САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ
АКАДЕМИЧЕСКИЙ УНИВЕРСИТЕТ РАН

На правах рукописи

Опарин Всеволод Владиславович

Оценки сложности вывода в системах доказательств, основанных на методе резолюций

01.01.06 — математическая логика, алгебра и теория чисел

Диссертация
на соискание ученой степени
кандидата физико-математических наук

Научный руководитель:
к.ф.-м.н. Д. М. Ицыксон

Санкт-Петербург
2016

Оглавление

Введение

Актуальность темы

Теория сложности доказательств — одна из активно развиваемых областей математической логики. В рамках теории изучаются формальные системы доказательств.

Согласно теореме Кука–Рекхоу [1], классы **NP** и **co-NP** равны тогда и только тогда, когда существует система доказательств, которая для каждой тавтологии ϕ имеет доказательство размера полиномиального от длины ϕ. Программа Кука состоит в том, чтобы рассматривать все более и более сильные системы доказательств и получать нижние суперполиномиальные оценки на размеры доказательств. В идеале это могло бы привести к разделению классов **P** и **NP**.

Резолюционная система доказательств (Res) — одна из наиболее изученных систем. Впервые система была введена Блэком в 1938 г. [2] и позднее популяризована в работах Дэвиса и Путнама в 1960 г. [3] и Робинсона в 1965 г. [4].

Помимо программы Кука, резолюционная система доказательств интересна в рамках автоматизированного поиска доказательств и алгоритмов для задачи выполнимости пропозициональной формулы. В главной роли здесь выступают DPLL алгоритмы [3, 5] и их более современные версии, CDCL алгоритмы [6, 7, 8].

Известно, что оптимальный протокол работы DPLL алгоритма на невыполнимой формуле совпадает с минимальным древовидным резолюционным доказательством [9]. При анализе CDCL алгоритмов, протокол работы сравнивают с резолюционными системами доказательств в

общем виде [10, 11, 12]. Известно, что и DPLL, и CDCL алгоритмы вкладываются в резолюционные системы доказательств [10]. Потому трудные формулы для резолюционных систем доказательств оказываются также трудны для DPLL и CDCL алгоритмов.

Резолюционные системы доказательств

Резолюционная система доказательств предназначена для языка `UNSAT` — языка невыполнимых пропозициональных формул, заданных в конъюнктивной нормальной форме (КНФ). Кук и Левин показали, что любую пропозициональную формулу можно эффективно закодировать в КНФ [13, 14]. Поскольку любая невыполнимая формула является отрицанием тавтологии, задачи распознавания `TAUT` и `UNSAT` можно считать эквивалентными. В таком контексте термин “опровержение” (refutation) часто заменяется термином “доказательство” (proof).

Резолюционная система доказательств устроена следующим образом. Доказательство формулы ϕ — это последовательность дизъюнктов, каждый из которых является либо дизъюнктом формулы ϕ, либо выводится по правилу резолюции или правилу ослабления из предыдущих дизъюнктов. Правило резолюции принимает на вход два дизъюнкта: $x \vee C_1$ и $\neg x \vee C_2$ — и выводит дизъюнкт $C_1 \vee C_2$:

$$\frac{x \vee C_1 \quad \neg x \vee C_2}{C_1 \vee C_2}.$$

Правило ослабления принимает дизъюнкт C и выводит дизъюнкт $C \vee x^\sigma$, где x^σ — литерал x или $\neg x$, и переменная x не лежит в C:

$$\frac{C}{C \vee x^\sigma}.$$

Последний дизъюнкт доказательства — пустой: тождественная ложь.

Правила вывода гарантируют семантическое следствие: если подстановка выполняет все посылки, то она же выполняет и следствие. Поскольку пустой дизъюнкт выполнить нельзя, то нельзя выполнить все дизъюнкты формулы одновременно.

Структуру доказательства можно описать в виде ориентированного ациклического графа. Если граф оказывается деревом, доказательство называют древовидным (см. Рисунок 1).

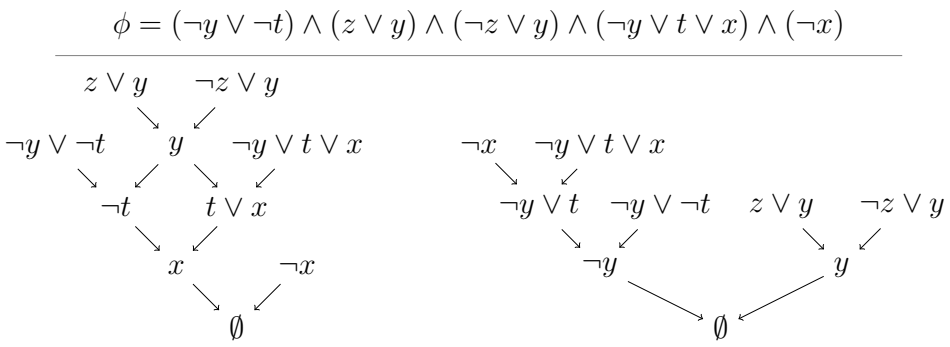

Рис. 1: Примеры резолюционных доказательств общего вида (слева) и древовидного (справа).

Степень разработанности темы

В 1968 г. Цейтин получил первые суперполиномиальные нижние оценки на размер регулярных резолюционных доказательств [15]. Цейтин использовал специальные формулы, кодирующие, что сумма степеней вершин графа нечетна. Формулы позднее были названы цейтинскими. В 1987 г. Уркухарт показал для цейтинских формул экспоненциальные нижние оценки на резолюционные доказательства в общем виде [16].

Первые экспоненциальные нижние оценки для резолюций в общем виде получил Хакен в 1985 г. [17]. В качестве трудных формул использовался принцип Дирихле. Формула $\mathtt{PHP}_n^m$ использует $n \cdot m$ переменных и кодирует, что можно посадить m кроликов в n клеток так, что ни одна клетка не будет содержать более одного кролика и каждый кролик будет где-то сидеть. При $m > n$ формула $\mathtt{PHP}_n^m$ невыполнима. Хакен показал, что размер минимального резолюционного доказательства для формулы $\mathtt{PHP}_{n-1}^n$ не меньше $2^{\Omega(n)}$. В 1988 Басс и Туран показали нижнюю оценку $2^{\Omega(n^2/m)}$ для формулы $\mathtt{PHP}_n^m$ [18]. В 1999 г. Басс и Питасси показали, что формула $\mathtt{PHP}_n^m$ имеет доказательство размера $2^{\Omega(n)}$ при

$m = 2^{\sqrt{n \log n}}$. Также Басс и Питасси показали, что для любых $m > n$ формула $\mathtt{PHP}_n^m$ имеет резолюционное доказательство размера $2^{O(n)}$, что показало точность уже полученных нижних оценок [19].

Оценки для принципа паросочетания

В 2004 г. Разборов рассмотрел формулу для принципа совершенного паросочетания $\mathtt{PMP}_G$. Формула $\mathtt{PMP}_G$ выполнима тогда и только тогда, когда в графе G есть совершенное паросочетание.

Разборов показал, что формула $\mathtt{PMP}_G$, построенная на графе G без совершенного паросочетания, имеет минимальное резолюционное доказательство размера $2^{\Omega(\delta(G)/\log^2 n)}$, где n — число вершин в графе G, а $\delta(G)$ — минимальная степень графа [20]. Данчев и Риис в 2001 г. [21] и Алекнович в 2004 г. [22] рассмотрели задачу замощения домино шахматной доски с двумя выколотыми угловыми клетками. Для доски размера $2n \times 2n$ соответствующие формулы содержат $\Theta(n^2)$ переменных и эквивалентны $\mathtt{PMP}_G$. Авторы показали нижнюю оценку $2^{\Omega(n)}$ на размер резолюционного доказательства в общем виде.

В 2015 Ицыксон, Слабодкин и Соколов показали, что если взять специальный двудольный граф с n и $m = O(n)$ вершинами в долях при $m > n$, и степень каждой вершины будет ограничена константой, то любое резолюционное доказательство формулы $\mathtt{PMP}_G$ будет иметь размер хотя бы $2^{\Omega(n)}$ [23]. Как следствие, авторы получили нижнюю оценку $2^{\Omega(n)}$ для двудольной клики $K_{n,m}$ и клики K_{2n+1}.

Для произвольного графа G на n вершинах известна тривиальная верхняя оценка $2^{O(n \log n)}$ для формул $\mathtt{PMP}_G$.

Вопрос 1. Является ли нижняя оценка $2^{\Omega(n)}$ для формулы $\mathtt{PMP}_G$ точной или ее можно улучшить?

Оценки на линейные расщепления

Некоторые из трудных примеров для DPLL алгоритмов по сути являются системами линейных уравнений, например, цейтинские формулы [15].

Кажется естественным разрешить алгоритмам оперировать с линейными уравнениями.

В 2013 г. Сето и Тамаки предложили алгоритм для задачи выполнимости пропозициональной формулы [24]. Алгоритм решает задачу выполнимости пропозициональной формулы, зависящей от n переменных, длины $c \cdot n$ и использующей все возможными бинарные операции, за время $2^{(1-\mu_c)\cdot n}$, где константа μ_c зависит только c.

Одной из основных идей была дополнительная поддержка системы линейных уравнений над полем $\mathbb{F}_2$. Алгоритм похож на DPLL, но может находить в формуле подформулу, которая считает сумму переменных по модулю два, и расщепляться по возможным значениям этой суммы. Соответствующая подформула заменяется на константу, а в систему добавляется уравнение с соответствующей суммой.

В 2014 г. Ицыксон и Соколов рассмотрели расширение DPLL алгоритма — алгоритм линейных расщеплений [25]. Пусть есть пропозициональная формула ϕ. Обычный DPLL алгоритм выбирает одиночную переменную x и смотрит, выполнима ли формула ϕ при условии, что переменная x равна нулю или единице. Обобщение выбирает сразу множество индексов переменных I и смотрит, выполнима ли формула, если сумма переменных по модулю два $\bigoplus_{i \in I} x_i$ равна нулю или единице.

Можно сказать, что алгоритм поддерживает некоторую систему линейных уравнений Ψ над полем $\mathbb{F}_2$, изначально пустую, и добавляет после каждого расщепления очередное уравнение. Алгоритм гарантирует, что система Ψ всегда имеет хотя бы одно решение и для каждого дизъюнкта C формулы ϕ существует хотя бы одно решение системы Ψ, которое выполняет дизъюнкт C (оба условия проверяются за полиномиальное время). Если система Ψ имеет ровно одно решение ρ, то ρ возвращается как выполняющий набор.

В соответствие алгоритму Ицыксон и Соколов сопоставили систему доказательств Res-Lin. Вместо обычных дизъюнктов система оперирует дизъюнкциями линейных уравнений по модулю два. Соответствующее

правило резолюции выглядит следующим образом:

$$\frac{\bigoplus_{i\in I} x_i = 0 \vee C_1 \quad \bigoplus_{i\in I} x_i = 1 \vee C_2}{C_1 \vee C_2}.$$

Сами линейные уравнения появляются в результате правила ослабления:

$$\frac{C}{D},$$

где дизъюнкция линейных уравнений D семантически следует из C.

В своей работе авторы показали, что древовидное доказательство в системе Res-Lin эквивалентно протоколу работы алгоритма линейных расщеплений. Размер минимального доказательств совпадает с оптимальным временем работы алгоритма линейных расщеплений с точностью до константы.

Формулы, кодирующие системы линейных уравнений по модулю два, например, цейтинские, являются тривиальными для алгоритма линейных расщеплений. Ицыксон и Соколов показали, что принцип совершенного паросочетания для графов на нечетном числе вершин также может быть решен за полиномиальное время.

В качестве трудного примера можно рассмотреть принцип Дирихле. В [25] была получена экспоненциальная нижняя оценка $2^{\Omega(n)}$ на размер древовидного доказательства для $\mathtt{PHP}_n^{n+1}$ в системе Res-Lin. В 1999 г. Ивама и Миязаки показали, что минимальное древовидное доказательство в системе Res для $\mathtt{PHP}_n^{n+1}$ имеет размер $2^{\Theta(n \log n)}$ [26].

Вопрос 2. Является ли нижняя оценка $2^{\Omega(n)}$ на размер древовидного доказательства формулы $\mathtt{PHP}_n^m$ в системе Res-Lin точной или ее можно улучшить?

Задача выполнения ограничений

Задача выполнения ограничений (constraint satisfaction problem, CSP) обобщает задачу выполнимости пропозициональной формулы. CSP состоит из множества переменных X, множества значений D, *алфавита*,

размера k, и множества предикатов S. Каждый предикат зависит от подмножества переменных из X. В отличии от пропозиционального случая, переменные могут принимать не два, а k значений. Задача удовлетворения ограничений состоит в поиске подстановки $\rho : X \to D$, которая выполняет все ограничения из S.

По аналогии с `SAT` существуют алгоритмы с возвратом для поиска выполняющего набора. В 2002 г. Митчелл рассмотрел резолюционные системы доказательств NG-Res для CSP [27]. Система NG-Res оперирует особым типом ограничений: запрещающим набором, который имеет вид

$$\neg(x_1 = \sigma_1 \wedge x_2 = \sigma_2 \wedge \cdots \wedge x_t = \sigma_t),$$

где $\sigma_i \in D$ для $i \in [t]$. Мы говорим, что запрещающий набор N соответствует ограничению f, если множества переменных, от которых зависят N и f, совпадают, и набор N семантически следует из f (т.е. если подстановка ρ нарушает N, то она же нарушает f).

Доказательство в NG-Res — это последовательность запрещающих наборов, каждый из которых либо соответствует ограничению исходной CSP, либо является резолюцией или ослаблением предыдущих. Последний набор пустой.

Правило резолюции в NG-Res принимает на вход k запрещающих наборов с общей переменной x вида $\neg(x = a_i \wedge \alpha_i)$ для всех $a_i \in D$ и выглядит следующим образом:

$$\frac{\neg(x = a_1 \wedge \alpha_1) \quad \cdots \quad \neg(x = a_k \wedge \alpha_k)}{\neg(\alpha_1 \wedge \alpha_2 \wedge \cdots \wedge \alpha_k)}.$$

Правило ослабления имеет вид

$$\frac{\neg(\alpha)}{\neg(x = a \wedge \alpha)}.$$

В 2001 г. Бен–Сассон и Вигдерсон предложили технику для получения нижних оценок на размеры резолюционных доказательств для пропозициональных формул [28] через минимальную ширину доказательства. Для пропозициональной формулы ϕ ширина $W(\phi)$ — это

максимальное число переменных в отдельном дизъюнкте. Ширина доказательства π — максимальное число переменных среди всех дизъюнктов доказательства. Минимальная ширина доказательства формулы $W(\phi \vdash 0)$ — это минимум ширины по всем доказательствам формулы ϕ.

Теорема ([28])**.** Для формулы ϕ, зависящей от n переменных, выполняются следующие неравенства:

$$\begin{aligned} S_T(\phi) &\geq 2^{W(\phi\vdash 0)-W(\phi)}, & (1)\\ S(\phi) &\geq \exp\left(\Omega\left(\tfrac{(W(\phi\vdash 0)-W(\phi))^2}{n}\right)\right), & (2)\end{aligned}$$

где S и S_T — размеры минимальных доказательств в Res общего вида и древовидного соответственно.

Понятие ширины естественно переносится на CSP и систему NG-Res. В 2002 Митчелл получил эквивалентные оценки для системы NG-Res [27].

Теорема ([27])**.** Для CSP ϕ, зависящей от n переменных, выполняются следующие неравенства:

$$\begin{aligned} S_T(\phi) &\geq 2^{W(\phi\vdash 0)-W(\phi)}, & (1)\\ S(\phi) &\geq \exp\left(\Omega\left(\tfrac{(W(\phi\vdash 0)-W(\phi))^2}{n}\right)\right), & (2)\end{aligned}$$

где S и S_T — размеры минимальных доказательств в NG-Res общего вида и древовидного соответственно.

Бен–Сассон и Вигдерсон получили нижние оценки на минимальную ширину доказательства через расширительную способность формулы. Пусть есть множество дизъюнктов F. Будем говорить, что переменная x лежит на границе ∂F, если ровно один дизъюнкт $f \in F$ зависит от x. Расширительная способность формулы

$$e_k(\phi) = \min_{F\subseteq S} |\partial F|,$$

где минимум берется по всем множествам F таким, что $\frac{1}{k+1}|S| \leq |F| \leq \frac{k}{k+1}|S|$. Для определенного типа формул можно показать, что $W(\phi \vdash 0) \geq e_2(\phi)$.

Прямой перенос техники Бен–Сассона и Вигдерсона на CSP дают нижнюю оценку $W(\phi \vdash 0) \geq e_k(\phi)$, где k — размер алфавита. Можно заметить, что чем больше k, тем больше вариантов для подмножества F. Соответственно, с увеличением размера алфавита, значение $e_k(\phi)$ уменьшается.

Вопрос 3. Можно ли улучшить нижнюю оценку на ширину так, чтобы она не зависела от размера алфавита?

Для CSP ϕ, чувствительной к подстановкам, можно получить нижнюю оценку $W(\phi \vdash 0) \geq e_2(\phi) - 1$.

Бен–Сассон и Вигдерсон рассмотрели невыполнимые цейтинские формулы в качестве одного из примеров для получения нижних оценок на размеры доказательств. Цейтинская формула $\texttt{Ts}(G)$ строится на основе графа G.

Расширительная способность графа $G = \langle V, E \rangle$ определяется как $e_k(G) = \min_{F \subseteq V} |E(F, V \backslash F)|$, где $E(F, V \backslash F)$ — множество ребер между множеством вершин F и его дополнением, и минимум берется по всем F для $\frac{1}{k+1}|V| \leq |F| \leq \frac{k}{k+1}|V|$. Для цейтинской формулы $\texttt{Ts}(G)$ Бен–Сассон и Вигдерсон показали, что $e_2(\texttt{Ts}(G)) \geq e_2(G)$.

В 1999 г. Басс и соавторы [29] рассмотрели обобщенные цейтинские формулы для произвольного алфавита и показали линейные нижние оценки на степень в системе полиномиального исчисления. Теорема Митчелла дает нижнюю оценку $2^{e_2(G)-d-1}$ на размер древовидного доказательства обобщенной цейтинской формулы $\texttt{Ts}(G)$ в системе NG-Res, где d — максимальная степень графа G. Кажется естественным, чтобы в основании степени стояло значение k, размер алфавита, а не 2. Возможно, детальный анализ конкретной формулы может дать такую оценку.

Вопрос 4. Верно ли, что для невыполнимой цейтинской формулы $\texttt{Ts}(G)$ можно показать нижнюю оценку $k^{e_2(G)-d}$ на размер древовидных доказательств в NG-Res?

Техники Бен–Сассона и Вигдерсона, а затем Митчелла, активно используют расширительную способность формулы для получения нижних оценок.

Вопрос 5. Является ли расширительная способность формулы необходимым свойством для получения нижних оценок на размер доказательства, хотя бы на примере древовидного случая в NG-Res?

Задача избегаемости

Рассмотрим множество слов, конечных и бесконечных, над фиксированным алфавитом Σ. В комбинаторике слов известен класс задач, где ищутся строки, избегающие определенных подстрок.

Мы можем зафиксировать набор конечных строк P над алфавитом Σ и спросить, а существует ли бесконечная в обе стороны строка, не содержащая в качестве подстроки строки из P (иначе говоря, избегающая P). Задача решается за полиномиальное время с применением алгоритма Ахо и Корасика [30].

В 2002 г. Лотар предложил систему вывода для бинарного алфавита [31]. Если бесконечная строка избегает строки $xy0$ и $y1$ (или $xy1$ и $y0$), где x и y — некоторые конечные строки над Σ, то она же избегает строку xy. Лотар доказал, что строки, избегающей P, не существует тогда и только тогда, когда можно вывести пустую строку.

В 2009 г. Блэкели и Блаше–Садри с соавторами рассмотрели задачу избегаемости частичных строк [32, 33]. Пусть строки из P заданы над алфавитом $\{0, 1, \square\}$. Мы говорим, что символ 0 согласуется с 0, символ 1 — с 1, а символ $\square$ (дырка) согласуется с чем угодно. Две строки s_1 и s_2 равной длины l согласуются друг с другом, если для каждого $i \in [l]$ согласуются символы $s_1[i]$ и $s_2[i]$. Строки из P будем называть частичными. Требуется узнать, существует ли бесконечная строка, никакая подстрока которой не согласуется ни с одной частичной строкой из P.

Бланше–Садри с соавторами показали, что задача **NP**–трудна [33]. Блэкели с соавторами показали принадлежность к классу **PSPACE** [32]. В работе [34], мои соавторы Ицыксон и Охотин показали, что задача является **PSPACE**–полной.

По аналогии с Лотаром, мы рассмотрим задачу избегаемости частичных строк в рамках сложности доказательств. Возьмем бесконечную в обе стороны строку переменных $\cdots x_{-1}x_0x_1\cdots$. Каждой частичной строке мы сопоставим счетное число дизъюнктов. Например, частичной строке $0\square\square 1$ будут соответствовать дизъюнкты $x_{1+j} \vee \neg x_{4+j}$ для $j \in \mathbb{Z}$. Бесконечной строки не существует тогда и только тогда, когда существует вывод противоречия для конъюнкции всех дизъюнктов.

