменных ограничено сверху некоторой константой (Арвинд и Шулер, 2003; Данцин и Вольперт, 2006).

Типичный алгоритм расщепления сначала некоторым образом расщепляет формулу, после чего производит рекурсивные вызовы для формул меньшей сложности. Анализ такого алгоритма содержит разбор большого количества случаев, в каждом из которых показывается, что алгоритм всегда производит рекурсивные вызовы для формул, сложность которых меньше сложности исходной формулы хотя бы на некоторую константу. Для улучшения оценки на время работы алгоритма можно добавлять в алгоритм новые правила упрощения либо же проводить более детальный разбор случаев.

Цели работы.

1. Разработать и реализовать программу, осуществляющую автоматический анализ случаев, возникающих при доказательстве верхних оценок на время работы в худшем случае алгоритмов расщепления.

2. Разработать правило упрощения, обобщающее известные правила.

3. Построить алгоритм, решающий задачу MAX-SAT на формулах константной плотности Δ в худшем случае за время c^N, где $c = c(\Delta) < 2$ — константа, с использованием лишь полиномиальной памяти.

4. Доказать новые верхние оценки для задач MAX-2-SAT и $(n, 3)$-MAX-2-SAT.

Общая методика работы. Для оценки времени работы алгоритмов используются стандартные методы решения рекуррентных неравенств. Од-

нако почти во всех доказательствах верхних оценок на время работы алгоритмов используются нестандартные меры сложности. Также приводятся новые методы сокращения перебора случаев и описывается метод автоматического анализа алгоритмов расщепления.

Основные результаты.

1. Реализована программа, автоматически доказывающая верхние оценки на время работы алгоритмов расщепления путем анализа случаев. При помощи данной программы получено несколько новых оценок.

2. Разработано правило упрощения для задач выполнимости и максимальной выполнимости, обобщающее известные правила упрощения, присваивающие значения переменным формулы.

3. Разработан алгоритм, решающий задачу MAX-SAT на формулах константной плотности Δ за время c^N, где $c = c(\Delta) < 2$ — константа, с использованием лишь полиномиальной памяти.

4. Доказаны следующие новые верхние оценки на время работы алгоритмов расщепления:

   - $2^{K/6}$, где K — количество дизъюнктов входной формулы, для MAX-2-SAT;
   - $2^{N/6.7}$, где N — количество дизъюнктов входной формулы, для $(n, 3)$-MAX-2-SAT.

Научная новизна. Алгоритмы для $(n, 3)$-MAX-SAT и MAX-2-SAT являются самыми быстрыми из известных. Алгоритм для MAX-SAT, работающий на формулах константной плотности быстрее, чем 2^N, является первым известным алгоритмом для данной задачи, улучшающим тривиальную оценку и при этом пользующимся лишь полиномиальной памятью. Реализованная программа для автоматического анализа алгоритмов расщепления и обобщенное правило упрощения являются первыми в своем роде.

Практическая и теоретическая ценность. Новые идеи доказательства верхних оценок на время работы алгоритмов расщепления имеют, скорее, теоретическую ценность. В то же время идеи, использующиеся для сокращения перебора случаев, могут быть применены и на практике (к примеру, в SAT-солверах).

Апробация работы. Основные результаты обсуждались на следующих конференциях и семинарах: Международная студенческая школа Estonian Winter School in Computer Science (EWSCS 2004), Эстония, 2004; Международная студенческая школа Summer School on Experimental Algorithmics, Дания, 2004; Международная конференция International Workshop on Parameterized and Exact Computation (IWPEC 2004), Норвегия, 2004; Международная конференция Eighth International Conference on Theory and Applications of Satisfiability Testing (SAT 2005), Шотландия, 2005; Международный семинар Dagstuhl Seminar on Exact Algorithms and Fixed-Parameter Tractability, Германия, 2005; Международный симпозиум Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA 2006), США, 2006; Международный симпозиум Second International Computer Science Symposium in Russia (CSR 2007), Россия, 2007; Международный семинар Dagstuhl Seminar on Moderately Exponential Algorithms, Германия, 2008; Результаты, лежащие в основе диссертации, также были доложены в Университете Торонто (Univeristy of Toronto), Университете Нью-Йорка (City University of New York) и Чешской академии наук (Czech Academy of Sciences), а также на семинаре ПОМИ РАН. Доклад на студенческой школе в Эстонии занял первое место.

Публикации. Основные результаты диссертации опубликованы в семи работах [1, 3, 4, 6, 5, 7, 2]. Работы [1, 3, 4, 2] опубликованы в изданиях, входящих в список рекомендованных Высшей аттестационной комиссией на момент публикации. В работах [7, 2] диссертанту принадлежит новая идея использования нестандартной меры сложности для оценки времени работы алгоритмов, доказательства получены совместно с соавтором К. Куцковым. Описанные в работах [3, 4] алгоритмы и результаты принадлежат

диссертанту. В работе [5] диссертантом доказана основная оценка на время работы алгоритма, сам же алгоритм разработан совместно с соавтором А. Кожевниковым.