В общем случае мы определим понятие подвижной формулы. Пусть ϕ обычная пропозициональная формула в КНФ, определенная на n переменных. Мы говорим, что $x^0 = x$ и $x^1 = \neg x$. Каждому дизъюнту $C = x_{i_1}^{\sigma_1} \vee \cdots \vee x_{i_k}^{\sigma_k}$ мы сопоставим счетное число дизъюнктов $C_{\to j} = x_{i_1+j}^{\sigma_1} \vee \cdots \vee x_{i_k+j}^{\sigma_k}$ для всех $j \in \mathbb{Z}$. Формулу, определенную как конъюнкцию всех дизъюнктов $C_{\to j}$ для каждого дизъюнкта C из ϕ, мы будем называть подвижной.

Ицыксон и Охотин показали, что невыполнимость подвижных формул можно доказывать, используя классические системы доказательств (здесь под классическими системами мы подразумеваем резолюционную систему доказательств, систему секущих плоскостей и систему полиномиального исчисления). Например, Ицыксон показал, что любую невыполнимую подвижную формулу на основе формулы ϕ, зависящей от n переменных, можно доказать в Res с доказательством размера не больше $2^{O(n^2)}$ [34].

Структура формул позволяет ввести дополнительное правило — правило сдвига. На примере резолюционных доказательств это правило выглядит так: для дизъюнкта C и для любого $j \in \mathbb{Z}$ можно вывести дизъюнкт $C_{\to j}$ как он определен выше.

Вопрос 6. Может ли правило сдвига существенно сократить размер доказательства подвижной формулы?

Аналогичное правило сдвига можно добавить в систему секущих плоскостей, полиномиального исчисления или другую.

Вопрос 7. Существуют ли трудные формулы с экспоненциальными нижними оценками на размеры доказательств в системах доказательств

с правилом сдвига (резолюционных, секущих плоскостях, полиномиальном исчислении)?

Цели, результаты и структура диссертации

Цели работы.

1. Получить верхнюю оценку на принцип паросочетания в произвольном графе в системе Res, которая совпадает с нижней с точностью до константы в экспоненте.

2. Получить верхнюю оценку на принцип Дирихле для древовидных доказательств в системе Res-Lin, которая совпадает с нижней с точностью до константы в экспоненте.

3. Доказать нижнюю оценку на минимальную ширину доказательства в NG-Res, не зависящую от размера алфавита.

4. Показать, что для невыполнимой цейтинской формулы $\mathtt{Ts}(G)$, построенной на графе G с максимальной степенью d, древовидное доказательство в системе NG-Res имеет нижнюю оценку $k^{e_2(G)-d}$.

5. Построить верхнюю оценку на древовидные доказательства в системе NG-Res через расширительную способность CSP.

6. Построить пример подвижных формул, на которых классические системы доказательств отделяются от систем доказательств со сдвигом.

7. Получить нижнюю оценку на системы доказательств со сдвигом: резолюционные, секущих плоскостей и полиномиального исчисления.

Научная новизна. Все результаты диссертации являются новыми.

Теоретическая и практическая ценность. Работа носит теоретический характер. Ее результаты могут быть использованы для дальнейших исследований в структурной теории сложности и теории сложности доказательств.

Методы исследований. В работе используются методы теории сложности вычислений и доказательств.

Основные результаты.

1. Доказано, что формула $\texttt{PMP}_G$ для любого графа G на n вершинах без совершенного паросочетания имеет доказательство в системе Res размера $O(n^2 \cdot 2^n)$. Оценка совпадает с ранее известной нижней с точностью до константы в показателе экспоненты.

2. Доказано, что формула $\texttt{PHP}_n^m$ при $m > n$ имеет древовидное доказательство размера $2^{O(n)}$ в системе Res-Lin. Оценка совпадает с нижней с точностью до константы в показателе экспоненты.

3. Доказано, что минимальная ширина доказательства чувствительной к подстановкам CSP ϕ ограничена снизу $e_2(\phi) - 1$. Нижняя оценка не зависит от размера алфавита.

4. Доказано, что обобщенные цейтинские формулы $\texttt{Ts}(G, f)$, построенные на основе графа G с максимальной степенью d и расширительной способностью $e_2(G)$ над алфавитом размера k, имеют древовидные доказательства в NG-Res размера как минимум $k^{e_2(G)-d}$.

5. Показана формально необходимость расширительной способности графа для нижних оценок на древовидные доказательства в системе NG-Res. По CSP ϕ над алфавитом размера k можно построить граф зависимостей $G = \langle V, E \rangle$, который описывает сколько общих переменных имеет каждая пара ограничений. Показано, что в графе G существует подграф H такой, что древовидная сложность CSP ϕ в системе NG-Res не больше $k^{e(H)\cdot\log_{3/2}|V|}$.

6. Построен пример невыполнимой подвижной формулы, которая имеет вывод полиномиального размера в резолюционной системе доказательств со сдвигом, однако в любой классической системе доказательств без сдвига вывод имеет экспоненциальный размер.

7. Доказаны нижние экспоненциальные оценки для систем доказательств с правилом сдвига (резолюционных, секущих плоскостей и полиномиального исчисления).

Достоверность и надежность результатов. Все положения диссертационной работы являются достоверными научными фактами, получившими в диссертации полные математически строгие доказательства.

Апробация работы. Результаты диссертационной работы были изложены на следующих конференциях и семинарах.

1. Международная конференция "First Russian-Finnish Symposium on Discrete Mathematics" (Турку, RuFiDim 2012).

2. Международная конференция "The 8th International Computer Science Symposium in Russia" (Екатеринбург, CSR 2013).

3. Международная конференция "19th International Conference on Theory and Applications of Satisfiability Testing" (Бордо, SAT 2016).

4. Международная конференция "41st International Symposium on Mathematical Foundations of Computer Science" (Краков, MFCS 2016)

5. Научный семинар в LIRMM, Университет Монпелье 2, 2016.

Разделение результатов. Основные результаты диссертации опубликованы в рецензируемых научных изданиях — [34, 35, 36, 37]. Работы [34, 35, 37] написаны в соавторстве.

В работе [34] **PSPACE**–полнота задачи избегаемости принадлежит соавторам; диссертанту принадлежит доказательство теоремы о разделении систем доказательств с правилом сдвига и без него, при этом разделяющие формулы предложил Д.М. Ицыксон. Нижние оценки на системы доказательств с правилом сдвига получены диссертантом.

Работа [35] написана в соавторстве с научным руководителем. Научному руководителю принадлежит постановка задачи, диссертанту принадлежит техническая часть всех доказательств.

В работе [37] диссертанту принадлежит верхняя оценка на формулы, кодирующие принцип совершенного паросочетания, остальные результаты принадлежат соавторам.

Структура диссертации. В главе 1 определяются основные понятия, используемые в диссертации. В главе 2 дается верхняя оценка на принцип совершенного паросочетания для резолюционной системы доказательств в общем виде; даются верхние оценки на древовидные доказательства принципа Дирихле и принципа совершеннного паросочетания в системе Res-Lin. В главе 3 дается усиленная версия метода Бен–Сассона и Вигдерсона для получения нижних оценок на ширину доказательства в системе NG-Res; дается усиленная нижняя оценка на размер древовидных доказательств в системе NG-Res для обобщенных цейтинских формул; дается верхняя оценка на размеры древовидных доказательств в системе NG-Res через расширительную способность CSP. В главе 4 вводится система доказательств с правилом сдвига; дается сведение нижних оценок для систем доказательств с правилом сдвига к систем доказательствам без него; приводится пример семейства формул, для которых любое классическое доказательство имеет экспоненциальный размер, а доказательство в резолюционной системе с правилом сдвига — полиномиальный от длины формулы. В заключении подводятся итоги диссертации и ставятся открытые вопросы.

Глава 1

Основные понятия

1.1 Основные обозначения

Множества.

- Обозначим $[n] := \{1, 2, \cdots, n\}$, где $n \in \mathbb{Z}_0^+$. При этом $[0] = \emptyset$.
- Обозначим $[a, b] := \{a, a+1, \cdots, b\}$, где $a, b \in \mathbb{Z}$ и $a \leq b$.
- Для множества S и элемента $a \in S$ обозначим $S - a := S \backslash \{a\}$.
- Кольцо вычетов по модулю n обозначим $\mathbb{Z}_n$.
- Поле вычетов по простому модулю p обозначим $\mathbb{F}_p$.

Логические операции.

- $\wedge$ — логическое И.
- $\vee$ — логическое ИЛИ.
- $\neg$ — отрицание.
- $\oplus$ — сложение по модулю два, а также в поле $\mathbb{F}_2$.

Графы. Рассмотрим простой неориентированный граф $G = \langle V, E \rangle$. В обозначении пары первый элемент будет всегда множеством вершин, второй — множеством ребер.

- $E(A, B)$ — подмножество ребер из E, соединяющих вершины из подмножеств $A \subseteq V$ с вершинами из подмножества $B \subseteq V$.
- $E(A, v) := E(A, \{v\})$.
- Для подмножества вершин $S \subseteq V$ определим граф $G - S$ как подграф графа G, индуцированный на множестве $V \backslash S$.
- Для ребра $e \in E$ определим подграф $G - e := \langle V, E - e \rangle$.

Предикаты. Два предиката P и S, зависящие от множества переменных X, равносильны, если на любой подстановке $\rho : X \to \{0, 1\}$ значения P и S равны.

- Равносильные предикаты P и S обозначим $P \equiv S$.
- Тождественно–истинный предикат P обозначим $P \equiv 1$. Тождественно–истинный предикат будем также называть тривиальным.
- Тождественно–ложный предикат P обозначим $P \equiv 0$.

1.2 Классы сложности

Зафиксируем алфавит $\Sigma = \{0, 1\}$. Язык L — это подмножество множества всех конечных слов Σ^*.

Отождествим характеристическую функцию языка с самим языком: $L(x) = 1 \Leftrightarrow x \in L$. Длину слова x будем обозначать $|x|$. Во всех определениях ниже мы подразумеваем, что машина Тьюринга детерминированная и многоленточная.

- Язык L разрешим за полиномиальное время, если существует машина Тьюринга M такая, что $M(x) = L(x)$ и время работы машины

не превосходит полинома от $|x|$. Класс всех полиномиально разрешимых языков обозначается **P**.

- Язык L полиномиально проверяем, если существуют два полинома p и q, и машина Тьюринга M такие, что для любого слова $x \in L$ существует слово y длины не более $q(|x|)$; $M(x, y) = 1$; и время работы машины M не превосходит $p(|x|)$. Для любого слова $x \notin L$ на любом входе y' результат $M(x, y') = 0$. Класс всех полиномиально проверяемых языков обозначается **NP**.

- Язык $\bar{L} = \Sigma^* \backslash L$ называется дополнением к языку L. Класс дополнений к языкам из **NP** обозначается **co-NP**.

- Язык L разрешим полиномиально по памяти, если существует машина Тьюринга M такая, что $M(x) = L(x)$, и машина затрагивает не более чем полином от $|x|$ ячеек лент. Класс языков, разрешимых полиномиально по памяти, обозначается **PSPACE**.

1.3 Системы доказательств

Определение 1.1 ([1, 38])**.** Система доказательств для языка L — это полиномиальный по времени алгоритм $\Pi : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$, обладающий двумя свойствами:

- полнота — для любого $x \in L$ существует строка y, что $\Pi(x, y) = 1$;

- корректность — для любого $x \notin L$ и любой строки y выполняется равенство $\Pi(x, y) = 0$.

В указанных обозначениях строку y называют Π*–доказательством* принадлежности x языку L. Говорят, что система *полиномиально ограниченна*, если существует полином p такой, что для любого $x \in L$ существует строка y такая, что $|y| \leq p(|x|)$ и $\Pi(x, y) = 1$.

Теорема 1.1 ([1])**.** Язык $L \in$ **NP** тогда и только тогда, когда для него существует полиномиально ограниченная система доказательств.

Язык пропозициональных тавтологий обозначается TAUT. Одной из основных проблем теории сложности является вопрос равенства классов **NP** и **co-NP**. Следующая теорема переводит этот вопрос на язык теории сложности доказательств.

Теорема 1.2 ([1])**.** Классы **NP** и **co-NP** равны тогда и только тогда, когда для языка TAUT существует полиномиально ограниченная система доказательств.

Если показать, что такой системы не существует, то мы получим разделение классов **NP** и **co-NP** и, как следствие, классов **P** и **NP**.

Системы доказательств можно сравнивать.

Определение 1.2 ([1])**.** Пусть Π_1 и Π_2 — системы доказательств для языка L. Система Π_1 *моделирует* систему Π_2, если для любого слова $x \in L$ выполняет неравенство $|y_{\Pi_1}| \leq \text{poly}(y_{\Pi_2})$, где y_{Π_1}, y_{Π_2} — кратчайшие доказательства для x в системах Π_1 и Π_2 соответственно.

Определение 1.3 ([1])**.** Система Π_1 *p–моделирует* систему Π_2, если существует полиномиальный по времени алгоритм A такой, что для любого слова $x \in L$ и любого слова y если $\Pi_2(x, y) = 1$, то $\Pi_1(x, A(x, y)) = 1$.

1.3.1 Язык UNSAT

Определим язык формул в конъюнктивной нормальной форме (КНФ). Для переменной x записи x и $\neg x$ являются литералами. Мы иногда будем записывать x как x^0 и $\neg x$ как x^1. Когда знак не важен, будем использовать запись x^σ.

Дизъюнкт — это дизъюнкция нуля и более литералов $x_1^{\sigma_1} \vee x_2^{\sigma_2} \vee \cdots \vee x_k^{\sigma_k}$. Дизъюнкт с нулем литералов, или пустой дизъюнкт, является тождественной ложью. Формула представлена в КНФ, если записана в виде конъюнкции нуля и более дизъюнктов. Конъюнкция нуля дизъюнктов по определению считается тождественной истиной.

Подстановка ρ — это отображение из множества переменных X в $\{0,1\}$. Подстановка выполняет формулу, если после подстановки значений переменным, соответствующее логическое выражение равно истине. Формула выполнима, если существует подстановка, выполняющая формулу. Язык невыполнимых формул в КНФ обозначим `UNSAT`.

Частичная подстановка ρ — это отображение из X в $\{0,1,*\}$, где $*$ означает, что мы не назначаем переменной конкретного значения.

1.3.2 Резолюционная система доказательств

В этом подразделе и следующих мы будем рассматривать системы доказательств для языка `UNSAT`.

Резолюционная система доказательств, Res, оперирует дизъюнктами. Мы накладываем ограничение, что ни один дизъюнкт не включает в себя переменную с положительным и отрицательным знаком одновременно. Резолюционное доказательство формулы ϕ — это последовательность дизъюнктов $C_1, C_2, \cdots, C_l$, где каждый дизъюнкт является либо дизъюнктом формулы ϕ, либо получен из предыдущих дизъюнктов по следующим правилам вывода.

- Правило ослабления:
$$\frac{C}{C \vee x^\sigma}.$$
Из дизъюнкта C можно вывести дизъюнкт $C \vee x^\sigma$ при условии, что переменная x не лежит в C, но используется в записи формулы ϕ.

- Правило резолюции:
$$\frac{x \vee C_1 \quad \neg x \vee C_2}{C_1 \vee C_2}.$$

Последний дизъюнкт C_l в доказательстве пустой, следовательно является тождественно ложными. Длина доказательства — это число дизъюнктов в нем. Доказательство является древовидным, если каждый дизъюнкт, полученный по правилу вывода, используется как посылка не более одного раза.

Мы определим вывод дизъюнкта C из формулы ϕ по аналогии с доказательством ϕ с той лишь разницей, что последний дизъюнкт вывода совпадает с C, а не пустой.

Определение 1.4. Формула ψ семантически следует из формулы ϕ, если любая подстановка, выполняющая ϕ, выполняет ψ.

Лемма 1.1 ([28])**.** Пусть формула ϕ задана в КНФ, и дизъюнкт C семантически следует из ϕ. Тогда дизъюнкт C выводим в резолюционной системе доказательств из дизъюнктов формулы ϕ.

1.3.3 Система доказательств секущих плоскостей

Система доказательств секущих плоскостей (или система секущих плоскостей) оперирует линейными неравенствами с целочисленными коэффициентами. Для каждого дизъюнкта

$$C = x_1^{\sigma_1} \vee x_2^{\sigma_2} \vee \cdots \vee x_k^{\sigma_k} \tag{1.1}$$

формулы ϕ мы введем неравенство

$$\sum_{i \in [k]} [(1 - \sigma_i) \cdot x_i + \sigma_i \cdot (1 - x_i)] \geq 1. \tag{1.2}$$

Для каждой переменной x_i введем две аксиомы: $x_i \geq 0$ и $-x_i \geq -1$.

На любой подстановке $\rho : \{x_i\}_{i=1}^n \to \{0, 1\}$ дизъюнкт 1.1 и неравенство 1.2 равносильны.

Доказательство в системе секущих плоскостей — это последовательность неравенств, каждое из которых либо соответствует дизъюнкту формулы, либо является аксиомой, либо выводится из предыдущих по следующим правилам вывода.

- Сложение:
$$\frac{\sum_{i \in I} a_i \cdot x_i \geq \alpha \quad \sum_{i \in I} b_i \cdot x_i \geq \beta}{\sum_{i \in I} (a_i + b_i) \cdot x_i \geq \alpha + \beta}.$$

- Умножение:
$$\frac{\sum_{i\in I} a_i \cdot x_i \geq \alpha}{\sum_{i\in I} c \cdot a_i \cdot x_i \geq c \cdot \alpha},$$
для любого $c \in \mathbb{Z}$.

- Деление с округление:
$$\frac{k \cdot \sum_{i\in I} a_i \cdot x_i \geq \alpha}{\sum_{i\in I} a_i \cdot x_i \geq \lceil \frac{\alpha}{k} \rceil},$$
где k — целое положительное число.

Последнее неравенство в выводе — противоречие: $0 \geq 1$.

Теорема 1.3 ([39])**.** Система секущих плоскостей p–симулирует резолюционную систему доказательств.

1.3.4 Система доказательств полиномиального исчисления

Система доказательств полиномиального исчисления (или просто система полиномиального исчисления) оперирует полиномиальными уравнениями над полем K. Каждому дизъюнкту формулы $C = x_1^{\sigma_1} \vee x_2^{\sigma_2} \vee \cdots \vee x_k^{\sigma_k}$ мы сопоставим уравнение $\prod_{i\in[k]}(1 - x_i)^{1-\sigma_i} \cdot x_i^{\sigma_i} = 0$. Для каждой переменной x_i мы введем аксиому $x_i(1 - x_i) = 0$.

Доказательство в системе полиномиального исчисления — это последовательность уравнений, каждое из которых либо аксиома, либо соответствует дизъюнкту формулы, либо выводится из предыдущих уравнений по следующим правилам вывода:

- Линейная комбинация:
$$\frac{R = 0 \quad S = 0}{aR + bS = 0}$$
для любых полиномов R и S и любых элементов $a, b \in K$.

- Умножение на переменную:
$$\frac{R = 0}{x_i \cdot R = 0},$$
где x_i — любая переменная.

Вывод заканчивается противоречием: $1 = 0$.

1.3.5 Резолюционная система доказательств с резолюцией по линейным формам Res-Lin

Система Res-Lin является обобщением Res и оперирует дизъюнкциями линейных уравнений в поле $\mathbb{F}_2$. Преобразуем каждый дизъюнкт $C = x_1^{\sigma_1} \vee x_2^{\sigma_2} \vee \cdots \vee x_k^{\sigma_k}$ формулы ϕ в дизъюнкт $\bigvee_{i\in[k]} [x_i = 1 - \sigma_i]$.

Пусть формула ϕ невыполнима и зависит от n переменных. Доказательство формулы ϕ в системе Res-Lin определяется аналогично предыдущим со следующими правилами вывода:

- Правило ослабления:
$$\frac{C}{D},$$
Из дизъюнкции линейных уравнений C можно вывести дизъюнкцию линейных уравнений D при условии, что D семантически следует из C: любая подстановка ρ выполняющая C, выполняет D.

- Правило резолюции:
$$\frac{[\bigoplus_{i\in I} x_i = 0] \vee C_1 \quad [\bigoplus_{i\in I} x_i = 1] \vee C_2}{C_1 \vee C_2}.$$

Последняя дизъюнкция в доказательстве пуста.

1.4 Алгоритмы для SAT и деревья расщеплений

1.4.1 DPLL алгоритм

Пусть дана пропозициональная формула ϕ в КНФ. Задача поиска выполняющей подстановки для ϕ называется задачей выполнимости; обозначается SAT. DPLL алгоритм решает задачу SAT и параметризуются двумя эвристиками A и B.

Входные данные: пропозициональная формула ϕ в КНФ.
Результат: выполняющая подстановка ρ или ответ `UNSAT`, если такого не существует.

1 **if** *формула ϕ содержит пустой дизъюнкт* **then**
2 | вернуть `UNSAT`;
3 **end**
4 $x := A(\phi)$;
5 $\alpha := B(\phi, x)$;
6 **if** DPLL($\phi|_{x=\alpha}$) *вернул подстановку* ρ **then**
7 | вернуть ρ;
8 **end**
9 **if** DPLL($\phi|_{x=\neg\alpha}$) *вернул подстановку* ρ **then**
10 | вернуть ρ;
11 **end**
12 вернуть `UNSAT`;

Описание 1: DPLL алгоритм

Первая эвристика по формуле ϕ выбирает переменную x. Вторая эвристика по формуле ϕ и переменной x выбирает значение $\alpha \in \{0, 1\}$. Псевдокод алгоритма приведен в описании 1.

Формула $\phi|_{x=\alpha}$ в строке 6 получается путем подстановки переменной x значения α. Мы убираем все выполненные дизъюнкты, зависящие x, и все вхождения литералов x^σ, которые обнулились. Аналогичное происходит в строке 9

Пусть формула ϕ невыполнима. Протокол работы DPLL алгоритма можно записать в виде дерева. В каждом внутреннем узле дерева записана переменная. Каждое ребро помечено значением 0 или 1. Каждый лист дерева помечен дизъюнктом. При этом, частичная подстановка, которая задается путем от корня до листа, нарушает дизъюнкт, записанный в листе.

Описанное дерево называется *деревом расщеплений*. Минимальный размер дерева расщеплений совпадает с размером минимального резолюционного доказательства и совпадает с точностью до константы с минимальным временем работы DPLL алгоритма на невыполнимых фор-

мулах [9].

1.4.2 Алгоритм линейных расщеплений

Входные данные: пропозициональная формула ϕ в КНФ и система линейных уравнений Ψ над полем $\mathbb{F}_2$
Результат: выполняющая подстановка ρ или ответ `UNSAT`, если такого не существует.

if *система* Ψ *несовместна* **then**
  вернуть `UNSAT`;
end
for *каждого дизъюнкта* C *формулы* ϕ **do**
  if *для каждого литерала* $x^\sigma \in C$ *система* $\Psi \wedge [x = 1 \oplus \sigma]$ *несовместна* **then**
    вернуть `UNSAT`;
  end
end
if *система* Ψ *имеет ровно одно решение* ρ **then**
  вернуть ρ;
end
$I := A(\phi, \Psi)$;
$\alpha := B(\phi, \Psi, I)$;
if $\mathrm{DPLL}_{lin}\left(\phi, \Psi \wedge \left[\bigoplus_{i \in I} x_i = \alpha\right]\right)$ *вернул подстановку* ρ **then**
  вернуть ρ;
end
if $\mathrm{DPLL}_{lin}\left(\phi, \Psi \wedge \left[\bigoplus_{i \in I} x_i = 1 \oplus \alpha\right]\right)$ *вернул подстановку* ρ **then**
  вернуть ρ;
end
вернуть `UNSAT`;

Описание 2: Алгоритм линейных расщеплений DPLL_{lin}

Алгоритм линейных расщеплений, DPLL_{lin}, принимает на вход пропозициональную формулу ϕ в КНФ и систему линейных уравнений Ψ над полем $\mathbb{F}_2$, изначально пустую; ищет выполняющую подстановку для формулы ϕ среди всех решений системы Ψ. Если система Ψ пуста, алгоритм решает задачу `SAT`.

Алгоритм линейных расщеплений параметризуется двумя эвристиками A и B. Эвристика A по формуле ϕ и системе Ψ выбирает множество индексов переменных I. Эвристика B по формуле ϕ, системе Ψ и множеству индексов I выбирает значение $\alpha \in \{0, 1\}$.

Псевдокод алгоритма приведен на описании 2.

Отметим, что проверки на совместность в строках 1 и 5 и на единственность решения в строке 9 осуществляются за полиномиальное время. Будем говорить, что система Ψ нарушает дизъюнкт C, если любое решение Ψ не выполняет дизъюнкт C. В строках 4-8 алгоритм проверяет, не нарушает ли система Ψ хотя бы один дизъюнкт формулы ϕ.

Рассмотрим протокол работы алгоритма на невыполнимой формуле ϕ. Протокол можно представить в виде дерева. В каждом внутреннем узле записана линейная форма $f = \bigoplus_{i \in I} x_i$. Каждое ребро помечено уравнением вида $f = 0$ или $f = 1$.

Мы сделаем допущение, что эвристика $A(\phi, \Psi)$ всегда выбирает форму, линейно независимую от форм в системе Ψ. Это значит, что алгоритм никогда не попадает внутрь условия на строке 1. Потому все листья дерева помечены дизъюнктами формулы ϕ. При этом система уравнений, составленная на пути от корня до листа, нарушает дизъюнкт в листе.

Описанное выше дерево назовем *деревом линейных расщеплений*. Минимальное дерево линейных расщеплений совпадает с минимальным древовидным доказательством в Res-Lin и совпадает с точностью до константы с минимальным временем работы алгоритма линейных расщеплений на невыполнимой формуле [25].

1.5 Задача выполнения ограничений

Пусть есть конечный алфавит D размера k и множество переменных $X = \{x_1, x_2, \cdots, x_n\}$, которые принимают значения из алфавита. Пусть также есть множество предикатов (ограничений) S, которые зависят от переменных из X.

Определение 1.5. Задача выполнения ограничений (consraint satisfaction problem, CSP) $\phi = \langle X, D, S\rangle$ состоит в поиске подстановки $\rho : X \to D$, выполняющей каждое ограничений из S.

В случае, когда алфавит бинарный, CSP совпадает с задачей выполнимости пропозициональной формулы. CSP невыполнима, если не существует подстановки ρ, выполняющей все ограничения одновременно.

Определим запрещающий набор как ограничение, имеющее вид

$$\neg\left(\bigwedge_{i\in I} x_i = a_i\right),$$

где $I \subseteq [n]$ — подмножество индексов переменных, $a_i \in D$. Набор с $I = \emptyset$ будем обозначать $\neg(\texttt{True})$.

Пусть ограничение $f \in S$ зависит от множества переменных X_f. Будем говорить, что запрещающий набор N соответствует ограничению f, если он использует все переменные X_f и подстановка, нарушающая N, нарушает f.

Определим резолюционную систему доказательств NG-Res для CSP. Пусть, для простоты, $D = [k]$. Система NG-Res оперирует запрещающими наборами. Доказательство в NG-Res определяется аналогично пропозициональной системе Res. Каждый запрещающий набор либо соответствует ограничению из S, либо выводится из предыдущих по следующим правилам вывода.

- Правило ослабления:
$$\frac{\neg(\alpha)}{\neg(x = a \wedge \alpha)}.$$
Из запрещающего набора $\neg(\alpha)$ можно вывести запрещающий набор $\neg(x = a \wedge \alpha)$ при условии, что x не лежит в α.

- Правило резолюции:
$$\frac{\neg(x = 1 \wedge \alpha_1)\ \ \neg(x = 2 \wedge \alpha_2)\ \ \cdots\ \ \neg(x = k \wedge \alpha_k)}{\neg(\wedge_{a\in D}\alpha_a)}.$$

Последний набор в доказательстве — пустой: $\neg(\texttt{True})$.

Кроме размера доказательства мы будем использовать понятие ширины. Ширина CSP ϕ — это максимальное число переменных от которых зависит ограничения ϕ. Ширина доказательства π в системе NG-Res для CSP ϕ определяется как максимальное число переменных в отдельном запрещающем наборе N доказательства π. Минимальная ширина доказательства $W(\phi \vdash 0)$ для CSP ϕ — это минимальная ширина по всем возможным доказательствам CSP ϕ.

Теорема 1.4 ([27])**.** Пусть CSP $\phi = \langle X, D, S\rangle$ зависит от n переменных, каждое ограничение $g \in S$ зависит не более чем от d переменных. Тогда

$$\begin{aligned} S_T(\phi) &\geq 2^{W(\phi\vdash 0)-d}, \\ S(\phi) &\geq \exp\left(\Omega\left(\frac{(W(\phi\vdash 0)-d)^2}{n}\right)\right), \end{aligned}$$

где $S_T(\phi)$ — размер минимального древовидного доказательства в системе NG-Res для CSP ϕ, а $S(\phi)$ — размер минимального доказательства общего вида в системе NG-Res для CSP ϕ.

Определение 1.6. Пусть CSP $\phi = \langle X, D, S\rangle$ невыполнима. Мы скажем, что CSP ϕ минимально невыполнима, если для любого ограничения $f \in S$ CSP $\phi_{-f} = \langle X, D, S - f\rangle$ выполнима.

Пусть $F \subseteq S$ — подмножество ограничений CSP $\phi = \langle X, D, S\rangle$. Будем говорить, что переменная x лежит на границе F, если существует ровно одно ограничение $f \in F$, которое формально зависит от x. Множество всех переменных, которые лежат на границе F, обозначим ∂F.

Определение 1.7. Для CSP $\phi = \langle X, D, S\rangle$ расширительная способность с параметром t определяется как

$$e_t(\phi) = \min_F \partial F,$$

где минимум берется по всем $F \subseteq S : \frac{1}{t+1}|S| \leq |F| \leq \frac{t}{t+1}|S|$.

По аналогии с пропозициональным случаем, мы определим дерево расщеплений для CSP. Дерево T является деревом расщеплений, если

- каждый внутренний узел помечен переменной из X;
- каждый внутренний узел имеет k детей;
- каждое ребро, идущее в ребенка, помечено значением из D, при этом, на всех ребрах, исходящих из одной вершины, все метки разные;
- каждый лист помечен запрещающим набором, который соответствует ограничению из S;
- частичная подстановка, которая получается на пути от корня до листа, нарушает запрещающий набор, записанный в листе.

Минимальное по размеру дерево расщеплений мы будем иногда называть оптимальным.

1.6 Графы

Паросочетанием в графе $G = \langle V, E \rangle$ называется множество ребер $M \subseteq E$ таких, что никакая пара ребер не имеют общей вершины. Паросочетание называется *совершенным*, если каждой вершины графа инцидентно ребро из паросочетания.

Расширительная способность графа $G = \langle V, E \rangle$ с параметром t — это число

$$e_t(G) = \min_{U \subseteq V} |E(U, V \backslash U)|,$$

где минимум берется по всем U таким, что $\frac{1}{t+1} \cdot |V| \leq |U| \leq \frac{t}{t+1} \cdot |V|$. Если параметр t не упоминается, мы подразумеваем $e(G) = e_2(G)$.

Глава 2

Верхние оценки для резолюционных систем доказательств

Данная глава посвящена верхним оценкам для резолюционных систем доказательств общего вида и резолюциям по линейным формам.

В разделе 2.1 мы определим два семейства формул, для которых будут показаны верхние оценки. Первое семейство формул $\mathtt{PHP}_n^m$ кодирует принцип Дирихле, второе семейство $\mathtt{PMP}_G$ — принцип совершенного паросочетания.

При обсуждении принципа совершенного паросочетания, мы будем активно использовать критерий Татта.

Теорема 2.1 (Критерий Татта, 1952, [40])**.** Граф $G = \langle V, E \rangle$ не содержит совершенного паросочетания тогда и только тогда, когда существует подмножество вершин $S \subseteq V$ такое, что число компонент связности нечетного размера в графе $G - S$, обозначим $o(G - S)$, больше размера множества S:

$$o(G - S) > |S|.$$

В разделе 2.2 мы покажем верхнюю оценку на размер резолюционного доказательства для принципа совершенного паросочетания. Пусть граф G содержит n вершин и не содержит совершенного паросочетания.

В теореме 2.2 мы покажем, что формула $\mathtt{PMP}_G$ имеет доказательство размера $O\left(n^2 \cdot 2^n\right)$.

В разделе 2.3 мы рассмотрим древовидные доказательства в системе Res-Lin. Для описания структуры доказательства мы будем использовать деревья линейных расщеплений, определенные в главе 1.

В подразделе 2.3.1 мы покажем верхнюю оценку на размер деревьев линейных расщеплений для формулы $\mathtt{PHP}_n^{n+1}$. В теореме 2.3 мы докажем, что для любых $m > n$ формула $\mathtt{PHP}_n^m$ имеет дерево линейных расщеплений размера $2^{O(n)}$.

В разделе 2.3.2 мы покажем верхнюю оценку на размер деревьев линейных расщеплений для принципа совершенного паросочетания как следствие из теоремы 2.3. Пусть граф G содержит n вершин и не содержит совершенного паросочетания. В теореме 2.5 мы докажем, что формула $\mathtt{PMP}_G$ имеет дерево линейных расщеплений размера $2^{O(n)}$, при этом n может быть четным.

Результаты этой главы опубликованы в работах [36, 37].

2.1 Формулы для принципов Дирихле и совершенного паросочетания

В этой главе мы будем работать с двумя семействами формул: первые кодируют принцип Дирихле, вторые — принцип совершенного паросочетания.

2.1.1 Принцип Дирихле

Пусть есть m кроликов и n клеток. Каждого кролика нужно посадить хотя бы в одну клетку. Принцип Дирихле сообщает, что если $m > n$, то найдется клетка хотя бы с двумя кроликами.

Мы определим семейство формул $\{\mathtt{PHP}_n^m\}_{n,m\in\mathbb{N}}$ в конъюнктивной нормальной форме. Каждая формула $\mathtt{PHP}_n^m$ кодирует утверждение, что m кроликов можно посадить в n клеток так, что в каждой клетке окажется

не более одного кролика. Формула использует $m \cdot n$ переменных $x_{i,j}$ для $i \in [m]$ и $j \in [n]$. Переменная $x_{i,j}$ равна единице тогда и только тогда, когда кролик i посажен в клетку j.

Мы составим два набора ограничений.

1. Для каждого кролика $i \in [m]$ запишем дизъюнкт:

$$\bigvee_{j \in [n]} x_{i,j},$$

   который гарантирует, что кролик i будет посажен хотя бы в одну клетку. Заметим, что мы не запрещаем отправлять одного кролика в несколько клеток сразу.

2. Для каждой пары различных кроликов $i_1, i_2 \in [m]$ и каждой клетки $j \in [n]$ запишем дизъюнкт:

$$\neg x_{i_1,j} \vee \neg x_{i_2,j}.$$

   Этот набор дизъюнктов гарантирует, что в каждой клетке сидит не более одного кролика.

Формула $\mathtt{PHP}_n^m$ является конъюнкцией описанных выше дизъюнктов. При $m > n$ формула $\mathtt{PHP}_n^m$ невыполнима.

2.1.2 Принцип совершенного паросочетания

Рассмотрим граф $G = \langle V, E \rangle$ на n вершинах. На основе графа G мы определим формулу $\mathtt{PMP}_G$ в конъюнктивной нормальной форме. Каждому ребру $e \in E$ мы сопоставим переменную x_e. Мы считаем, что ребро e входит в паросочетание, если значение x_e равно единице.

Определим формулу $\mathtt{PMP}_G$ так, что любой выполняющий набор будет соответствовать совершенному паросочетанию. Для этого составим два набора дизъюнктов.

1. Каждая вершина $v \in V$ имеет инцидентное ей ребро в паросочетании:

$$\bigvee_{(u,v) \in E} x_{(u,v)}.$$

2. Никакая пара различных ребер e_1, e_2 с общей вершиной не входят в паросочетание одновременно:

$$\neg x_{e_1} \vee \neg x_{e_2}.$$

Формула $\texttt{PMP}_G$ является конъюнкцией указанных выше дизъюнктов. Если граф G не имеет совершенного паросочетания, формула $\texttt{PMP}_G$ невыполнима.

2.2 Верхняя оценка на принцип совершенного паросочетания в общей резолюции

Результат этого раздела основывается на теореме Татта и верхней экспоненциальной оценке Басса и Питасси для принципа Дирихле [41].

Теорема 2.2. Пусть граф $G = \langle V, E\rangle$ связен, содержит n вершин и не имеет совершенного паросочетания. Тогда формула $\texttt{PMP}_G$ имеет резолюционное доказательство размера $O(n^2 \cdot 2^n)$.

Доказательство теоремы 2.2 состоит из двух частей.

1. Мы покажем, что если подмножество $U \subseteq V$ — нечетного размера, то любое совершенное паросочетание содержит хотя бы одно ребро, которое имеет ровно один конец в U. Для этого мы выведем одновременно все дизъюнкты вида

$$\texttt{Odd-Out}_U = \bigvee_{e \in E(U, V\setminus U)} x_e$$

Размер вывода составит $O(n^2 \cdot 2^n)$.

2. По теореме 2.1 в графе G можно выделить множество $S = \{s_1, s_2, \cdots, s_l\}$. При этом в графе $G - S$ окажется m компонент связности нечетного размера $C_1, C_2, \cdots C_m$ и m будет больше l.

Мы сведем задачу к принципу Дирихле. Каждую компоненту связности мы будем рассматривать как кролика, а вершину множества

S — как клетку. Будем говорить, что кролик i посажен в клетку j, если между компонентой связности C_i и вершиной $s_j \in S$ хотя бы одно ребро взято в паросочетание. На предыдущем шаге мы вывели, что из каждой компоненты C_i должно исходить хотя бы одно ребро, взятое в паросочетание, т.е. каждый кролик должен быть куда-то посажен. По принципу совершенного паросочетания, каждая клетка способна принять не более одного кролика.

По аналогии с выводом Басса и Питасси для принципа Дирихле мы построим вывод размера $O(2^n)$.

Лемма 2.1. Существует резолюционный вывод размера $O(n^2 \cdot 2^n)$ дизъюнктов $\texttt{Odd-Out}_U$ для всех подмножеств $U \subseteq V$ нечетного размера.

Доказательство. Мы будем выводить дизъюнкты $\texttt{Odd-Out}_U$ по индукции на размер множества U. При $|U| = 1$, мы берем дизъюнкты исходной формулы для каждой отдельной вершины.

Пусть мы вывели все дизъюнкты $\texttt{Odd-Out}_{U'}$ для всех подмножеств U' размера меньше, чем k, и k — нечетно. Возьмем множество U размера k. Вывод дизъюнкта $\texttt{Odd-Out}_U$ зависит от одного из двух случаев.

1. Множество U содержит две вершины u и v такие, что между множествами $\{u, v\}$ и $U \backslash \{u, v\}$ ребер нет. Возьмем ранее выведенный дизъюнкт $\texttt{Odd-Out}_{U'}$ для $U' = U \backslash \{u, v\}$ и воспользуемся правилом ослабления, чтобы вывести дизъюнкт $\texttt{Odd-Out}_U = \texttt{Odd-Out}_{U'} \vee \texttt{Odd-Out}_{\{u,v\}}$.

2. Любое множество из двух вершин $\{u, v\} \subseteq U$ соединено хотя бы одним ребром с $U \backslash \{u, v\}$. Для каждого ребра $(u, v) \in E$, соединяющего вершины из U, определим дизъюнкт

$$\texttt{Neg}_U(u, v) = \neg x_{(u,v)} \vee \bigvee_{e \in E(U \backslash \{u,v\}, V \backslash U)} x_e.$$

Чтобы вывести дизъюнкт $\texttt{Neg}_U(u, v)$, нужно провести последовательную резолюцию дизъюнкта $\texttt{Odd-Out}_{U - \{u,v\}}$ с дизъюнктами $\neg x_e \vee \neg x_{(u,v)}$ для каждого ребра e между множествами $\{u, v\}$ и $U \backslash \{u, v\}$.

Мы проведем следующую последовательность действий.

(a) Возьмем дизъюнкт $\texttt{Odd-Out}_{\{v\}}$ для произвольной вершины $v \in U$.

(b) Для каждой вершины $u \in U$, соединенной ребром с v, выведем дизъюнкт $\texttt{Neg}_U(u, v)$. Проведем резолюцию дизъюнкта $\texttt{Odd-Out}_{\{v\}}$ с выведенными дизъюнктами $\texttt{Neg}_U(u, v)$.

(c) Как результат, мы получим дизъюнкт $\bigvee_{e \in E'} x_e$ для некоторого множества $E' \subseteq E(U, V \backslash U)$. Правилом ослабления добавим недостающие переменные, чтобы получить дизъюнкт $\texttt{Odd-Out}_U$.

Оценим размер вывода. В первом случае мы используем $O(n)$ правил ослабления. Во втором — на вывод каждого дизъюнкта $\texttt{Neg}_U(u, v)$ мы используем $O(n)$ резолюций, всего мы выводим $O(n)$ дизъюнктов $\texttt{Neg}_U(u, v)$. Ослабление задействует не более $O(n^2)$ переменных. Итого, на каждом шаге мы добавляем в доказательство $O(n^2)$ дизъюнктов.

Множество V содержит 2^n различных подмножеств. Значит, размер всего вывода $O(n^2 \cdot 2^n)$. □

Доказательство теоремы 2.2. Граф G не содержит совершенного паросочетания. По теореме 2.1 в графе $G = \langle V, E \rangle$ есть множество вершин $S = \{s_1, \cdots, s_l\}$, и число компонент связности нечетного размера в графе $G - S$ больше $|S|$. Обозначим такие компоненты $C_1, C_2, \cdots, C_m$. Мы знаем, что $m > l$. Для вывода противоречия нам достаточно взять только первые $l + 1$ компоненту связности. Будем считать $m = l + 1$.

Воспользуемся леммой 2.1 и выведем для каждой компоненты связности C_i дизъюнкт $\texttt{Odd-Out}_{C_i}$. Поскольку C_i — компонента связности графа $G - S$, то все исходящие из C_i ребра $E(C_i, V \backslash C_i)$ совпадают с ребрами, идущими в S, т.е. $E(C_i, V \backslash C_i) = E(C_i, S)$.

Определим множество $S_t = \{s_t, \cdots, s_l\}$. Для каждого набора компонент связности $T = \{C_{i_1}, \cdots, C_{i_t}\}$ размера t определим множество вершин $V(T) = \cup_{C \in T} C$ и дизъюнкт

$$\texttt{Subset-Out}_T = \bigvee_{e \in E(S_t, V(T))} x_e.$$

Будем выводить $\texttt{Subset-Out}_T$ для всех возможных T по индукции по размеру T. Очевидно, при $t > l$, множество $S_t = \emptyset$, и соответствующий дизъюнкт $\texttt{Subset-Out}_T$ пуст.