Структура и объем диссертации. Диссертация объемом 94 страницы состоит из введения и пяти основных глав, разбитых на разделы и подразделы. Список цитируемой литературы состоит из сорока наименований.

Содержание работы

Во **введении** обсуждаются рассматриваемые в диссертации задачи, состояние исследований в этой области, формулируются основные результаты, описывается структура диссертации.

В **первой главе** определяются основные понятия и вводятся обозначения, используемые на протяжении всей диссертации.

Формулы в КНФ. Пусть V — множество булевых переменных, то есть переменных, принимающих значения True и False (константы True и False обозначаются также 1 и 0, соответственно). Отрицание переменной $v \in V$ обозначается через $\neg v$. *Литералом* называется переменная или ее отрицание. Через $var(l)$ мы обозначаем переменную, соответствующую литералу l. *Дизъюнктом* и *набором* называются множества литералов, не содержащие одновременно переменной и ее отрицания. *Длиной дизъюнкта* называется количество содержащихся в нем литералов, k-дизъюнкт — это дизъюнкт длины ровно k. *Формулой в конъюнктивной нормальной форме (КНФ)* называется мультимножество дизъюнктов. Формулой в k-КНФ называется формула, содержащая только дизъюнкты длины не более k. *Плотностью* формулы называется отношение количества дизъюнктов к количеству переменных. Мы считаем, что формула может содержать также специальный дизъюнкт $\mathcal{T}$, который выполняется любым набором.

Полный набор — это набор значений всем переменным формулы. Мы говорим, что дизъюнкт C *выполняется* набором α, если $C \cap \alpha \neq \emptyset$; дизъюнкт C *опровергается* набором α, если $\forall l \in C$, $\neg l \in \alpha$. С помощью $V(\alpha)$

мы обозначаем множество переменных набора α. Через $\mathrm{Cl}(F, \alpha)$ обозначается количество дизъюнктов формулы F, выполненных набором α, а через $\mathrm{MCl}(F)$ — максимальное количество одновременно выполнимых дизъюнктов (таким образом, $\mathrm{MCl}(F) = \max_\alpha \mathrm{Cl}(F, \alpha)$).

Пусть F — формула в КНФ, v — переменная F, α — набор переменным F. Наборы $\{v\}$ и $\{\neg v\}$ мы будем записывать просто как v и $\neg v$. Через $F[\alpha]$ мы обозначаем формулу, полученную из F заменой каждого выполненного набором α дизъюнкта на дизъюнкт $\mathcal{T}$ и удалением всех вхождений литералов l, таких что $\neg l \in \alpha$, из оставшихся дизъюнктов.

Задачи выполнимости и максимальной выполнимости. Задача SAT заключается в проверке существования набора, выполняющего все дизъюнкты данной формулы F в КНФ, а задача MAX-SAT — в нахождении набора, выполняющего $\mathrm{MCl}(F)$ дизъюнктов. Под решением для SAT мы подразумеваем выполняющий набор, в то время как решением для MAX-SAT мы называем набор, выполняющий максимальное количество дизъюнктов.

Приведем два очень простых примера. Формула $F_1 = (x \vee y) \wedge (\neg x) \wedge (x \vee \neg y)$ содержит две переменные и три дизъюнкта и является невыполнимой. Набор $\{x, \neg y\}$ выполняет $\mathrm{MCl}(F_1) = 2$ дизъюнкта формулы F_1. Формула же $F_2 = (x \vee y) \wedge (x \vee \neg y)$, очевидно, выполнима.

Анализ алгоритмов расщепления. Получив на вход некоторую формулу F, типичный алгоритм расщепления сначала некоторым образом упрощает ее, потратив на это полиномиальное от размера F время, после чего производит рекурсивные вызовы для нескольких формул вида $F[\alpha_1], \ldots, F[\alpha_r]$, где $\alpha_1, \ldots, \alpha_r$ суть наборы значений переменным F. Таким образом, работу алгоритма расщепления можно рассматривать как дерево, вершины которого помечены формулами в КНФ. Каждой формуле F такого дерева мы приписываем неотрицательное вещественное число $\mu(F)$, обозначающее *сложность* F. Стандартными мерами сложности формул являются следующие: $\mu(F) = N(F)$ — количество переменных F; $\mu(F) = K(F)$ — количество дизъюнктов F; $\mu(F) = L(F)$ — длина (то

есть общее количество литералов) F. Дерево является *деревом расщепления*, если сложность формулы каждой вершины строго больше сложности каждой из формул, которыми помечены сыновья этой вершины.