При $t = 1$ все дизъюнкты $\texttt{Subset-Out}_{\{C_i\}}$ совпадают с дизъюнктами $\texttt{Odd-Out}_{C_i}$. Последние выводятся по лемме 2.1. Пусть мы вывели все дизъюнкты $\texttt{Subset-Out}_{T'}$ для $|T'| < t$. Выведем дизъюнкт $\texttt{Subset-Out}_T$ для T размера t. Рассмотрим два случая.

1. Множество T содержит не более одной компоненты связности C, соединенной ребрами с вершиной s_{t-1}. Если ни одна компонента не соединена с вершиной s_{t-1}, возьмем в качестве C произвольную. Рассмотрим дизъюнкт

$$\texttt{Subset-Out}_{T-C} = \bigvee_{e \in E(S_{t-1}, V(T-C))} x_e.$$

   Заметим, что $E(S_{t-1}, V(T-C)) = E(S_t, V(T-C))$, поскольку ни одна компонента связности в $T-C$ не соединена ребром с s_{t-1}. Ослабим дизъюнкт $\texttt{Subset-Out}_{T-C}$, добавив ребра исходящие из C в S_t. В результате получим дизъюнкт

$$\texttt{Subset-Out}_T = \bigvee_{e \in E(S_t, V(T))} x_e.$$

2. Множество T содержит хотя бы две компоненты связности, которые соединены ребрами с вершиной s_{t-1}.

   Для каждой компоненты связности $C \in T$ и каждого ребра $e \in E(C, s_{t-1})$, выведем дизъюнкт

$$\texttt{Kill-Edge}_{C,T}(e) = \neg x_e \vee \bigvee_{e' \in E(V(T-C), S_t)} x_{e'}.$$

   Дизъюнкт $\texttt{Kill-Edge}_{C,T}(e)$ выводится последовательной резолюцией ранее выведенного дизъюнкта $\texttt{Subset-Out}_{T-C}$ с дизъюнктами $\neg x_e \vee \neg x_{e'}$ для всех $e' \in E(V(T-C), s_{t-1})$.

Возьмем произвольное подмножество $T' \subset T$ размера $t-1$. Возьмем дизъюнкт $\texttt{Subset-Out}_{T'}$ и для каждого ребра $e \in E(V(T'), s_{t-1})$ проведем резолюцию с дизъюнктом $\texttt{Kill-Edge}_{C,T}(e)$. Как результат, получим дизъюнкт вида $\bigvee_{e' \in E'} x_{e'}$, где $E' \subseteq E(V(T'), S_t)$. Полученный дизъюнкт ослабим до $\texttt{Subset-Out}_T$.

Дойдя до $t = l + 1$, мы получим пустой дизъюнкт и завершим доказательство.

Оценим размер вывода. По лемме 2.1 вывод всех дизъюнктов $\texttt{Odd-Out}_{C_i}$ имеет размер $O(n^2 \cdot 2^n)$.

$|S| \leq n/2$. Мы считаем, что число компонент связности $|S| + 1$. Потому всех возможных поднаборов T не более $2^{n/2+1}$.

Каждый дизъюнкт $\texttt{Subset-Out}_T$ имеет вывод полиномиального от n размера. В первом случае, мы делаем не более n^2 ослаблений. Во втором — на вывод дизъюнктов Kill-Edge мы тратим не более n^2 резолюций, всего таких дизъюнктов не более n, размер общего вывода всех Kill-Edge — $O(n^3)$. Резолюция $\texttt{Subset-Out}_{T'}$ с дизъюнктами Kill-Edge проводится не более n раз. Заключительное ослабление использует не более n^2 переменных.

Итого, вывод всех дизъюнктов $\texttt{Subset-Out}_T$, включая пустой, имеет размер $O(n^3) \cdot 2^{n/2} + O(n^2 \cdot 2^n) = O(n^2 \cdot 2^n)$.

□

2.3 Верхние оценки на древовидные доказательства в системе Res-Lin

2.3.1 Принцип Дирихле

Теорема 2.3. Для любых $m > n$ формула $\texttt{PHP}_n^m$ имеет дерево линейных расщеплений размера $2^{O(n)}$.

Доказательство. Заметим, что формула $\texttt{PHP}_n^{n+1}$ является подформулой $\texttt{PHP}_n^m$. Поэтому нам достаточно построить дерево линейных расщеплений только для формулы $\texttt{PHP}_n^{n+1}$.

Индукцией по n мы построим дерево линейных расщеплений размера $2^{O(n)}$ для каждой формулы $\mathtt{PHP}_n^{n+1}$. Случай с $n = 1$ тривиален.

Для $n > 1$ мы будем строить дерево линейных расщеплений для формулы $\mathtt{PHP}_n^{n+1}$, используя множество копий дерева для формулы $\mathtt{PHP}_{n/2}^{n/2+1}$. На первой стадии, мы построим дерево T размера $2^{O(n)}$, каждый лист которого будет либо помечен нарушенным дизъюнктом, либо заменен на поддерево для $\mathtt{PHP}_{n/2}^{n/2+1}$.

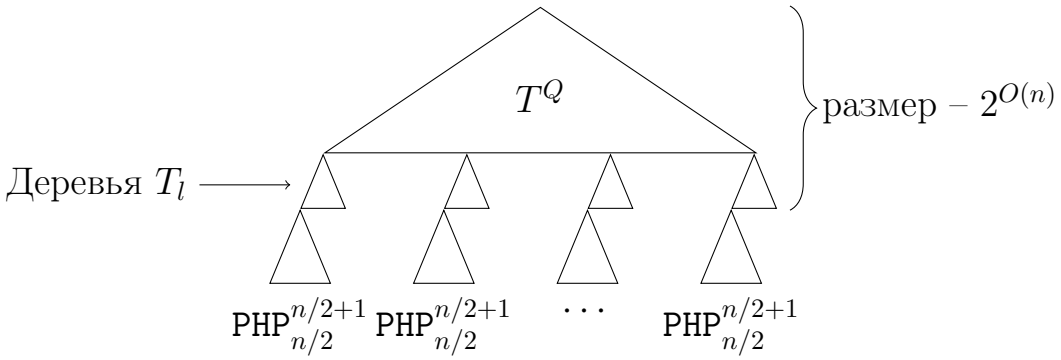

Рис. 2.1: Общая структура дерева линейных расщеплений для формулы $\mathtt{PHP}_n^{n+1}$.

Пусть $L(n)$ — логарифм размера дерева линейных расщеплений для формулы $\mathtt{PHP}_n^{n+1}$. Дерево T имеет размер $2^{O(n)}$. Каждый лист может быть заменен на поддерево размера $2^{L(n/2)}$. Значит,

$$L(n) \leq \log(2^{O(n)} \cdot 2^{L(n/2)}) = O(n) + L(n/2).$$

Т.е. $L(n) = O(n)$, и размер всего дерева — $2^{O(n)}$.

Для построения дерева T мы разделим кроликов на две примерно равные части: левую $L = [1, \lfloor \frac{n+1}{2} \rfloor]$ и правую $R = [\lfloor \frac{n+1}{2} \rfloor + 1, n + 1]$. Для каждой клетки мы определим две линейные формы

$$\begin{aligned} \mathrm{LEFT}(j) &= \textstyle\bigoplus_{i \in L} x_{i,j}, \\ \mathrm{RIGHT}(j) &= \textstyle\bigoplus_{i \in R} x_{i,j}. \end{aligned}$$

Дерево T начинается с полного бинарного дерева T^Q высоты $2n$ с 2^{2n} листьями. Каждая ветвь дерева T^Q соответствует запросам к значениям линейных форм $\mathrm{LEFT}(j)$ и $\mathrm{RIGHT}(j)$ для каждой клетки j.

Дерево T получается из дерева T^Q заменой каждого листа l на поддерево T_l полиномиального размера. Для каждого поддерева T_l, все листья, за исключением, быть может, одного, помечены нарушенными дизъюнктами. Последний лист используется, чтобы подвесить дерево для формулы $\mathtt{PHP}_{n/2}^{n/2+1}$ (см. рисунок 2.1).

Зафиксируем лист l дерева T^Q. Для каждой клетки j, мы имеем фиксированные значения $\mathrm{LEFT}(j)$ и $\mathrm{RIGHT}(j)$. Возможны четыре случая.

1. $\mathrm{LEFT}(j) = 1$, $\mathrm{RIGHT}(j) = 1$.
2. $\mathrm{LEFT}(j) = 0$, $\mathrm{RIGHT}(j) = 1$.
3. $\mathrm{LEFT}(j) = 1$, $\mathrm{RIGHT}(j) = 0$.
4. $\mathrm{LEFT}(j) = 0$, $\mathrm{RIGHT}(j) = 0$.

Мы сначала опишем общую структуру дерева T_l, а затем подробно разберем каждый случай.

Если хотя бы для одной клетки j выполняется случай 1, то в клетке j сидит хотя бы один кролик из левой части и хотя бы один из правой. Мы построим дерево T_l размера $O(n^2)$ так, что каждый лист будет помечен нарушенным дизъюнктом.

Если для всех клеток выполняется один из случаев 2-4, то мы построим дерево τ_j для каждой клетки j. Ключевое свойство дерева τ_j в том, что все листья, за исключением ровно одного, *свободного*, помечены нарушенным дизъюнктом. Дерево T_l — это цепочка деревьев τ_j, где каждое следующее подвешивается к свободному листу предыдущего. Последний свободный лист используется для дерева формулы $\mathtt{PHP}_{n/2}^{n/2+1}$

1. Предположим, для некоторой клетки j выполняется случай 1. Поскольку $\mathrm{LEFT}(j) = 1$ и $\mathrm{RIGHT}(j) = 1$, мы знаем, что как минимум два кролика находятся в клетке j. Мы построим дерево T_l размера $O(n^2)$ в соответствии со следующим алгоритмом.

   Сперва переберем все переменные, которые соответствуют кроликам в левой части. Как только будет найден кролик i_1 с $x_{i_1,j} = 1$,

перейдем на правую часть и повторим процесс. Как только мы найдем кролика i_2 в правой части, оставим текущую вершину листом и пометим ее дизъюнктом $\neg x_{i_1,j} \vee \neg x_{i_2,j}$.

Пусть в текущем листе нет клеток, для которых выполняется случай 1. Для каждой клетки j мы построим дерево τ_j в зависимости от случаев 2, 3 или 4.

2. Пусть для клетки j выполняется случай 2, т.е. $\mathrm{LEFT}(j) = 0$. Каждый лист дерева τ_j, за исключением одного, будет помечен нарушенным дизъюнктом. В последний лист можно будет попасть только в случае, если $x_{i,j} = 0$ для всех $i \in L$, т.е. если ни один кролик из левой части не будет посажен в клетку j. Последний лист будем называть свободным.

   Структура дерева τ_j определяется в соответствии со следующим алгоритмом. Для каждого кролика i из левой части L мы запрашиваем значение $x_{i,j}$. Если кролик i сидит в клетке j ($x_{i,j} = 1$), то поскольку

   $$\mathrm{LEFT}(j) = \bigoplus_{i \in L} x_{i,j} = 0,$$

   существует второй кролик i', который тоже сидит в клетке j. Продолжим запрашивать значения переменных $x_{i,j}$ для левой части, пока не найдем второго кролика i' с $x_{i',j} = 1$. Как только второй кролик найден, вернем нарушенный дизъюнкт $\neg x_{i,j} \vee \neg x_{i',j}$.

   Если

   $$x_{i,j} = 0 \ \ \forall i \in L,$$

   то мы пришли в свободный лист. В этом листе известно, что ни один кролик из левой части не сидит в клетке j.

3. Пусть для клетки j выполняется случай 3, т.е. $\mathrm{RIGHT}(j) = 0$. Дерево τ_j строится по аналогии с предыдущим случаем, с заменой левой части на правую.

4. Пусть для клетки j выполняется случай 4, т.е. и $\mathrm{LEFT}(j) = 0$, и $\mathrm{RIGHT}(j) = 0$. Дерево τ_j строится по аналогии с предыдущими двумя случаями с той разницей, что теперь мы запрашиваем значения $x_{i,j}$ для кроликов из обеих частей. В свободном листе дерева τ_j, мы гарантируем, что ни один кролик не сидит в клетке j.

Как описано выше, мы формируем дерево T_l, объединяя все деревья τ_j в цепочку. Каждое дерево присоединяется к свободному листу предыдущего. Любое дерево τ_j имеет размер $O(n^2)$. Дерево T_l — $O(n^3)$.

По построению цепочка имеет ровно один свободный лист. В этом листе мы гарантируем для каждой клетки j, что

- если $\mathrm{LEFT}(j) = 0$, то ни один кролик из левой части не сидит в клетке j;
- если $\mathrm{RIGHT}(j) = 0$, то ни один кролик из правой части не сидит в клетке j.

Поскольку цепочка формируется только в случае, когда нет клеток с $\mathrm{LEFT}(j) = \mathrm{RIGHT}(j) = 1$, мы можем выделить два непересекающихся множества клеток: множество клеток H_L, в которых сидят кролики только из левой части ($\mathrm{LEFT}(j) = 1$), и множество H_R, в которых сидят кролики только из правой части ($\mathrm{RIGHT}(j) = 1$).

Мы покажем от противного, что как минимум для одной стороны, клеток меньше чем кроликов. Пусть $h_l = |H_L|$, $h_r = |H_R|$ и пусть $h_l \geq |L|$ и $h_r \geq |R|$. Мы знаем, что $h_l + h_r \leq n$. Тогда

$$n \geq h_l + h_r \geq |L| + |R| = n + 1,$$

противоречие.

Поскольку размеры L и R не превышают $\lceil n/2 \rceil$, мы можем воспользоваться деревом линейных расщеплений для формулы $\mathtt{PHP}_{n/2}^{n/2+1}$, используя переменные соответствующей части.

Дерево для формулы $\mathtt{PHP}_{n/2}^{n/2+1}$ мы прикрепляем к последнему свободному листу цепочки. □

2.3.2 Принцип совершенного паросочетания

В этом разделе мы покажем верхнюю оценку на размер дерева линейных расщеплений для принципа совершенного паросочетания. Ицыксон и Соколов [25] показали, что в частном случае, для графа с нечетным числом вершин, размер дерева ограничен полиномом от числа вершин в графе.

Теорема 2.4 ([25])**.** Пусть граф G содержит n вершин и n нечетно. Тогда формула $\mathtt{PMP}_G$ имеет дерево линейных расщеплений полиномиального от n размера.

Мы покажем верхнюю оценку $2^{O(n)}$ на размер дерева линейных расщеплений для графа на n вершинах. При этом n может быть четно. В основе доказательства следующей теоремы мы воспользуемся Теоремой 2.3.

Теорема 2.5. Пусть граф $G = \langle V, E \rangle$ связен, содержит n вершин и не имеет совершенного паросочетания. Тогда формула $\mathtt{PMP}_G$ имеет дерево линейных расщеплений размера $2^{O(n)}$.

Идея доказательства — свести принцип совершенного паросочетания к принципу Дирихле. Используя критерий Татта, мы можем выделить множество вершин S размера l в графе G и m компонент связности $C_1, C_2, \cdots, C_m$ нечетного размера в графе $G - S$. При этом $m > l$.

Каждую компоненту мы отождествим с кроликом. Каждую вершину S — с клеткой. Кролик C_i отправляется в клетку s_j в случае, если число ребер в паросочетании между компонентой связности C_i и вершиной s_j нечетно.

Мы используем дерево линейных расщеплений для принципа Дирихле. Каждый лист дерева либо помечен дизъюнктом, который обнаруживает двух кроликов в одной клетке, либо одного кролика, который никуда не отправляется.

Первый случай соответствует ситуации, когда одной вершине инциденты два ребра из паросочетания. Эти ребра можно найти деревом полиномиального размера.

Второй случай распадается на два подслучая.

- Есть вершина $s_j \in S$ такая, что вершину s_j и компоненту C_i соединяет ненулевое четное число ребер, лежащих в паросочетании. В этом подслучае мы обнаруживаем ребра, используя дерево полиномиального размера.

- Ни одно ребро между C_i и S не лежит в паросочетании. В этом случае, мы используем дерево из теоремы 2.4, чтобы найти нарушенный дизъюнкт на компоненте связности C_i, как на графе нечетного размера.

Доказательство. Для доказательства теоремы 2.5 мы воспользуемся теоремой 2.1. Поскольку в графе G нет совершенного паросочетания, значит, есть подмножество $S \subseteq V$, такое, что число компонент связности нечетного размера в графе $G - S$ больше, чем размер S.

Обозначим вершины множества S как $s_1, s_2, \cdots, s_l$, а соответствующие компоненты как $C_1, C_2, \cdots, C_m$ Для каждой компоненты связности C_i и вершины s_j, мы введем переменную

$$y_{i,j} = \bigoplus_{(u,s_j)\in E, u\in C_i} x_{u,s_j}.$$

$m > l$. Рассмотрим формулу $\mathtt{PHP}_l^m$, основанную на переменных $y_{i,j}$, и дерево линейных расщеплений T_y размера $2^{O(l)}$, как это было сделано в Теореме 2.3. Поскольку $l = O(n)$, то размер дерева будет $2^{O(n)}$.

Мы построим дерево линейных расщеплений T_x для формулы $\mathtt{PMP}_G$, используя структуру дерева T_y для формулы $\mathtt{PHP}_l^m$. Заменим все переменные типа y на линейные формы соответствующих переменных типа x. В некоторых вершинах дерева T_x мы можем получить пустые линейные формы.

Если в вершине v записана пустая форма, то на двух исходящих ребрах будут записаны два уравнения $0 = 1$ и $0 = 0$. Первое уравнение делает систему несовместной, потому соответствующая ветвь может быть обрезана. Ребро со вторым уравнением может быть стянуто, объединяя родителя и ребенка.

Рассмотрим произвольный лист l дерева T_y, помеченный дизъюнктом D_l. Каждый лист l мы заменим на поддерево T_l полиномиального от n размера. Каждое дерево T_l находит нарушенные дизъюнкты формулы $\texttt{PMP}_G$. Структура дерева зависит от дизъюнкта D_l. Возможны два случая.

1. Дизъюнкт D_l имеет вид $\neg y_{i_1,j} \vee \neg y_{i_2,j}$. Значит, существуют две компоненты связности C_{i_1} и C_{i_2} и вершина s_j такие, что $y_{i_1,j} = 1$ и $y_{i_2,j} = 1$.

2. Дизъюнкт D_l имеет вид $\bigvee_{j \in [l]} y_{i,j}$. Тогда существует компонента связности C_i такая, что $y_{i,j} = 0$ для всех вершин $s_j \in S$.

Рассмотрим эти два случая.

1. У нас есть как минимум два ребра, которые приходят в вершину s_j. Структура дерева T_l определяется в соответствии со следующим алгоритмом. Проверим каждое ребро e между вершиной s_j и компонентой C_{i_1}. Как только найдется ребро e_1 с $x_{e_1} = 1$, перейдем ко второй компоненте и повторим поиск. Как только найдем второе ребро e_2 с $x_{e_2} = 1$, вернем нарушенный дизъюнкт $\neg x_{e_1} \vee \neg x_{e_2}$. Оба ребра существуют, поскольку

$$\begin{aligned} y_{i_1,j} &= \bigoplus\nolimits_{(u,s_j) \in E, u \in C_{i_1}} x_{u,s_j} = 1; \\ y_{i_2,j} &= \bigoplus\nolimits_{(u,s_j) \in E, u \in C_{i_2}} x_{u,s_j} = 1. \end{aligned}$$

   Дерево T_l имеет размер $O(n^2)$.

2. Мы знаем, что $y_{i,j} = 0$ для каждой вершины $s_j \in S$. Это значит, что либо $x_{u,s_j} = 0$ для всех вершин $u \in C_i$, $s_j \in S$, если $(u, s_j) \in E$, либо

существует вершина s_j и как минимум две переменных $x_{u_1,s_j} = 1$ и $x_{u_2,s_j} = 1$, где $u_1, u_2 \in C_i$.

Определим структуру дерева T_l в соответствии со следующим алгоритмом. Для каждого ребра $e \in E(S, C_i)$ запросим значение переменной x_e. Если для некоторого ребра e_1 между вершиной $u_1 \in C_i$ и вершиной $s_j \in S$, значение $x_{e_1} = 1$, то существует второе ребро e_2 между вершинами $u_2 \in C_i$ и s_j с $x_{e_2} = 1$. Начнем искать второе ребро. Как только ребро найдено, вернем нарушенный дизъюнкт $\neg x_{e_1} \vee \neg x_{e_2}$.

Построенное дерево имеет один свободный лист l'. В листе l' мы знаем, что ни одно ребро, исходящее из компоненты связности C_i, не было взято в паросочетание. Мы рассмотрим компоненту связности C_i как граф нечетного размера и по Теореме 2.4 получим дерево T_{C_i} полиномиального размера. Дерево T_{C_i} прикрепим к свободному листу l', завершив описание структуры дерева T_l.

□

Глава 3

Резолюционные системы доказательств над произвольным алфавитом

В этой главе мы рассмотрим резолюционные доказательства в системе NG-Res.

В разделе 3.1 мы рассмотрим нижние оценки на резолюционные доказательства в системе NG-Res через минимальную ширину доказательства. В теореме 3.1 мы покажем, что для CSP ϕ, чувствительной к подстановкам, минимальная ширина доказательства $W(\phi \vdash 0) \geq e_2(\phi) - 1$.

В разделе 3.2 мы определим понятие обобщенных цейтинских формул $\mathtt{Ts}(G, f)$. В лемме 3.2 мы покажем, что невыполнимые цетинские формулы минимально невыполнимы. В лемме 3.3 мы сформулируем и докажем критерий невыполнимости.

Далее мы применим оценки из раздела 3.1 и покажем, что если граф G обладает расширительной способностью $e_2(G)$ и его максимальная степень равна d, то

- $S_T(\mathtt{Ts}(G, f)) \geq 2^{e_2(G)-d-1}$;
- $S(\mathtt{Ts}(G, f)) \geq \exp\left(\Omega\left(\frac{(e_2(G)-d-1)^2}{n}\right)\right)$.

В разделе 3.3 мы усилим нижнюю оценку на размер древовидного резолюционного доказательства в системе NG-Res для невыполнимой

цейтинской формулы. Мы покажем, что $S_T(\mathtt{Ts}(G, f)) \geq k^{e_2(G)-d}$.

В разделе 3.4 мы предложим верхнюю оценку на размер древовидного резолюционного доказательства в системе NG-Res для произвольной CSP ϕ через расширительную способность графа. Каждой CSP ϕ мы сопоставим граф зависимостей $G = \langle V, E\rangle$ с числом вершин, равным числу ограничений в CSP ϕ. Две вершины графа мы соединим числом ребер, равным числу общих переменных для соответствующих ограничений. Размер доказательства будет ограничен сверху величиной $k^{e(H)\cdot\log_{2/3}|V|}$, где H — некоторый подграф графа G. В случае обобщенной цейтинской формулы $\mathtt{Ts}(G, f)$ — граф зависимостей будет изоморфен графу G.

Результаты этой главы опубликованы в работе [35].

3.1 Нижняя оценка на минимальную ширину доказательств в системе NG-Res

Теорема 3.1. Пусть $\phi = \langle X, D, S\rangle$ — минимально невыполнимая CSP, удовлетворяющая следующему свойству.

- Пусть произвольная подстановка ρ нарушает некоторое ограничение $f \in S$. Для любой переменной x, от которой зависит f, найдется значение $a \in D$ такое, что если заменить значение $\rho(x)$ на a, ограничение f будет выполнено.

Тогда $W(\phi \vdash 0) \geq e_2(\phi) - 1$.

Доказательство. Будем говорить, что запрещающий набор N семантически следует из $F \subseteq S$, если любая подстановка, выполняющая F, выполняет N. Семантическое следствие обозначим $F \vDash N$. Мы определим меру Бен-Сассона и Вигдерсона на множестве всех запрещающих наборов. Для запрещающего набора N определим $\mu(N) = \min\{|F| \mid F \subseteq S, F \vDash N\}$. Следующие свойства следуют непосредственно из определения.

- $\mu(N) \leq 1$ для любого запрещающего набора N, который соответствует ограничению из ϕ.

- $\mu(\neg(\texttt{True})) = |S|$ по минимальной невыполнимости.
- Если N — резольвента запрещающих наборов $\{N_a\}_{a \in D}$, то $\mu(N) \leq \sum_{a \in D} \mu(N_a)$.

Лемма 3.1. Пусть F минимальное множество ограничений, из которых семантически следует запрещающий набор N. Тогда запрещающий набор N зависит от каждой из переменных ∂F.

Доказательство. F — минимальное множество ограничений, из которых семантически следует N. Для каждого ограничения f существует подстановка ρ_f, которая нарушает N, нарушает f, но не нарушает любое другое ограничение $g \in F - f$.

Возьмем переменную $x \in \partial F$. Пусть $f \in F$ — единственное ограничение, которое зависит от x. По условию теоремы существует такое значение $a \in D$, что если поменять значение $\rho_f(x)$ на a, то ограничение f будет выполнено, все ограничения $F - f$ будут выполнены по выбору ρ_f, и, значит, запрещающий набор N будет выполнен. Следовательно, N зависит от x. □

Любое резолюционное доказательство CSP ϕ содержит запрещающий набор N такой, что

- N является резольвентой множества запрещающих наборов $\{N_a\}_{a \in D}$;
- $\mu(N) > \frac{1}{3}|S|$;
- $\mu(N_a) \leq \frac{1}{3}|S| \ \ \forall a \in D$.

Пусть F_a — минимальное подмножество ограничений, из которого семантически следует запрещающий набор N_a. Поскольку $|F_a| \leq \frac{1}{3}|S|$, мы можем выбрать подмножество значений $D' \subseteq D$ и $F' = \bigcup_{a \in D'} F_a$ так, что $\frac{1}{3}|S| \leq |F'| \leq \frac{2}{3}|S|$.

По лемме 3.1 запрещающий набор N_a зависит от всех переменных ∂F_a. Значит,

$$\partial F' \subseteq \bigcup_{a \in D'} \partial F_a \subseteq \bigcup_{a \in D'} \mathrm{Vars}(N_a),$$

где $\mathrm{Vars}(N_a)$ — множество переменных, от которых зависит N_a.

Поскольку запрещающий набор N — резольвента наборов $\{N_a\}_{a \in D}$, то $\bigcup_{a \in D'} \mathrm{Vars}(N_a) \backslash \{x\} \subseteq \mathrm{Vars}(N)$. Значит, $\partial F' \backslash \{x\} \subseteq \mathrm{Vars}(N)$. По выбору F' размер $\partial F'$ как минимум $e_2(\phi)$. Значит, число переменных в N как минимум $e_2(\phi) - 1$. □

3.2 Обобщенные цейтинские формулы

Мы определим обобщенные цетинские формулы на алфавите $\mathbb{Z}_k$.

Пусть есть граф $G = \langle V, E \rangle$ и функция $f : V \to \mathbb{Z}_k$. Сопоставим каждому ребру $e \in E$ переменную x_e. Кроме того, сопоставим ребру $(u, v) \in E$ два коэффициента $\gamma_{u,v}$ и $\gamma_{v,u}$, которые по модулю равны единице, но различаются по знаку. Переменные x_e для $e \in E$ могут принимать любые значения из $\mathbb{Z}_k$. Для каждой вершины $v \in V$ определим ограничение R_v вида

$$\sum_{(v,u) \in E} \gamma_{v,u} x_{(v,u)} = f(v) \pmod{k}.$$

Отметим, что каждая переменная x_e встречается ровно в двух уравнениях, но с разными по знаку γ.

Полученную CSP назовем *обобщенной цейтинской формулой* $\mathtt{Ts}(G, f)$.

Цейтинские формулы замкнуты относительно подстановок отдельных переменных. Если в формуле $\mathtt{Ts}(G, f)$ переменной $x_{(u,v)}$ подставить значение α, мы получим снова цейтинскую формулу $\mathtt{Ts}(G - e, f')$, где f' отличается от f только в вершинах u и v.

Лемма 3.2. Пусть граф G связен и содержит n вершин. Рассмотрим обобщенную цейтинскую формулу $\mathtt{Ts}(G, f) = \langle X, \mathbb{Z}_k, S \rangle$ и удалим про-

извольное ограничение $R_v \in S$. Тогда CSP $\mathtt{Ts}(G, f)_{-R_v} = \langle X, \mathbb{Z}_k, S - R_v \rangle$ выполнима.

Доказательство. Рассмотрим остовное дерево T графа G с корнем в вершине v. Пусть T_u поддерево дерева T с корнем в вершине u. Для каждого ребра $(u, w) \in T$, где w — родитель u определим значение $\rho(x_{u,w}) = \gamma_{u,w} \cdot \sum_{q \in T_u} f(q)$. Для всех остальных переменных определим значение ρ равным нулю.

Покажем, что подстановка ρ выполняет $\mathtt{Ts}(G, f)_{-R_v}$. Рассмотрим произвольную некорневую вершину u, отличную от v. Пусть вершина имеет детей $c_1, c_2, \cdots, c_t$ и родителя p в дереве T. Тогда

$$\begin{aligned}
&\textstyle\sum_{(u,w) \in E} \gamma_{u,w} \cdot \rho(x_{(u,w)}) = \\
&\gamma_{u,p}^2 \cdot \textstyle\sum_{q \in T_u} f(q) + \sum_{(u,c_i): i \in [t]} \gamma_{u,c_i} \cdot \gamma_{c_i,u} \cdot \sum_{q \in T_{c_i}} f(q) = \\
&\textstyle\sum_{q \in T_u} f(q) - \sum_{(u,c_i): i \in [t]} \sum_{q \in T_{c_i}} f(q) = \\
&f(u).
\end{aligned}$$

Т.е. все ограничения, кроме R_v, выполнены. □

Лемма 3.3. Пусть граф G связен и содержит n вершин. Обобщенная цейтинская формула $\mathtt{Ts}(G, f)$ выполнима тогда и только тогда, когда

$$\sum_{v \in V} f(v) = 0 \pmod{k}.$$

Доказательство. Запишем цейтинскую формулу как систему линейных уравнений $Ax = f$ над кольцом $\mathbb{Z}_k$. Мы пронумеруем вершины и отождествим вектор $f \in \mathbb{Z}_k^n$ с функцией $f : V \to \mathbb{Z}_k$.

Уберем первое уравнение. По лемме 3.2 полученная система $A'x = f'$ имеет решение.

Пусть $a_1^T, a_2^T, \cdots, a_n^T$ — вектора-строки матрицы A. По построению цейтинской формулы $\sum_{i \in [n]} a_i^T = 0$, т.е. вектора линейно зависимы. $a_1^T = (-1) \cdot \sum_{i \in [2,n]} a_i^T$. Значит, для любого решения x_0 системы $A'x = f'$ верно, что

$$a_1^T \cdot x_0 = (-1) \cdot \sum_{i \in [2,n]} a_i^T \cdot x_0 = - \sum_{i \in [2,n]} f_i.$$

Поскольку любое решение $Ax = f$ является решением $A'x = f'$, то система разрешима тогда и только тогда, когда $f_1 = -\sum_{i\in[2,n]} f_i$ или $\sum_{v\in V} f_v = 0$ в кольце $\mathbb{Z}_k$. □

Следствие 3.1. Если обобщенная цейтинская формула $\mathtt{Ts}(G, f)$ невыполнима, то $W(\mathtt{Ts}(G, f) \vdash 0) \geq e(G) - 1$.

Доказательство. Достаточно показать, что формула $\mathtt{Ts}(G, f)$ удовлетворяет условиям теоремы 3.1. По лемме 3.2 формула $\mathtt{Ts}(G, f)$ минимально невыполнима.

Пусть подстановка ρ нарушает ограничение $\sum_{(u,v)\in E} \gamma_{u,v} x_{(u,v)} = f(v) \pmod k$. Можно поменять значение ρ для любой переменной из ограничения так, чтобы оно оказалось выполненным. □

Пусть максимальная степень графа G ограничена константой d. Каждый запрещающий набор, соответствующий ограничению из $\mathtt{Ts}(G, f)$, использует не более d переменных. Следующие нижние оценки на размер резолюционных доказательств для обобщенных цейтинских формул следуют из теоремы 1.4 и следствия 3.1.

1. $S_T(\mathtt{Ts}(G, f)) \geq 2^{e(G)-d-1}$.
2. $S(\mathtt{Ts}(G, f)) \geq \exp\left(\Omega\left(\frac{(e(G)-d-1)^2}{n}\right)\right)$.

Уркухарт [16] построил семейство 3–регулярных связных графов $\{G_n = \langle V_n, E_n\rangle\}_{n\geq 1}$, каждый из которых имеет расширительную способность $e(G) = \Omega(V_n)$ и $|V_n| = n$. Таким образом, можно получить экспоненциальные нижние оценки на размеры доказательств в системе NG-Res.

3.3 Нижняя оценка на обобщенные цейтинские формулы

В этом разделе мы усилим нижнюю оценку на размер древовидного резолюционного доказательства для обобщенных цейтинских формул. Пусть

есть граф $G = \langle V, E \rangle$; максимальная степень графа ограничена константой d; переменные цейтинской формулы принимают значения из $\mathbb{Z}_k$ и есть некоторая функция $f : V \to \mathbb{Z}_k$ такая, что $\sum_{v \in V} f(v) \neq 0 \bmod k$.

Мы покажем, что размер древовидного доказательства не меньше $k^{e(G)-d}$.

3.3.1 Сокращенное дерево расщеплений

Пусть граф $G = \langle V, E \rangle$ связен. Определим функцию C на множестве связных графов и докажем, что она равна размеру минимального дерева расщеплений для CSP $\mathtt{Ts}(G, f)$:

$$C(G) = \begin{cases} 1, \text{ если } |V| = 1; \\ \min_{e \in E} C_{\text{help}}(G - e) + 1, \end{cases}$$

где функция $C_{\text{help}}(G)$ определена на графах с не более чем двумя компонентами связности:

$$C_{\text{help}}(G) = \begin{cases} k \cdot C(G), \text{ если } G \text{ связен}; \\ (k-1)C(G_1) + C(G_2), \text{ иначе.} \end{cases}$$

При этом G_1 и G_2 — компоненты связности графа G, и $C(G_1) \leq C(G_2)$.

Лемма 3.4. Пусть граф G связен. Размер минимального дерева расщеплений для невыполнимой цейтинской формулы $\mathtt{Ts}(G, f)$ не зависит от f и равен $C(G)$.

Доказательство. Докажем лемму по индукции по числу ребер в графе. База индукции тривиальна. Рассмотрим произвольный граф $G = \langle V, E \rangle$ и функцию $f : V \to \mathbb{Z}_k$.

Пусть корень оптимального дерева расщеплений T помечен переменной x_e. Разберем два случая для функции C_{help}.

1. Граф $G - e$ связен. В таком случае, в каждом ребенке корня дерева T находится поддерево для цейтинской формулы $\mathtt{Ts}(G - e, f')$, где

функция f' отличается от f только на концах ребра e. По индукционному предположению, размер таких деревьев равен $C(G-e)$. Всего детей — k, следовательно, размер дерева составит $k \cdot C(G-e)+1$.

2. Граф $G-e$ разбивается на две компоненты связности G_1 и G_2. После подстановки $x_e = a$, формула $\texttt{Ts}(G-e, f'_a)$ разбивается на две независимые подформулы для графов G_1 и G_2 с функциями $f_{1,a}$ и $f_{2,a}$ соответственно. Таким образом, для каждого поддерева достаточно опровергнуть одну из подформул.

   Пусть $C(G_1) \leq C(G_2)$. Заметим, что существует ровно одно значение a для x_e, при котором формула $\texttt{Ts}(G_1, f_{1,a})$ становится выполнимой. Тогда, по невыполнимости, существует дерево расщеплений для $\texttt{Ts}(G_2, f_{2,a})$. Минимизируя суммарный размер всех деревьев, получим $(k-1) \cdot C(G_1) + C(G_2) + 1$.

□

Лемма 3.4 позволяет представить дерево расщеплений в компактном виде. Определим *сокращенное дерево расщеплений* как корневое дерево, каждая вершина которого помечена связным графом. Для цейтинской формулы $\texttt{Ts}(G, f)$ дерево выглядит следующим образом.

1. Корень дерева помечен графом G.

2. Каждый лист дерева помечен графом с ровно одной вершиной графа G.

3. Каждая внутренняя вершина имеет либо одного, либо двух детей.

4. Пусть вершина v сокращенного дерева расщеплений помечена графом G_v. Если вершина v содержит одного ребенка, то он помечен графом $G_v - e$ для некоторого ребра e, и граф $G_v - e$ связен. Если вершина v имеет двух детей, то каждый из них помечен компонентой связности графа $G - e$, и ребро e — мост.

Определим функцию q на вершинах сокращенного дерева расщеплений.

$$q(v) = \begin{cases} 1, \text{ если вершина } v \text{ — лист;} \\ k \cdot q(u) + 1, \text{ если } v \text{ имеет единственного ребенка } u; \\ (k-1) \cdot q(u_1) + q(u_2) + 1, \text{ если } v \text{ имеет двух детей } u_1, u_2. \end{cases}$$

При этом $q(u_1) \leq q(u_2)$

Для сокращенного дерева расщеплений T определим функцию $Q(T) = q(r)$, где r — корень дерева T. Каждому сокращенному дереву расщеплений соответствует некоторое обычное дерево расщеплений. Поскольку оптимальному дереву расщеплений для формулы $\mathtt{Ts}(G, f)$ можно сопоставить сокращенное, то

$$C(G) = \min_T Q(T),$$

где минимум берется по всем сокращенным деревьям расщеплений для цейтинской формулы $\mathtt{Ts}(G, f)$.

3.3.2 Нижняя оценка

Введем понятие ширины для сокращенного дерева расщеплений.

Пусть граф $G = \langle V, E \rangle$ связен, и цейтинская формула $\mathtt{Ts}(G, f)$ невыполнима. Пусть T — сокращенное дерево расщеплений для $\mathtt{Ts}(G, f)$. Рассмотрим вершину v дерева T, помеченную графом $G_v = \langle V_v, E_v \rangle$. Пусть $E_{\text{ext}} = \{(x, y) \in E \mid x \in V_v \vee y \in V_v\}$ — множество ребер, имеющих хотя бы один конец в графе G_v. Определим ширину $w(v) = |E_{\text{ext}} \backslash E_v|$, т.е. как число уже удаленных ребер, инцидентных хотя бы одной вершине из V_v. Ширина дерева T:

$$W(T) = \max_{v \in T} w(v),$$

где максимум берется по всем вершинам T.

Лемма 3.5. Пусть граф $G = \langle V, E \rangle$ связен и имеет расширительную способность $e(G)$; цейтинская формула $\mathtt{Ts}(G, f)$ невыполнима, и T — произвольное сокращенное дерево расщеплений для $\mathtt{Ts}(G, f)$. Тогда

$$W(T) \geq e(G).$$

Доказательство. Дерево T содержит вершину, помеченную графом $G_v = \langle V_v, E_v \rangle$, такую, что

- $|V_v| \geq \frac{2}{3} \cdot |V|$;
- вершина v имеет двух детей u_1, u_2;
- $|V_{u_i}| \leq \frac{2}{3} \cdot |V|$ для $i \in \{1, 2\}$.

Хотя бы один ребенок, пусть u_1, помечен графом графом $G_{u_1} = \langle V_{u_1}, E_{u_1} \rangle$ таким, что $\frac{1}{3} \cdot |V| \leq |V_{u_1}| \leq \frac{2}{3} \cdot |V|$. По определению расширительной способности графа $w(u_1) \geq e(G)$. □

Лемма 3.6. Пусть граф G связен и степени его вершин ограничены константой d. Пусть T — сокращенное дерево расщеплений для цейтинской формулы $\texttt{Ts}(G, f)$. Тогда существует сокращенное дерево расщеплений T' для $\texttt{Ts}(G, f)$ такое, что $W(T') \leq d + \log_k Q(T)$.

Доказательство. Доказательство будет по индукции по числу вершин в дереве T. Мы покажем, что если $Q(T) \leq k^b$, то существует дерево T' с $W(T') \leq d + b$. База индукции очевидна.

Пусть r — корень дерева T. Построим дерево T' на основе дерева T. Построение зависит от числа детей корня r.

1. Корень r имеет одного ребенка v. Пусть T_v — поддерево дерева T с корнем в вершине v. Если $Q(T) \leq k^b$, то $Q(T_v) \leq k^{b-1}$. По индукции, для дерева T_v существует дерево T'_v такое, что $W(T') \leq b - 1 + k$. Построим дерево T', взяв корень r и прикрепив к нему поддерево T'_v. Тогда $W(T') \leq (b - 1 + k) + 1 = b + k$.

2. Корень r имеет двух детей v_1 и v_2. Пусть T_1 и T_2 — соответствующие поддеревья; G_1 и G_2 — графы, которыми помечены вершины v_1 и v_2. Пусть также ребро e — мост графа G, соединяющий вершины z подграфа G_1 с вершиной y подграфа G_2.

   Мы считаем, что $Q(T_1) \leq Q(T_2)$. По определению Q:

$$Q(T) = (k - 1) \cdot Q(T_1) + Q(T_2).$$

Мы знаем, что $k \cdot Q(T_1) \leq Q(T)$. Поскольку $Q(T) \leq k^b$, то $Q(T_1) \leq k^{b-1}$ и $Q(T_2) \leq k^b$. По индукции, существуют деревья T_1' и T_2' для графов G_1 и G_2 такие, что $W(T_1') \leq d + b - 1$ и $W(T_2') \leq d + b$. Мы построим дерево T' с шириной $W(T') \leq d + b$, используя деревья T_1' и T_2'.

Перестроим дерево T_2'. Пусть вершина $v \in T_2$ помечена графом G_v, и y лежит в G_v. Прикрепим к вершине y граф G_1 на ребре e. Пусть l — лист дерева T_2', помеченный вершиной y. После перестроения метка листа l будет графом G_1, к которому присоединена вершина y через ребро e. Проведем расщепление по переменной x_e в этой вершине. Один ребенок-лист будет помечен подграфом из единственной вершины y, а второй — графом G_1. Присоединим ко второму ребенку дерево T_1'. Таким образом, мы получаем дерево T' с шириной $W(T') \leq \max\{W(T_2'), W(T_1') + 1, d\} \leq d + b$.

□

Следствие 3.2. $e(G) \leq d + \log_k C(G)$

Теорема 3.2. Пусть граф G связен и имеет максимальную степень не больше d. Тогда размер минимального древовидного доказательства в NG-Res для невыполнимой цейтинской формулы $\mathtt{Ts}(G, f)$ не меньше $k^{e(G)-d}$.