Рассмотрим вершину дерева расщепления, помеченную формулой F_0. Пусть ее сыновья помечены формулами $F_1, F_2, \ldots, F_r$. Вектор $(t_1, t_2, \ldots, t_r) \in \mathbb{R}^r_{>0}$ называется *вектором расщепления* данной вершины, если $t_i \geq \mu(F_0) - \mu(F_i)$ для всех $i = 1, \ldots, r$. Характеристический многочлен данного вектора расщепления определяется следующим образом: $h(x) = 1 - \sum_{i=1}^{r} x^{-t_i}$. Единственный положительный корень этого многочлена называется *числом расщепления* и обозначается через $\tau(t_1, t_2, \ldots, t_r)$ (несложно видеть, что при положительных x функция $h(x)$ монотонно возрастает от $-\infty$ к 1). *Числом расщепления дерева* называется максимальное из чисел расщепления всех его вершин.

Следующая лемма доказана Кульманном и Люкхардтом.

Лемма 1. *Количество листьев в дереве расщепления, корень которого помечен формулой F и число расщепления которого равно τ, не превосходит $\tau^{\mu(F)}$.*

Как было упомянуто, алгоритм расщепления сначала упрощает входную формулу, после чего производит несколько рекурсивных вызовов на формулах меньшей сложности. Понятно, что общее время работы такого алгоритма складывается из времени работы всех рекурсивных вызовов, а также времени, потраченного на то, чтобы произвести эти вызовы. Таким образом, с точностью до полинома от размера входной формулы время работы алгоритма есть количество вершин (или листьев) соответствующего дерева расщепления.

Во **второй главе** последовательно перечислены основные идеи новых алгоритмов, представленных в работе. Сами же алгоритмы и доказательства оценок на время их работы представлены в последующих главах. В начале главы приведен простой алгоритм расщепления, а также два способа его улучшения.

В **третьей главе** приводится общее правило упрощения, обобщающее многие известные правила упрощения для рассматриваемых в диссертации задач. С использованием данного правила приводится простое доказательство следующей теоремы.

Теорема 1. *Существует алгоритм, решающий задачу* $(n, 3)$-MAX-SAT *за время* 1.272021^N *в наихудшем случае.*

Данное правило также используется в программе автоматического анализа алгоритмов, описанной далее.

Рассмотрим правило *доминирующий единичный дизъюнкт* для задачи MAX-SAT, введенное впервые Нидермайером и Росмани: если формула F содержит литерал l, такой что F содержит хотя бы столько же единичных дизъюнктов (l), сколько и дизъюнктов, содержащих литерал $\neg l$, то заменить F на $F[l]$. Корректность такой замены очевидна. Действительно, для любого полного набора α переменным формулы F, содержащего литерал $\neg l$, набор $\alpha \setminus \{\neg l\} \cup \{l\}$ выполняет хотя бы столько же дизъюнктов. Поэтому при поиске набора, выполняющего максимальное количество дизъюнктов, нет смысла рассматривать наборы, содержащие $\neg l$. В такой ситуации мы говорим, что $\{l\}$ сильнее $\{\neg l\}$. Мы обобщаем данную идею и представляем некоторое общее правило упрощения, содержащее в качестве частных случаев многие известные правила.

В **четвертой главе** представляется программа для автоматического анализа сложности алгоритмов расщепления, подробно описываются входные и выходные данные программы, ее структура, приводятся детали реализации и результаты экспериментов.

Анализ алгоритма расщепления состоит из длинного списка случаев. В каждом случае показывается, что соответствующее число расщепления не превосходит требуемой константы. Разработанная программа осуществляет такой разбор случаев автоматически. Данной программой были доказаны несколько нетривиальных оценок для задач SAT и MAX-SAT. Некоторые из них до сих пор остаются наилучшими из известных.

Пятая глава посвящена идеям использования нестандартных мер сложности и запоминания дизъюнктов. С использованием данных идей доказываются две следующие теоремы.

Теорема 2. *Для любой константы* $\Delta > 0$ *найдутся константы* $D > 0$ *и* $c < 2$*, такие что* MaxSatAlg(D) *выдает* $\mathrm{MCl}(F)$ *для любой формулы* F *с не более чем* $\Delta N(F)$ *дизъюнктами за время* $c^{N(F)}$*.*

Алгоритм 1 MaxSatAlg

Вход: КНФ формула F

Параметры: положительное целое число D

Метод:

1. Присвоить значение True всем чистым литералам формулы F.
2. Если F содержит только дизъюнкты $\mathcal{T}$, вернуть их количество.
3. Если F содержит D^+-литерал a, то произвести рекурсивные вызовы для $F[a]$ и $F[\neg a]$ и вернуть максимум из полученных ответов.
4. Пусть a — произвольный литерал F. Пусть $i = d(a)$, $j = d(\neg a)$.
5. Построить оптимальные наборы для F_a и $F_{\neg a}$. Обозначим их через $\alpha_a = \{x_1, \ldots, x_p\}$ и $\alpha_{\neg a} = \{y_1, \ldots, y_q\}$, соответственно, и пусть
$$k_a = \mathrm{MCl}(F_a) = \mathrm{Cl}(F, \alpha_a)\,, k_{\neg a} = \mathrm{MCl}(F_{\neg a}) = \mathrm{Cl}(F, \alpha_{\neg a})\ .$$
6. Если $i + k_{\neg a} \geq j + k_a$, то произвести рекурсивные вызовы для
$$F[a], F[\neg a, \neg y_1], F[\neg a, y_1, \neg y_2], \ldots, F[\neg a, y_1, \ldots, y_{q-1}, \neg y_q]$$
и вернуть максимум из полученных ответов.
7. В противном случае произвести рекурсивные вызовы для
$$F[\neg a], F[a, \neg x_1], F[a, x_1, \neg x_2], \ldots, F[a, x_1, \ldots, x_{p-1}, \neg x_p]$$
и вернуть максимум из полученных ответов.

Теорема 3. *Существуют алгоритмы, решающие задачи* MAX-2-SAT *и* $(n, 3)$-MAX-2-SAT *за время* $2^{K/6}$ *и* $2^{N/6.7}$*, соответственно, в наихудшем случае.*

Для доказательства верхних оценок на время работы алгоритма расщепления сперва фиксируется некоторая *мера сложности формул*, после чего показывается, что алгоритм всегда расщепляет входную формулу на несколько формул, сложности которых меньше сложности исходной формулы. Естественными мерами сложности формул являются количество переменных N, количество дизъюнктов K и длина формулы L. Известно большое количество верхних оценок относительно этих мер для задач SAT и MAX-SAT и их частных случаев. В данной работе мы используем комбинированные меры сложности для доказательства новых оценок для MAX-SAT и MAX-2-SAT. Наша мера сложности для задачи MAX-SAT выглядит следующим образом: $\gamma(F) = N(F) + wK(F)$, где w — некоторая константа. Мы показываем, что для любой формулы найдется достаточно хорошее расщепление либо относительно количества переменных, либо относительно количества дизъюнктов. После этого мы подбираем константу w, зависящую от плотности формулы, так, чтобы соответствующая оценка была меньше 2^N.

Идея запоминания дизъюнктов используется во многих современных SAT-солверах. Программа запоминает найденные частичные наборы, делающие формулу невыполнимой. Например, набор $\{x, \neg y\}$ делает невыполнимой формулу $(\neg y \vee z) \wedge (z \vee \neg x) \wedge (x) \wedge (\neg z \vee y)$. Для того, чтобы учесть этот факт, программа добавляет к рассматриваемой формуле дизъюнкт $(\neg x \vee y)$. Сама по себе идея довольно естественна и не является новой: она используется, например, в теоретических алгоритмах, практических программах и даже в системах доказательств. Мы приводим обобщение этой простой идеи для задачи MAX-SAT.

Список литературы

[1] *Куликов А. С.* Верхняя оценка $O(2^{0.16254n})$ для точной 3-выполнимости: более простое доказательство // Записки научных семинаров ПОМИ. 2002. Т. 293. С. 118–128.

[2] *Куликов А. С., Куцков К.* Новые верхние оценки для задачи максимальной выполнимости // Дискретная математика. 2009. Т. 21, № 1. С. 139–157.

[3] *Куликов А. С., Федин С. С.* Решение задачи о максимальном разрезе за время $2^{|E|/4}$ // Записки научных семинаров ПОМИ. 2002. Т. 293. С. 129–138.

[4] *Куликов А. С., Федин С. С.* Автоматические доказательства верхних оценок на время работы алгоритмов расщепления // Записки научных семинаров ПОМИ. 2004. Т. 316. С. 111–128.

[5] *Kojevnikov A., Kulikov A. S.* A New Approach to Proving Upper Bounds for MAX-2-SAT // Proceedings of the Seventeenth Annual ACM-SIAM Symposium on Discrete Algorithms. 2006. P. 11–17.

[6] *Kulikov A. S.* Automated Generation of Simplification Rules for SAT and MAXSAT // Proceedings of the Eighth International Conference on Theory and Applications of Satisfiability Testing. Vol. 3569 of *Lecture Notes in Computer Science*. 2005. P. 430–436.

[7] *Kulikov A. S., Kutzkov K.* New Bounds for MAX-SAT by Clause Learning // Proceedings of the 2nd International Computer Science Symposium in Russia. Vol. 4649 of *Lecture Notes in Computer Science*. 2007. P. 194–204.