3.4 Верхняя оценка на древовидные резолюционные доказательства

Рассмотрим произвольную CSP $\phi = \langle X, D, S \rangle$. Пусть $|D| = k$. Для любого ограничения $g \in S$, обозначим множество переменных, от которых зависит g, через $\mathrm{Vars}(g)$.

Мы построим граф зависимостей $G = \langle V, E \rangle$ для CSP ϕ. Вершины графа соответствуют ограничениям S. Два ограничения g_i и g_j соединяются $|\mathrm{Vars}(g_i) \cap \mathrm{Vars}(g_j)|$ ребрами, где каждое ребро соответствует общей переменной g_i и g_j.

Заметим, что граф зависимостей для обобщенной цейтинской формулы $\mathtt{Ts}(G, f)$ совпадает с графом G.

Лемма 3.7. Пусть CSP ϕ невыполнима; граф $G = \langle V, E \rangle$ — граф зависимостей ϕ. Тогда в графе G есть подграф H с расширительной способностью

$$e(H) \geq \frac{\log_k S_T(\phi)}{\log_{\frac{3}{2}} |V|},$$

где $S_T(\phi)$ — размер минимального древовидного доказательства ϕ в системе NG-Res.

Доказательство. Построим дерево расщеплений T, и соответственно древовидное доказательство, для формулы ϕ. Дерево расщеплений T определим в соответствии со следующим алгоритмом $A(\phi)$.

1. Возьмем граф зависимостей G для CSP ϕ.
2. Найдем минимальный сбалансированный разрез U такой, что
$$\frac{1}{3} \cdot |V| \leq |U| \leq \frac{2}{3} \cdot |V|.$$
3. Проведем последовательно расщепление для каждой переменной, соответствующей ребру в разрезе $E(U, V \backslash U)$.
4. После расщепления граф разбивается на несколько компонент связности. Алгоритм выбирает любую компоненту, которой соответствует невыполнимая CSP, и запускается рекурсивно.

Каждую вершину в дереве T, соответствующую первому ребру в разрезе $E(U, V \backslash U)$, пометим как интересную. Заметим, что на пути от корня до любого листа находится не более $\log_{\frac{3}{2}} |V|$ интересных вершин.

Пусть высота всего дерева h. Тогда размер дерева $|T| \leq k^h$. Т.е. $h \geq \log_k |T|$. Заметим, что в дереве T найдется интересная вершина v такая, что

- вершина v помечена CSP ϕ_v;

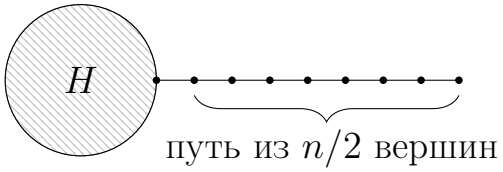

Рис. 3.1: Подграф H с большой расширительной способностью и хвост из $n/2$ вершин.

- граф зависимостей H для CSP ϕ_v имеет минимальный разрез с $h/\log_{\frac{3}{2}}|V|$ ребрами.

Т.е.

$$e(H) \geq \frac{h}{\log_{\frac{3}{2}}|V|} \geq \frac{\log_k|T|}{\log_{\frac{3}{2}}|V|} \geq \frac{\log_k S_T(\phi)}{\log_{\frac{3}{2}}|V|}.$$

□

Теорема 3.3. Пусть ϕ — невыполнимая CSP с графом зависимостей $G = \langle V, E\rangle$, определенная над алфавитом размера k. Тогда в графе G есть подграф H такой, что

$$S_T(\phi) \leq k^{e(H)\cdot\log_{\frac{3}{2}}|V|},$$

где $S_T(\phi)$ — размер минимального древовидного доказательства ϕ в системе NG-Res.

Следствие 3.3. Пусть $\mathtt{Ts}(G, f)$ — невыполнимая цейтинская формула над алфавитом размера k, построенная на связном графе $G = \langle V, E\rangle$. Тогда в графе G есть подграф H такой, что

$$S_T(\mathtt{Ts}(G, f)) \leq k^{e(H)\cdot\log_{\frac{3}{2}}|V|}.$$

Доказательство. Граф зависимостей цейтинской формулы $\mathtt{Ts}(G, f)$ изоморфен графу G. По теореме 3.3 в графе G есть подграф H такой, что

$$S_T(\mathtt{Ts}(G, f)) \leq k^{e(H)\cdot\log_{\frac{3}{2}}|V|}.$$

□

Отметим, что для верхней оценки важна расширительная способность именно *подграфа* графа G. Можно взять граф G, у которого расширительная способность $e(G) = 1$, но при этом невыполнимая цейтинская формула $\mathtt{Ts}(G, f)$ имеет большое резолюционное доказательство. В качестве примера, можно взять граф H из $n/2$ вершин с большой расширительной способностью и присоединить к нему хвост из $n/2$ вершин (см. рисунок 3.1).

Глава 4

Доказательства с правилом сдвига

В этой главе мы получим верхние и нижние оценки для систем доказательств с правилом сдвига.

В разделе 4.1 мы определим специальный тип подвижных формул, зависящих от бесконечного числа переменных, и расширение классических систем доказательств через правило сдвига.

В разделе 4.2 мы покажем нижние экспоненциальные оценки для систем доказательств с правилом сдвига. Пусть ϕ — пропозициональная формула в КНФ, которая зависит от n переменных. Рассмотрим систему доказательств П: резолюционную, секущих плоскостей или полиномиального исчисления. Мы построим подвижную формулу Ψ на основе ϕ и покажем, что если формула ϕ имеет минимальный вывод размера $S_{\Pi}(\phi)$ в системе П, то размер вывода подвижной формулы Ψ в системе Shift-П не меньше $\Omega\left(\frac{S_{\Pi}(\phi)}{f_{\Pi}(n)}\right)$, где $f_{\Pi}(n)$ — полином, зависящий от системы П.

Используя известные результаты о нижних оценках для классических систем, мы покажем нижние экспоненциальные оценки для соответствующих расширений с правилом сдвига.

В разделе 4.3 мы покажем, что размеры вывода в системах доказательств с правилом сдвига и без него могут различаться экспоненциально. Мы построим семейство подвижных формул $\{\Phi_n\}_{n\in\mathbb{N}}$, для которых

любое классическое доказательство будет иметь размер $\Omega(2^n)$, а доказательство в системе Shift-Res будет полиномиального от n размера. При этом формула Φ_n будет содержать полиномиальное от n число подвижных дизъюнктов.

Регулярные выражения. В этой главе мы будем активно оперировать множествами строк, обладающих определенной формой. Мы будем задавать их регулярными выражениями.

- Выражение a для символа a задает строку из единственного символа a.
- Выражение $\square$ задает строку из одного произвольного символа.
- Выражение $\{a_1, \cdots, a_k\}$ задает строку из одного символа a_i для некоторого $i \in [k]$.
- Конкатенация двух регулярных выражений rs — задает конкатенацию двух строк. Первая строка согласуется с выражением r, вторая — с выражением s.
- $(s)^n$ — конкатенация n строк, каждая из которых согласуется с выражением s. В случае, когда $n = \infty$, мы берем бесконечную в обе стороны конкатенацию.

Результаты этой главы опубликованы в работе [34].

4.1 Подвижные формулы и системы доказательств с правилом сдвига

Мы определим специальный тип формул на бесконечном множестве переменных $\Gamma = \{x_i\}_{i \in \mathbb{Z}}$.

Пусть ϕ — пропозициональная формула в КНФ, которая зависит от конечного числа переменных $x_1, x_2, \cdots, x_n$. Для каждого дизъюнкта $C = \bigvee_{i \in I} x_i^{\sigma_i}$ формулы ϕ мы определим сдвиг на j позиций для

любого $j \in \mathbb{Z}$ как дизъюнкт $C_{\to j} = \bigvee_{i \in I} x_{i+j}^{\sigma_i}$. Определим подвижный дизъюнкт $\mathrm{Shifts}(C)$ как множество всех сдвигов дизъюнкта C для всех возможных $j \in \mathbb{Z}$. Отметим, что подвижный дизъюнкт зависит от бесконечного множества переменных Γ. Определим *подвижную формулу* $\Phi = \mathrm{Shifts}(\phi)$ как конъюнкцию подвижных дизъюнктов $\mathrm{Shifts}(C)$ для каждого дизъюнкта C формулы ϕ. Мы будем называть формулу Φ подвижным расширением формулы ϕ.

Мы считаем, что подвижная формула Φ выполнима, если существует подстановка $\rho : \Gamma \to \{0, 1\}$, выполняющая каждый дизъюнкт формулы.

Невыполнимость подвижной формулы Φ можно доказать, используя классическую резолюционную систему доказательств [34]. Однако структура формулы позволяет ввести дополнительное правило вывода — правило сдвига:

$$\frac{C}{C_{\to j}},$$

где C — ранее выведенный дизъюнкт, а $C_{\to j}$ — сдвиг на j позиций для любого $j \in \mathbb{Z}$.

Систему резолюционных доказательств со сдвигом обозначим Shift-Res. Для любой классической системы доказательств П, оперирующей последовательностью предикатов, определим систему Shift-П с правилом вывода

$$\frac{P(x_{i_1}, x_{i_2}, \cdots, x_{i_k})}{P(x_{i_1+j}, x_{i_2+j}, \cdots, x_{i_k+j})},$$

где P — некоторый предикат. Для секущих плоскостей — это линейное неравенство, для полиномиального исчисления — полиномиальное уравнение.

Эквивалентность задаче избегаемости. Выполняющий набор можно рассматривать как бесконечную в обе стороны строку, а подвижные дизъюнкты — как множество запрещенных подстрок. Запрещенные подстроки мы будем записывать тремя символами: 0, 1 и $\square$, где символ $\square$ означает пропуск.

Мы сопоставим запрещенной строке $s = a_1 a_2 \cdots a_k$ подвижный дизъюнкт $\mathrm{Shifts}(C)$ для $C = \bigvee_{i:a_i=0} x_i \vee \bigvee_{i:a_i=1} \neg x_i$. Подвижному дизъюнкту $\mathrm{Shifts}(\bigvee_{i\in I} x_i^{\sigma_i})$, где I — конечное множество индексов, сопоставим запрещенную строку $s = a_1 a_2 \cdots a_l$ длины $l = \max I - \min I + 1$, где

$$a_i = \begin{cases} 0, & \text{если } (i - 1 + \min I) \in I \text{ и } \sigma_{i-1+\min I} = 0; \\ 1, & \text{если } (i - 1 + \min I) \in I \text{ и } \sigma_{i-1+\min I} = 1; \\ \square, & \text{иначе,} \end{cases}$$

для $i \in [l]$.

Строке $0\square\square 1$ будет соответствовать подвижный дизъюнкт $\mathrm{Shifts}(x_1 \vee \neg x_4)$, и наоборот.

Вопрос выполнимости подвижной формулы эквивалентен вопросу существованию бесконечной в обе стороны строки, избегающей соответствующего множества запрещенных подстрок.

4.2 Нижние оценки на размер доказательств с правилом сдвига

Пусть ϕ_n — невыполнимая пропозициональная формула в КНФ, которая зависит от переменных $x_1, x_2, \cdots, x_n$. В этом разделе мы покажем, как построить подвижную формулу Ψ_n на основе ϕ_n такую, что если в системе Π размер вывода формулы ϕ_n составляет $S_\Pi(\phi_n)$, то размер вывода формулы Ψ_n в системе Shift-Π будет не меньше $\Omega\left(\frac{S_\Pi(\phi_n)}{f_\Pi(n)}\right)$, f_Π — некоторый полином, зависящий от системы доказательств Π.

Основная идея доказательства — построить такую подвижную формулу Ψ_n, что из доказательства в системе Shift-Π формулы Ψ_n можно извлечь доказательство формулы ϕ_n в системе Π, увеличив размер не более чем в $f_\Pi(n)$ раз.

В подразделе 4.2.1 мы построим формулу Ψ_n на основе ϕ_n. В подразделе 4.2.2 мы определим понятие устойчивости к замене переменных для классической системы доказательств и покажем, что если система Π устойчива, то формула Ψ_n имеет вывод в системе Shift-Π размера не

меньше $\Omega\left(\frac{S_{\Pi}(\phi_n)}{f_{\Pi}(n)}\right)$. В подразделе 4.2.3 мы проверим, что резолюционная система доказательств, система секущих плоскостей и система полиномиального исчисления устойчивы к замене переменных. Как следствие, мы получим нижние оценки для их расширений с правилом сдвига.

4.2.1 Кодировка

Мы определим подвижную формулу Ψ_n, которая зависит от бесконечной в обе стороны строки переменных $\cdots y_{-1}, y_0, y_1, \cdots$. Идея преобразования — представить строку в виде блоков по n переменных в каждом: $\cdots(x_1^{(-1)}x_2^{(-1)}\cdots x_n^{(-1)})(x_1^{(0)}x_2^{(0)}\cdots x_n^{(0)})(x_1^{(1)}x_2^{(1)}\cdots x_n^{(1)})\cdots$, и задать над каждым блоком набор дизъюнктов формулы ϕ_n. Проблема в том, что любой подвижный дизъюнкт содержит все возможные сдвиги, и задает ограничение не только для блоков вида $(x_1^{(t)}x_2^{(t)}\cdots x_n^{(t)})$, но и для блоков вида $x_i^{(t)}x_{i+1}^{(t)}\cdots x_n^{(t)})(x_1^{(t+1)}x_2^{(t+1)}\cdots x_{i-1}^{(t+1)}$ для всех $i \in \{2, 3, \cdots, n\}$.

Мы определим специальную кодировку, которая позволит обойти эту проблему. В новой кодировке мы используем три символа: $\widetilde{0}$, $\widetilde{1}$ и \$. Символы задаются четырьмя битами: $\$ = 0100$, $\widetilde{0} = 0110$, $\widetilde{1} = 0111$.

Ниже мы определим специальную формулу Enc_n. Выполняющий набор для $\mathrm{Shifts}(\mathrm{Enc}_n)$ будет бесконечной строкой, имеющей вид $(\$\{\widetilde{0}, \widetilde{1}\}^n)^\infty$. Символ \$ будем называть *сепаратором*, он будет разделять блоки из n цифр $\widetilde{0}$ и $\widetilde{1}$.

Затем мы определим формулу $\mathrm{Formula}_n$, которая переносит дизъюнкты ϕ на блоки. Каждый дизъюнкт C формулы $\mathrm{Shifts}(\mathrm{Formula}_n)$ будет начинаться с поддизъюнкта из четырех переменных $y_i \vee \neg y_{i+1} \vee y_{i+2} \vee y_{i+3}$. Если в подстановке ρ переменным $y_i y_{i+1} y_{i+2} y_{i+3}$ не соответствует код сепаратора, дизъюнкт C может быть опущен как выполненный. В противном случае, будем говорить, что дизъюнкт C выравнен по сепаратору относительно подстановки ρ.

По сути, любая подстановка ρ, выполняющая формулу $\mathrm{Shifts}(\mathrm{Enc}_n)$, прореживает дизъюнкты формулы $\mathrm{Shifts}(\mathrm{Formula}_n)$, оставляя только выровненные по сепаратору. Выровненные по сепаратору дизъюнкты в

свою очередь задают ограничения на блоки $\{\widetilde{0}, \widetilde{1}\}^n$ в подстановке ρ.

Формула Ψ_n — это конъюнкция формул $\mathrm{Shifts}(\mathrm{Enc}_n)$ и $\mathrm{Shifts}(\mathrm{Formula}_n)$.

Определим формулу Enc_n. Сначала определим набор подвижных дизъюнктов, чтобы любой выполняющий набор имел вид $\{\$, \widetilde{0}, \widetilde{1}\}^\infty$. Для этого мы зададим три условия из девяти подвижных дизъюнктов. Для удобства мы будем обозначать подвижные дизъюнкты в качестве запрещенных подстрок.

1. Любая подстрока длины 5 должна содержать контрольную последовательность 01. Ограничения: 11111, 11110, 11100, 11000, 10000, 00000.

2. Контрольная последовательность 01 повторяется через каждые два бита. Ограничения: 01□□1, 01□□00.

3. В строке не встречается подстрока 0101: символ, отличный от \$, $\widetilde{0}$ и $\widetilde{1}$. Ограничение: 0101.

Следующие два условия задают расстояние между парой соседних сепараторов.

4. Сепараторы не стоят ближе, чем на расстоянии n. Ограничения: $0100\square^{4k}0100$ для всех $k \in \{0, \cdots, n-1\}$.

5. На блок из $n+1$ закодированного символа приходится хотя бы один сепаратор. Ограничение: $(011\square)^{n+1}$.

Формула Enc_n является конъюнкцией описанных ограничений.

Определим формулу $\mathrm{Formula}_n$. Каждому дизъюнкту $C = x_{i_1}^{\sigma_1} \vee x_{i_2}^{\sigma_2} \vee \cdots \vee x_{i_k}^{\sigma_k}$ формулы ϕ_n сопоставим дизъюнкт

$$C_\Psi = \underbrace{y_1 \vee \neg y_2 \vee y_3 \vee y_4}_{D_\$\text{: нарушается только на 0100(\$)}} \vee \underbrace{y_{4i_1+4}^{\sigma_1} \vee y_{4i_2+4}^{\sigma_2} \vee \cdots \vee y_{4i_k+4}^{\sigma_k}}_{p(C)}.$$

Пусть C'_Ψ — сдвиг дизъюнкта C_Ψ. На всех подстановках ρ, где дизъюнкт C'_Ψ выравнен по сепаратору, дизъюнкт C'_Ψ равносилен дизъюнкту $p(C)$.

Отметим, что из любого выполняющего набора формулы Ψ_n можно извлечь выполняющий набор для формулы ϕ_n. Таким образом, если формула ϕ_n невыполнима, то и формула Ψ_n невыполнима.

4.2.2 Нижние оценки для систем, устойчивых к замене переменных

В этом разделе мы покажем, как перенести нижние оценки с классических систем доказательств на системы доказательств с правилом сдвига.

Мы будем работать с системами доказательств определенной формы, которые будем называть классическими. Классическая система доказательств П оперирует некоторым видом предикатов (дизъюнкты, неравенства, уравнения) и правилами вывода над ними. Будем называть такие предикаты заключениями. Доказательство формулы ϕ в системе П это последовательность заключений $P_1, P_2, \cdots, P_s$. Каждое заключение либо равносильно дизъюнкту формулы ϕ, либо аксиома, либо следует по правилам вывода из предыдущих заключений. Правила вывода гарантируют семантическое следствие и проверяемы за полиномиальное время. Последнее заключение доказательства — тождественно ложный предикат.

Для примера, система секущих плоскостей оперирует неравенствами с целочисленными коэффициентами. Каждому дизъюнкту сопоставляется эквивалентное неравенство. Для каждой переменной определяются аксиомы $x \geq 0$ и $-x \geq -1$. Также мы используем аксиому $0 \geq -1$. Система использует три правила вывода: сложение неравенств, умножение на константу и деление с округлением. Полиномиальная проверяемость и семантическое следствие для правил очевидны. Последнее неравенство, $0 \geq 1$, тождественно ложный предикат.

Отметим, что не любая система доказательств по определению

Кука–Рекхоу является классической, но резолюционная система доказательств, система секущих плоскостей и система полиномиального исчисления под определение подходят. Мы потребуем еще одно свойство, которому эти системы также удовлетворяют.

Рассмотрим отображение $\tau : \Gamma \to \Gamma \cup \{0, 1\}$. Будем называть τ заменой переменных, если любая переменная y_i, которую заменили на другую переменную или константу ($\tau(y_i) \neq y_i$), сама не участвует в качестве замены ($y_i \notin \tau(\Gamma)$).

Определим предикат $P(y_{i_1}, \cdots, y_{i_k})[\tau]$ после замены как $P(\tau(y_{i_1}), \cdots, \tau(y_{i_k}))$. Определим формулу ϕ после замены как $\phi[\tau] = \bigwedge_{C \in \phi} C[\tau]$.

Определение 4.1. Система Π является *f–устойчивой к замене переменных*, если для любой замены τ и любого заключения системы Π такого, что $P[\tau] \not\equiv 1$, соблюдаются три условия.

1. Предикат $P[\tau]$ является заключением в системе Π.

2. Если заключение P — аксиома в системе Π, то и $P[\tau]$ — аксиома в системе Π.

3. Пусть заключение P является результатом правила вывода из заключений $P_1, \cdots, P_k$, тогда заключение $P[\tau]$ имеет вывод размера не более чем f из нетривиальной части предикатов $P_1[\tau], \cdots, P_k[\tau]$.

   Отметим, что если $P[\tau] \not\equiv 1$, то хотя бы одно из заключений $P_1[\tau], \cdots, P_k[\tau]$ тоже нетривиально.

Также отметим, что если предикат P равносилен дизъюнкту C, то после любой замены τ, предикат $P[\tau]$ равносилен дизъюнкту $C[\tau]$.

Лемма 4.1. Если классическая система доказательств Π f–устойчива, то из доказательства π формулы ϕ размера s можно построить доказательство π' формулы $\phi[\tau]$ размера $f \cdot s$.

Доказательство. Пусть $\pi = P_1, P_2, \cdots, P_s$ — доказательство формулы ϕ в классической системе доказательств Π, и τ — замена переменных.

Рассмотрим последовательность предикатов $\pi[\tau] = P_1[\tau], \cdots, P_s[\tau]$. Опустим все тривиальные предикаты. Получим последовательность π'. По определению 4.1 каждый предикат последовательности π' — заключение системы Π. Пусть $P_i[\tau] \not\equiv 1$. Если заключение P_i равносильно некоторому дизъюнкту C формулы ϕ, то $P_i[\tau]$ равносильно $C[\tau]$. Если заключение P_i — аксиома системы Π, то и $P_i[\tau]$ — тоже аксиома системы Π. Если заключение P_i — результат правила вывода в системе Π, то заключение $P_i[\tau]$ имеет вывод размера не более чем f из предыдущих заключений в последовательности π'. После замены переменных, тождественно ложное заключение остается тождественно ложным.

Таким образом, по последовательности π' можно построить доказательство π'' для формулы $\phi[\tau]$ размера не более $f \cdot s$. □

Теорема 4.1. Пусть Π — классическая система доказательств, f–устойчивая к замене переменных. Пусть формула ϕ_n имеет кратчайшее доказательство в системе Π размера $S_\Pi(\phi)$. Тогда длина кратчайшего доказательства формулы Ψ_n в системе Shift-Π как минимум $\Omega\left(\frac{S_\Pi(\phi_n)}{f^2 \cdot n}\right)$.

План доказательства. Мы докажем теорему следующим образом. Сначала мы возьмем доказательство формулы Ψ_n и сделаем из него $m = 4 \cdot (n+1)$ параллельных копий со всем возможными сдвигами на j для $j \in [0, m-1]$. Затем, мы заменим каждую переменную y_i в формуле и выводе на $y_{i \bmod m}$. За счет того, что у нас есть m параллельных копий, нам не потребуется использовать правило сдвига: для каждого дизъюнкта мы выводим одновременно все возможные сдвиги по модулю m.

Далее, мы сделаем замену переменных, чтобы избавиться от сепаратора и вспомогательных переменных. Оставшиеся переменные будут соответствовать переменным формулы ϕ_n. Переименовав переменные, мы получим доказательство формулы ϕ_n.

В процессе преобразований размер доказательства увеличится не более чем в $O(f^2 \cdot n)$ раз.

Доказательство. Пусть $m = 4 \cdot (n+1)$ — длина закодированного блока $\$x_1 x_2 \cdots x_n$.

Пусть $\pi_0 = P_1, P_2, \cdots, P_l$ — доказательство формулы $\Psi_n = \text{Shifts}(\text{Enc}_n \wedge \text{Formula}_n)$ в системе Shift-П. Заменим каждое заключение $P(y_{i_1}, y_{i_2}, \cdots, y_{i_k})$ на m сдвигов $P(y_{i_1+j}, y_{i_2+j}, \cdots, y_{i_k+j})$ для $j \in [0, m-1]$. Новая последовательность π_1 длины $m \cdot S_\Pi(\phi)$ также будет доказательством формулы Ψ_n, состоящим из m параллельных копий π_0:

$$\begin{gathered} P_1^{(0)}, P_1^{(1)}, \cdots, P_1^{(m-1)}, \\ P_2^{(0)}, P_2^{(1)}, \cdots, P_2^{(m-1)}, \\ \vdots \\ P_l^{(0)}, P_l^{(1)}, \cdots, P_l^{(m-1)}, \end{gathered}$$

где $P_i^{(j)}$ сдвиг заключения P_i на j позиций.

Заменим каждую переменную y_i, используемую в доказательстве π_1, на переменную $y_{i \bmod m}$. Замену обозначим τ. Получим последовательность предикатов $\pi_1[\tau]$.

Предложение 4.1. Формула $\Psi_n[\tau]$ имеет доказательство в системе П размера $f \cdot |\pi_1[\tau]|$.

Доказательство. Мы построим доказательство π_2 в системе П на основе последовательности предикатов $\pi_1[\tau]$.

Будем рассматривать только нетривиальные предикаты в последовательности $\pi_1[\tau]$. Обозначим их $Q_1, Q_2, \cdots, Q_s$ согласно порядку в последовательности $\pi_1[\tau]$. Заметим, что любое заключение π_1 может быть выведено без правила сдвига. Поскольку система доказательств П f–устойчива, каждый предикат Q_j является заключением в системе П.

Покажем индукцией по t, что заключение Q_t имеет вывод в системе П размера не более чем f из заключений $Q_1, \cdots, Q_{t-1}$. Пусть заключению $Q_t = P_i^{(j)}[\tau]$ для заключения $P_i^{(j)} \in \pi_1$. Возможны четыре случая.

- Заключение $P_i^{(j)}$ соответствует дизъюнкту формулы Ψ, тогда заключение $Q_t = P_i^{(j)}[\tau]$ соответствует дизъюнкту формулы $\Psi[\tau]$.

- Заключение $P_i^{(j)}$ является аксиомой в системе Shift-П, тогда заключение $Q_t = P_i^{(j)}[\tau]$ является аксиомой в системе П.

- Заключение $P_i^{(j)}$ выводится через правило вывода системы П из заключений $P_1, P_2, \cdots, P_k$. По f–устойчивости системы П, заключение $P_i^{(j)}[\tau]$ имеет вывод размера f из нетривиальной части предикатов $P_1[\tau], P_2[\tau], \cdots, P_k[\tau]$.

- Заключение $P_i^{(j)}$ выводится в результате правила сдвига из заключения $P_{i'}^{(j')}$. По построению последовательности $\pi_1[\tau]$, строка для $P_{i'}$ является циклическим сдвигом строки для P_i. Значит, существует $j'' \in [0, m-1]$ такой, что $P_{i'}^{(j'')}[\tau] = P_i^{(j)}[\tau]$, и, по индукции, имеет вывод размера не более чем f.

Последний предикат последовательности $\pi_1[\tau]$ является тождественно ложным. Значит, по последовательности $\pi_1[\tau]$ можно построить доказательство π_2 формулы $\Psi[\tau]$ размера не более $f \cdot |\pi_1[\tau]|$. □

Проведем вторую замену τ', подставляющую переменным $y_0, \cdots, y_{m-1}$ строку $0100(011\square)^n$, где $\square$ означает, что соответствующая переменная не затрагивается. Заметим, что все дизъюнкты в формуле $\text{Shifts}(\text{Enc}_n)[\tau][\tau']$ окажутся выполненными. Каждый дизъюнкт C формулы $\text{Shifts}(\text{Formula}_n)[\tau][\tau']$ окажется либо выполненным, либо сведется к дизъюнкту $p(C)$, где C – дизъюнкт формулы ϕ. Таким образом, $\Psi_n[\tau][\tau'] = p(\phi_n) = \bigwedge_{C \in \phi_n} p(C)$.

Т.к. система П f–устойчива, по лемме 4.1 на основе доказательства π_2 для формулы $\Psi[\tau]$ мы построим доказательство π_3 для формулы $p(\phi_n)$ размера не более $f \cdot |\pi_2| \leq f^2 \cdot m \cdot S_{\text{П}}(\phi_n)$. Переименовав переменные, получим доказательство формулы ϕ_n.

Поскольку $m = O(n)$, то размер получившегося доказательства для формулы ϕ_n не больше $O(f^2 \cdot n \cdot S_{\text{П}}(\phi_n))$ и $S_{\text{Shift-П}}(\Psi_n) = \Omega\left(\frac{S_{\text{П}}(\phi_n)}{f^2 \cdot n}\right)$. □

4.2.3 Устойчивость к замене переменных

Лемма 4.2. Резолюционная система доказательств n–устойчива к замене переменных.

Доказательство. После замены переменных любой дизъюнкт остается дизъюнктом. Мы накладываем ограничение, что никакая переменная не участвует в дизъюнкте под разными знаками. Если после замены τ такая переменная появилась, дизъюнкт оказывается тождественно истинным и мы его опускаем. Значит, каждый нетривиальный дизъюнкт $P_i[\tau]$ является заключением резолюцинной системы доказательств.

Проверим, что если дизъюнкт P является результатом правила резолюции или правила ослабления, и дизъюнкт $P[\tau]$ нетривиален, то дизъюнкт $P[\tau]$ выводим за n шагов.

- Дизъюнкт P — результат правила ослабления. Пусть $P = C \vee x^\sigma$, и мы применяем правило
$$\frac{C}{C \vee x^\sigma}.$$
Дизъюнкт $P[\tau] = C[\tau] \vee x^\sigma[\tau] \not\equiv 1$, значит, $C[\tau] \not\equiv 1$ и $x^\sigma[\tau] \not\equiv 1$. Если $x^\sigma[\tau] = 0$, то заключение повторяет посылку. Иначе — является результатом ослабления.

- Дизъюнкт P — результат правила резолюции. Пусть $P = C_0 \vee C_1$, и мы применяем правило
$$\frac{x \vee C_0 \quad \neg x \vee C_1}{C_0 \vee C_1}.$$
Дизъюнкт $P[\tau] = C_0[\tau] \vee C_1[\tau] \not\equiv 1$, значит, $C_0[\tau] \not\equiv 1$ и $C_1[\tau] \not\equiv 1$. Если τ заменяет переменную x на константу $\sigma \in \{0, 1\}$, заключение $P[\tau]$ выводится не более чем через n ослаблений посылки $(x^\sigma \vee C_\sigma)[\tau] = C_\sigma[\tau]$.

  Если $(x \vee C_1)[\tau] \not\equiv 1$ и $(\neg x \vee C_2)[\tau] \not\equiv 1$, то заключение выводится резолюцией из упомянутых дизъюнктов по переменной $x[\tau]$.

Пусть, без ограничения общности, $(x \vee C_0)[\tau] \equiv 1$. Поскольку $x[\tau] \neq 1$ и $C_0[\tau] \not\equiv 1$, то $\neg x[\tau] \in C_0[\tau]$. Поскольку $P[\tau] \not\equiv 1$, то $(\neg x \vee C_1)[\tau] \not\equiv 1$. Значит, $P[\tau] = C_0[\tau] \vee C_1[\tau]$ выводится из дизъюнкта $\neg x[\tau] \vee C_1[\tau]$ (при $\neg x[\tau] \in C_0[\tau]$) не более чем через n ослаблений.

□

Лемма 4.3. Система секущих плоскостей $3 \cdot (n+1)$–устойчива к замене переменных.

Доказательство. После замены переменных любое неравенство с целочисленными коэффициентами остается неравенством с целочисленными коэффициентами. Значит, любой предикат после замены остается заключением системы секущих плоскостей.

Пусть неравенство P — аксиома, без ограничения общности, $x \geq 0$ для некоторой переменной x. Поскольку $P[\tau] \not\equiv 1$, то $x[\tau]$ — тоже переменная. Значит, неравенство $x[\tau] \geq 0$ — тоже аксиома.

Проверим, что если неравенство P — результат правила вывода, и $P[\tau] \not\equiv 1$, то неравенство $P[\tau]$ выводимо за $3 \cdot (n+1)$ шаг.

- Неравенство P — результат сложения двух других неравенств. Пусть $P \equiv [\sum_{i \in I}(a_i + b_i) \cdot x_i \geq \alpha + \beta]$ для некоторого конечного множества индексов I, и мы применяем правило

$$\frac{\sum_{i \in I} a_i \cdot x_i \geq \alpha \quad \sum_i b_{i \in I} \cdot x_i \geq \beta}{\sum_{i \in I}(a_i + b_i) \cdot x_i \geq \alpha + \beta}.$$

  Обозначим первое неравенство-посылку через P_a, второе — через P_b.

  Заключение $P[\tau] \not\equiv 1$. Значит, либо $P_a[\tau] \not\equiv 1$, либо $P_b[\tau] \not\equiv 1$, либо оба неравенства нетривиальны вместе. Если нетривиальны оба неравенства, то заключение $P[\tau]$ выводится через сумму неравенств $P_a[\tau]$ и $P_b[\tau]$.

  Пусть, без ограничения общности, $P_a[\tau] \equiv 1$. Обозначим I' индексы переменных, получившихся после замены, т.е. индексы переменных $\tau(\{x_i\}_{i \in I}) \backslash \{0, 1\}$.

Пусть $a'_j = \sum_{i \in I : \tau(x_i) = x_j} a_i$, $\alpha' = \alpha - \sum_{i \in I : \tau(x_i) = 1} a_i$. Очевидно, неравенства $P_a[\tau]$ и $\sum_{j \in I'} a'_j \cdot x_j \geq \alpha'$ равносильны. Пусть $A^- = \sum_{j \in I' : a'_j < 0} a'_j$. Поскольку $P_a[\tau] \equiv 1$, то $A^- \geq \alpha'$.

Для всех $a'_j < 0$ возьмем аксиому $-x_j \geq -1$ для $j \in I'$ и умножим на коэффициент $-a'_j$. Для всех $a'_j > 0$ возьмем аксиому $x_j \geq 0$ для $j \in I'$ и умножим на a'_j. В сумме полученные неравенства дадут $\sum_{i \in I} a_i^{(\tau)} \cdot x_i[\tau] \geq A^-$. Умножим неравенство $0 \geq -1$ на коэффициент $A^- - \alpha$. Все перечисленные неравенства в сумме дают $P_a[\tau]$, добавим их к неравенству $P_b[\tau]$ и получим $P_i[\tau]$.

Таким образом, $P[\tau]$ имеет вывод размера $3 \cdot (n + 1)$.

- Неравенство P — результат умножения. Пусть P обозначает неравенство $\sum_{i \in I} c \cdot a_i \cdot x_i \geq c \cdot \alpha$ для некоторого конечного множества индексов I, и мы применяем правило

$$\frac{\sum_{i \in I} a_i \cdot x_i \geq \alpha}{\sum_{i \in I} c \cdot a_i \cdot x_i \geq c \cdot \alpha}.$$

  Обозначим посылку через P'. Очевидно, $P' \equiv P_i$. Следовательно $P'[\tau] \equiv P[\tau]$. Поскольку $P[\tau] \not\equiv 1$, то $P'[\tau] \not\equiv 1$. Значит, $P[\tau]$ выводится из $P'[\tau]$ через правило умножения.

- Неравенство P — результат целочисленного деления. Случай разбирается аналогично умножению.

□

Лемма 4.4. Система полиномиального исчисления 1–устойчива к замене переменных.

Доказательство. После замены переменных любое полиномиальное уравнение остается полиномиальным уравнением. Значит, любой предикат после замены остается заключением системы полиномиального исчисления.

Мы будем считать, что два уравнений равны с точностью до произведения на скаляр поля K.

Пусть уравнение P — аксиома $x(1-x)=0$ для некоторой переменной x. Поскольку $P_i[\tau] \not\equiv 1$, то $x[\tau]$ — тоже переменная. Значит, уравнение $x[\tau](1-x[\tau])=0$ — тоже аксиома.

Проверим, что если уравнение P — результат правила вывода, и уравнение $P[\tau]$ нетривиально, то оно выводимо за один шаг.

- Уравнение P — линейная комбинация уравнений $R=0$ и $S=0$. Поскольку $P[\tau] \not\equiv 1$, то хотя бы одно уравнение нетривиально.

  Если оба уравнения нетривиальны, то $P[\tau]$ — линейная комбинация $(R=0)[\tau]$ и $(S=0)[\tau]$. Если, без ограничения общности, $(R=0)[\tau] \equiv 1$, то после приведения слагаемых мы получим, что $P[\tau]$ эквивалентно уравнению $(S=0)[\tau]$ с точностью до умножения на скаляр.

- Уравнение P — результат умножения уравнения $R=0$ на переменную x. Поскольку $P[\tau] \not\equiv 1$, то $x[\tau] \neq 0$ и $(R=0)[\tau] \not\equiv 1$. Если $x[\tau]=1$, то $P[\tau]$ эквивалентно $(R=0)[\tau]$. Иначе, уравнение $P[\tau]$ — результат умножения уравнения $(R=0)[\tau]$ на переменную $x[\tau]$.

□

Следствие 4.1. Существует семейство 3–КНФ формул $\{\phi_n\}_{n\geq 1}$, где каждая формула зависит от n переменных и содержит $O(n)$ дизъюнктов, что любое резолюционное доказательство со сдвигом соответствующей подвижной КНФ формулы Ψ_n имеет размер $2^{\Omega(n)}$.

Доказательство. Возьмем семейство формул, имеющих резолюционное доказательство $2^{\Omega(n)}$. Например, можно взять семейство, представленное в работе Уркухарта [16]. □

Следствие 4.2. Существует семейство КНФ формул $\{\phi_n\}_{n\geq 1}$, где каждая формула состоит из n дизъюкнтов, что любое доказательство в секущих плоскостях со сдвигом соответствующей подвижной КНФ формулы Ψ_n имеет размер $2^{n^{\Omega(1)}}$.

Доказательство. Возьмем семейство формул, имеющих доказательство в секущих плоскостях размера $2^{n^{\Omega(1)}}$. Например, можно взять семейство, представленное в работе Пудлака [42]. □

Следствие 4.3. Существует семейство КНФ формул $\{\phi_n\}_{n\geq 1}$, где каждая формула состоит из $O(n^2)$ дизъюкнтов, что любое доказательство в полиномиальном исчислении со сдвигом соответствующей подвижной КНФ формулы Ψ_n имеет размер $2^{n^{\Omega(1)}}$.

Доказательство. Возьмем семейство формул $\{\mathtt{PHP}_n^{n+1}\}_{n\geq 1}$. Из результатов Разборова [43] и Импальяццо и др. [44] следует, что каждая из формул $\mathtt{PHP}_n^{n+1}$ имеет размер $2^{n^{\Omega(1)}}$. □

4.3 Разделение систем доказательств с правилом сдвига и без него

В этом разделе мы определим семейство невыполнимых подвижных формул $\{\Phi_n\}_{n\geq 1}$. Формула Φ_n будет содержать полиномиальное от n число подвижных дизъюнктов. Размер любого классического доказательства без правила сдвига для формулы Φ_n составит $\Omega(2^n)$. При этом размер доказательства в системе Shift-Res будет всего лишь полиномиального от n размера.

4.3.1 Формула–счетчик

Мы воспользуемся кодировкой, описанной в разделе 4.2.1. Каждая формула Φ_n будет содержать подформулу $\mathrm{Shifts}(\mathrm{Enc}_n)$, выполнимую бесконечной строкой $\cdots w_{-1}\$w_0\$w_1\cdots$, где w_i — блок из n цифр $\widetilde{0}$ и $\widetilde{1}$, задающий n–битный счетчик.

В формуле Φ_n мы определим дополнительную подформулу–инкремент, гарантирующую, что каждый блок больше предыдущего ровно на единицу. Кроме того, мы запретим старшему разряду любого блока быть равным единице. Очевидно, последовательность из $2^{n-1}+1$ бло-

ков не сможет выполнить все ограничения. Таким образом, формула Φ_n будет невыполнима.

Нам потребуется вспомогательная формула. Для каждого $n \geq 1$ и каждого $k \in [0, n-1]$ определим формулу $\mathrm{Step}_k^n(x_0, x_1, \cdots, x_{n-1}, y_0, y_1, \cdots, y_{n-1})$, которая принимает бинарное представление чисел x и y. При этом x_0 и y_0 — это младшие биты. Формула $\mathrm{Step}_k^n(x_0, x_1, \cdots, x_{n-1}, y_0, y_1, \cdots, y_{n-1})$ будет выполнима тогда и только тогда, когда $y = x + 2^k$ по модулю 2^n.

Формула Step_k^n определяется конъюнкцией следующих условий.

$$
\begin{aligned}
x_i = y_i \quad & \forall i \in \{0, 1, \cdots, k-1\}; \\
x_k \neq y_k; \quad & \\
x_i = y_i \;\rightarrow\; x_{i+1} = y_{i+1} \quad & \forall i \in \{k+1, \cdots, n-2\}; \\
x_i \neq y_i \wedge x_i = 1 \;\rightarrow\; x_{i+1} \neq y_{i+1} \quad & \forall i \in \{k, \cdots, n-2\}; \\
x_i \neq y_i \wedge x_i = 0 \;\rightarrow\; x_{i+1} = y_{i+1} \quad & \forall i \in \{k, \cdots, n-2\}.
\end{aligned}
$$

Заметим, что каждое ограничение зависит не более, чем от четырех переменных, и формула Step_k^n имеет представление в КНФ размера $\Theta(n)$.

Введем дополнительное обозначение. Пусть есть формула $\phi = C_1 \wedge C_2 \wedge \cdots \wedge C_k$ и дизъюнкт D. Припишем дизъюнкт D к каждому дизъюнкту ϕ и обозначим полученную формулу как $OR(D, \phi) = (C_1 \vee D) \wedge (C_2 \vee D) \wedge \cdots \wedge (C_k \vee D)$.

Мы определим формулу Φ_n как подвижное расширение конъюнкции четырех подформул.

1. Формула Enc_n из раздела 4.2.1; задает кодировку.

2. Формула $\mathrm{OR}(D_\$, \mathrm{Step}_0^n(x_8, x_{12}, \cdots, x_{4n+4}; x_{4n+12}, \cdots, x_{8n+8}))$, где $D_\$ = x_1 \vee \neg x_2 \vee x_3 \vee x_4$ — дизъюнкт для выравнивания по сепаратору; задает инкремент между двумя соседними блоками.

3. Формула V_n; посимвольно задает следующий сепаратор относительно предыдущего. Формула выводима из Enc_n, но нам удобнее запи-

сать ее в явном виде. Определяется следующими дизъюнктами:

$$\begin{aligned} D_\$ &\vee \neg x_{4n+5}, \\ D_\$ &\vee x_{4n+6}, \\ D_\$ &\vee \neg x_{4n+7}, \\ D_\$ &\vee \neg x_{4n+8}. \end{aligned}$$

4. Формула Over_n; запрещает единицу в старшем разряде:

$$\mathrm{Over}_n = D_\$ \vee \neg x_{4n+4}.$$

Определим формулу Φ_n как $\mathrm{Shifts}(\mathrm{Enc}_n \wedge \mathrm{OR}(D_\$, \mathrm{Step}_n^0) \wedge V_n \wedge \mathrm{Over}_n)$.

4.3.2 Верхняя и нижняя оценки

Лемма 4.5. Пусть П — классическая система доказательств без правила сдвига. Минимальный вывод формулы Φ_n в системе П имеет размер $\Omega(2^n)$.

Доказательство. Пусть вывод использует t дизъюнктов формулы Φ_n и опровергает соответствующую подформулу ϕ_n. Разобьем бесконечную строку на блоки по $m = 4(n+1)$ переменной в каждом. Скажем, что дизъюнкт затрагивает отрезок переменных, задаваемый минимальным и максимальным индексами переменных в дизъюнкте. Дизъюнкт затрагивает блок, если хотя бы одна переменная блока лежит внутри этого отрезка. По построению, каждый дизъюнкт формулы Φ_n затрагивает не более трех блоков.

Разобьем все затрагиваемые блоки на последовательности без разрывов. Формулу ϕ_n можно разбить на независимые подформулы, каждая из которых соответствует такой последовательности. Пусть последовательность $s = b_0 b_1 \cdots b_{k-1}$ имеет длину $k \leq 2^{n-1}$, и ей соответствует формула ψ_s. Запишем в блоке b_i число i в двоичном виде. Очевидно, формула ψ_s будет выполнена.

Поскольку ϕ_n невыполнима, то существует последовательность длины хотя бы $2^{n-1}+1$. Значит, $t \geq \frac{1}{3}2^{n-1}$, и доказательство имеет размер $\Omega(2^n)$. □

Покажем верхнюю оценку для резолюционных доказательств с правилом сдвига. Нам потребуется следующая лемма.

Лемма 4.6. Для любого $n \geq 1$ и любого $k \in [0, n-2]$ из конъюнкции формул

$$\mathrm{Step}^n_k(x_0, x_1, \cdots, x_{n-1}; y_0, y_1, \cdots, y_{n-1})$$
$$\mathrm{Step}^n_k(y_0, y_1, \cdots, y_{n-1}; z_0, z_1, \cdots, z_{n-1})$$

можно вывести все дизъюнкты формулы

$$\mathrm{Step}^n_{k+1}(x_0, x_1, \cdots, x_{n-1}; z_0, z_1, \cdots, z_{n-1})$$

с выводом размера $O(n)$.

Доказательство. Пусть две формулы A и B, представленные в КНФ, зависят от константного числа переменных. Пусть формула A семантически следует из формула B, тогда каждый дизъюнкт формулы A семантически следует из формулы B. По лемме 1.1 каждый дизъюнкт формулы A имеет вывод в резолюционной системе доказательств. Поскольку формула B зависит от константного числа переменных, вывод имеет константный размер.

Таким образом, нам будет достаточно показать семантическое следствие каждого ограничения формулы Step^n_{k+1} из константного числа ограничений формул Step^n_k.

- $x_i = z_i \ \forall i \in [0, k]$. Для $i \in [0, k-1]$ формула $x_i = z_i$ следует из $x_i = y_i$ и $y_i = z_i$. Для $i = k$ формула $x_i = z_i$ следует из $x_i \neq y_i$ и $y_i \neq z_i$.

- $x_{k+1} \neq z_{k+1}$. Если $x_k = 1$, то $y_k = 0$ и $z_k = 1$. По следствиям

  $$\begin{aligned} x_k \neq y_k \wedge x_k = 1 \ &\rightarrow \ x_{k+1} \neq y_{k+1}; \\ y_k \neq z_k \wedge y_k = 0 \ &\rightarrow \ y_{k+1} = z_{k+1}. \end{aligned}$$

  получим $x_{k+1} \neq z_{k+1}$.

Аналогично, если $x_k = 0$, то $y_k = 1$ и $z_k = 0$. Используя следствия

$$\begin{aligned} x_k \neq y_k \wedge x_k = 0 &\rightarrow x_{k+1} = y_{k+1}; \\ y_k \neq z_k \wedge y_k = 1 &\rightarrow y_{k+1} \neq z_{k+1}, \end{aligned}$$

получим $x_{k+1} \neq z_{k+1}$.

- $x_i = z_i \rightarrow x_{i+1} = z_{i+1}$ $\forall i \in [k+2, n-2]$. Покажем, что если $x_i = z_i$ для $i \in [k+2, n-2]$, то $x_i = y_i = z_i$ и, значит, $x_{i+1} = y_{i+1} = z_{i+1}$.

  Пусть $x_i \neq y_i \neq z_i$. Следующие утверждения верны для $i \in [k+2, n-2]$.

  $$\begin{aligned} x_{i-1} \neq y_{i-1} \wedge x_{i-1} = 0 &\rightarrow x_i = y_i \\ y_{i-1} \neq z_{i-1} \wedge y_{i-1} = 0 &\rightarrow y_i = z_i \end{aligned}$$

  По контрпозиции получим, что $x_{i-1} = y_{i-1} \vee x_{i-1} = 1$ и $y_{i-1} = z_{i-1} \vee y_{i-1} = 1$. Поскольку $x_{i-1} = y_{i-1} \rightarrow x_i = y_i$ и $x_i \neq y_i$, получим, что $x_{i-1} \neq y_{i-1}$ и, значит, $x_{i-1} = 1$. Аналогично, получим, что $y_{i-1} = 1$. Тогда $x_{i-1} = y_{i-1}$ и $x_i = y_i$, противоречие.

- $x_i \neq z_i \wedge x_i = 1 \rightarrow x_{i+1} \neq z_{i+1}$ $\forall i \in [k+1, n-2]$.

  Пусть $1 = x_i = y_i \neq z_i$. Используем импликации $x_i = y_i \rightarrow x_{i+1} = y_{i+1}$ и $y_i \neq z_i \wedge y_i = 1 \rightarrow y_{i+1} \neq z_{i+1}$. Получим, что $x_{i+1} = y_{i+1} \neq z_{i+1}$.

  Пусть $1 = x_i \neq y_i = z_i$. Используем импликации $x_i \neq y_i \wedge x_i = 1 \rightarrow x_{i+1} \neq y_{i+1}$ и $y_i = z_i \rightarrow y_{i+1} = z_{i+1}$. Получим, что $x_{i+1} \neq y_{i+1} = z_{i+1}$.

- $x_i \neq z_i \wedge x_i = 0 \rightarrow x_{i+1} = z_{i+1}$ $\forall i \in [k+1, n-2]$.

  Пусть $0 = x_i = y_i \neq z_i = 1$. Используем импликации $x_i = y_i \rightarrow x_{i+1} = y_{i+1}$ и $y_i \neq z_i \wedge y_i = 0 \rightarrow y_{i+1} = z_{i+1}$. Получим, что $x_{i+1} = y_{i+1} = z_{i+1}$.

  Пусть $0 = x_i \neq y_i = z_i = 1$. Используем импликации $x_i \neq y_i \wedge x_i = 0 \rightarrow x_{i+1} = y_{i+1}$ и $y_i = z_i \rightarrow y_{i+1} = z_{i+1}$. Получим, что $x_{i+1} = y_{i+1} = z_{i+1}$.

Заметим, что всего ограничений $O(n)$, и потому их резолюционный вывод также будет иметь размер $O(n)$. $\square$

Лемма 4.7. Для формулы Φ_n существует резолюционное доказательство с правилом сдвига полиномиального от n размера.

Доказательство. Мы покажем, что если блоки находятся на расстоянии 2^k, то записанные в них числа различаются на 2^k. Определим вспомогательную формулу V_n^k, равную конъюнкции следующих дизъюнктов:

$$\begin{aligned} D_\$ &\vee \neg x_{4\cdot(n+1)\cdot 2^k+1},\\ D_\$ &\vee x_{4\cdot(n+1)\cdot 2^k+2},\\ D_\$ &\vee \neg x_{4\cdot(n+1)\cdot 2^k+3},\\ D_\$ &\vee \neg x_{4\cdot(n+1)\cdot 2^k+4}. \end{aligned}$$

Доказательство проведем по индукции по k. Будем выводить $\mathrm{OR}(D_\$, \mathrm{Step}_n^k)$ и V_n^k для всех $k \in [0, n-1]$.

При $k = 0$ мы используем формулу $\mathrm{OR}(D_\$, \mathrm{Step}_n^0)$ и $V_n^0 = V_n$. Пусть мы вывели $\mathrm{OR}(D_\$, \mathrm{Step}_n^{k-1})$ и V_n^{k-1}. Возьмем три блока переменных y, z и w размера $4\cdot(n+1)$, находящиеся последовательно на расстоянии 2^{k-1} таких же блоков. Определим сдвиг формулы $\mathrm{OR}(D_\$, \phi)$ на блок q для $q \in \{y, z, w\}$, как сдвиг, где переменным дизъюнкта $D_\$$ соответствуют первые четыре переменных блока q. Будем говорить в таком случае, что дизъюнкт $D_\$$ находится на блоке q.

Сдвинем $\mathrm{OR}(D_\$, \mathrm{Step}_n^{k-1})$ сначала на блок y, потом на блок z и воспользуемся выводом из леммы 4.6.

Пусть $D_\$^y$ — дизъюнкт, находящийся на блоке y; $D_\$^z$ — дизъюнкт на блоке z. В результате вывода по лемме 4.6 мы получим формулу $\mathrm{OR}(D_\$^y \vee D_\$^z, \mathrm{Step}_n^k)$. Сдвинем формулу V_n^{k-1} на блок y и последовательно удалим резолюцией все литералы $D_\$^z$ из каждого дизъюнкта формулы $\mathrm{OR}(D_\$^y \vee D_\$^z, \mathrm{Step}_n^k)$.

Для вывода V_n^k достаточно взять формулу V_n^{k-1} и ее сдвиг на 2^{k-1} блоков и провести аналогичное удаление.

Пусть мы вывели формулу Step_n^{n-1}. Значит, для блоков y и z находящихся на расстоянии 2^{n-1}, у нас есть ограничение $\mathrm{OR}(D_\$^y, y_{n-1} \neq z_{n-1})$. Воспользуемся подформулой Over_n. Сдвигом получим дизъюнкты $D_\$^y \vee \neg y_{n-1}$ и $D_\$^z \vee \neg z_{n-1}$. Используя дизъюнкты V_n^{n-1}, выведем $D_\$^y \vee \neg z_{n-1}$.

Заметим, что из $y_{n-1} \neq z_{n-1}$, $\neg y_{n-1}$ и $\neg z_{n-1}$ следует противоречие. Значит, из выведенных формул существует резолюционный вывод дизъюнкта $D^y_{\$}$ константного размера.

Лемма 4.8. Подвижная формула $\mathrm{Shift}(D_{\$} \wedge \mathrm{Enc}_n)$ имеет вывод противоречия в системе Shift-Res полиномиального от n размера.

Доказательство. Дизъюнкт $D_{\$}$ можно рассмотреть как запрет подстроки 0100.

Для наглядности изложения, мы определим резолюцию на строках, которая будет соответствовать резолюции дизъюнктов в системе Shift-Res. Пусть есть две строки s_1 и s_2 равной длины l такие, что

- ровно в одной позиции i_0 выполняется $s_1[i_0] = 0$ и $s_2[i_0] = 1$;
- для всех остальных $i \neq i_0$ либо $s_1[i] = \square$, либо $s_2[i] = \square$, либо $s_1[i] = s_2[i]$.

Резолюция строк s_1 и s_2 — это строка r длины l такая, что

- $r[i_0] = \square$;
- для $i \neq i_0$ если $s_1[i] = \square$, то $r[i] = s_2[i]$;
- для $i \neq i_0$ если $s_2[i] = \square$, то $r[i] = s_1[i]$;
- для $i \neq i_0$ если $s_1[i] = s_2[i]$, то $r[i] = s_1[i]$.

Мы также разрешим дополнять строки слева и справа символами $\square$, чтобы делать сдвиги и проводить резолюцию для строк разной длины. После применения правила, мы можем убрать все символы $\square$, которые находятся на левом и правом концах строки.

Например, для строк 01$\square$01 и 000 можно провести резолюцию

$$\begin{array}{l} \texttt{01}\square\texttt{01} \\ \square\square\texttt{000} \\ \hline \texttt{0100}\square \end{array}$$

и получить запрет 0100.

Мы покажем, как, используя ограничения для формулы Enc_n, определенные в разделе 4.2.1, и запрет 0100 вывести пустую строку, т.е. противоречие.

1. Резолюция строк для $D_\$ = 0100$ и 0101 (ограничение 3) дает запрещенную подстроку 010.

2. $n+1$ резолюция строки 010 со строкой $(011\square)^{n+1}$ (ограничение 5) дает строку $(01\square\square)^{n+1}$.

3. Используя запрещенные подстроки $01\square\square1$ и $01\square\square00$ (ограничение 2), будем убирать по символу с конца строки $(01\square\square)^{n+1}$:

$$\frac{\cdots 01\square\square01 \;\; 01\square\square00}{\cdots 01\square\square0}, \frac{\cdots 01\square\square0 \;\; 01\square\square1}{\cdots 01}.$$

   Как результат, получим запрещенную подстроку 01.

4. Используя четыре резолюции между 01 и 11111 (ограничение 1), выведем запрещенную строку 1.

5. Используя пять резолюций между строкой 00000 (ограничение 1) и 1, выведем противоречие.

□

□

Теорема 4.2. Существует семейство подвижных КНФ формул $\{\Phi_n\}$ таких, что

- для любой классической системы доказательств П, любое доказательство формулы Φ_n имеет размер $\Omega(2^n)$;
- формула Φ_n имеет резолюционное доказательство с правилом сдвига полиномиального от n размера.

Доказательство. Следует из лемм 4.5, 4.7. □

Заключение

В главе 2 показаны верхние оценки для систем Res и Res-Lin. Получены следующие результаты.

1. Доказано, что формула $\mathtt{PMP}_G$ для любого графа G на n вершинах без совершенного паросочетания имеет доказательство в системе Res размера $O(n^2 \cdot 2^n)$. Оценка совпадает с ранее известной нижней с точностью до константы в показателе экспоненты.

2. Доказано, что формула $\mathtt{PHP}_n^m$ при $m > n$ имеет древовидное доказательство размера $2^{O(n)}$ в системе Res-Lin. Оценка совпадает с нижней с точностью до константы в показателе экспоненты.

Как следствие результата 2, доказана верхняя оценка $2^{O(n)}$ на размер доказательства в системе Res-Lin для формулы $\mathtt{PMP}_G$ для любого графа G на n вершинах и без совершенного паросочетания.

Открытый вопрос 1. Существует ли полиномиальная верхняя оценка на размер вывода в системе Res-Lin для формулы $\mathtt{PMP}_G$?

Для системы Res-Lin известны нетривиальные нижние оценки только на размер древовидного доказательства.

Открытый вопрос 2. Доказать суперполиномиальные нижние оценки для доказательств общего вида в системе Res-Lin.

В главе 3 исследованы вопросы, связанные с CSP и системой доказательств NG-Res. Получены следующие результаты.

3. Доказано, что минимальная ширина доказательства чувствительных к подстановкам CSP ϕ ограничена снизу $e_2(\phi) - 1$. Нижняя оценка не зависит от размера алфавита.

4. Доказано, что невыполнимые обобщенные цейтинские формулы $\texttt{Ts}(G, f)$, построенные на основе графа G с максимальной степенью d и расширительной способностью $e_2(G)$ над алфавитом размера k, имеют древовидные доказательства в системе NG-Res размера как минимум $k^{e_2(G)-d}$.

5. Показана формально необходимость расширительной способности графа для нижних оценок на древовидные доказательства в системе NG-Res. По CSP ϕ над алфавитом размера k можно построить граф зависимостей $G = \langle V, E\rangle$, который описывает сколько общих переменных имеет каждая пара ограничений. Показано, что в графе G существует подграф H такой, что древовидная сложность CSP ϕ в системе NG-Res не больше $k^{e(H)\cdot\log_{3\setminus 2}|V|}$.

В результатах 4 и 5 между оценками на размер древовидного доказательств есть зазор: $k^{e(G)-d}$ против $k^{e(H)\cdot\log_{\frac{3}{2}}|V|}$.

Открытый вопрос 3. Установить асимптотически точную оценку на древовидную резолюционную сложность обобщенных цейтинских формул.

В главе 4 показаны оценки на размеры вывода в системах доказательств с правилом сдвига. Получены следующие результаты.

6. Построен пример невыполнимой подвижной формулы, которая имеет вывод полиномиального размера в резолюционной системе доказательств со сдвигом, однако, в любой классической системе доказательств без сдвига вывод имеет экспоненциальный размер.

7. Доказаны нижние экспоненциальные оценки для систем доказательств с правилом сдвига (резолюционных, секущих плоскостей и полиномиального исчисления).

Правило сдвига сильно меняет поведение системы доказательств. Существуют системы доказательств, для которых на сегодняшний день

неизвестно нижних оценок в классическом случае, например, система Ловаса-Схрейвера [45].

Открытый вопрос 4. Существуют ли нижние оценки для системы Ловаса-Схрейвера с правилом сдвига?

Литература

[1] Cook S., Reckhow R. The Relative Efficiency of Propositional Proof Systems // Journal of Symbolic Logic. 1979. 03. Vol. 44, no. 1. P. 36–50.

[2] Blake A. Canonical expressions in Boolean algebra. University of Chicago, 1938.

[3] Davis M., Putnam H. A Computing Procedure for Quantification Theory // J. ACM. 1960. Vol. 7, no. 3. P. 201–215.

[4] Robinson J. A. A Machine-Oriented Logic Based on the Resolution Principle // J. ACM. New York, NY, USA, 1965. jan. Vol. 12, no. 1. P. 23–41.

[5] Davis M., Logemann G., Loveland D. A Machine Program for Theorem-Proving // Commun. ACM. 1962. Vol. 5, no. 7. P. 394–397.

[6] Marques-Silva J. P., Sakallah K. A. GRASP: A search algorithm for propositional satisfiability // IEEE Transactions on Computers. 1999. Vol. 48, no. 5. P. 506–521.

[7] Silva J. P. M., Sakallah K. A. GRASP—a new search algorithm for satisfiability // Proceedings of the 1996 IEEE/ACM international conference on Computer-aided design / IEEE Computer Society. 1997. P. 220–227.

[8] Bayardo Jr R. J., Schrag R. Using CSP look-back techniques to solve real-world SAT instances // AAAI/IAAI. 1997. P. 203–208.

[9] Van Beek P. Backtracking search algorithms // Foundations of Artificial Intelligence. 2006. Vol. 2. P. 85–134.

[10] Beame P., Kautz H., Sabharwal A. Towards understanding and harnessing the potential of clause learning // Journal of Artificial Intelligence Research. 2004. Vol. 22. P. 319–351.

[11] Clause Learning Can Effectively P-Simulate General Propositional Resolution. / P. Hertel, F. Bacchus, T. Pitassi et al. // AAAI. 2008. P. 283–290.

[12] Buss S. R., Hoffmann J., Johannsen J. Resolution trees with lemmas: Resolution refinements that characterize DLL algorithms with clause learning // arXiv preprint arXiv:0811.1075. 2008.

[13] Cook S. A. The complexity of theorem-proving procedures // Proceedings of the third annual ACM symposium on Theory of computing / ACM. 1971. P. 151–158.

[14] Левин Л А. Универсальные задачи перебора // Проблемы передачи информации. 1973. Т. 9, № 3. С. 115–116.

[15] Цейтин Г. С. О сложности вывода в исчислении высказываний // Записки научных семинаров ЛОМИ. 1968. Т. 8. С. 234–259.

[16] Urquhart A. Hard Examples for Resolution // JACM. 1987. Vol. 34, no. 1. P. 209–219.

[17] Haken A. The intractability of resolution // Theoretical Computer Science. 1985. Vol. 39, no. 0. P. 297 – 308. Third Conference on Foundations of Software Technology and Theoretical Computer Science.

[18] Buss S. R., Turán G. Resolution proofs of generalized pigeonhole principles // Theoretical Computer Science. 1988. Vol. 62, no. 3. P. 311–317.

[19] Buss S., Pitassi T. Resolution and the weak pigeonhole principle // International Workshop on Computer Science Logic / Springer. 1997. P. 149–156.

[20] Razborov A. A. Resolution lower bounds for perfect matching principles // J. Comput. Syst. Sci. 2004. Vol. 69, no. 1. P. 3–27. URL: http://dx.doi.org/10.1016/j.jcss.2004.01.004.

[21] Dantchev Stefan, Riis Søren. "Planar"tautologies hard for Resolution // Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on / IEEE. 2001. C. 220–229.

[22] Alekhnovich M. Mutilated chessboard problem is exponentially hard for resolution // Theoretical Computer Science. 2004. Vol. 310, no. 1. P. 513–525.

[23] Itsykson D., Slabodkin M., Sokolov D. Resolution Complexity of Perfect Matching Principles for Sparse Graphs // Computer Science - Theory and Applications - 10th International Computer Science Symposium in Russia, CSR 2015, Listvyanka, Russia, July 13-17, 2015, Proceedings. 2015. P. 219–230. URL: http://dx.doi.org/10.1007/978-3-319-20297-615.

[24] Seto K., Tamaki S. A satisfiability algorithm and average-case hardness for formulas over the full binary basis // Computational Complexity. 2013. Vol. 22, no. 2. P. 245–274.

[25] Itsykson D. M., Sokolov D. O. Lower Bounds for Splittings by Linear Combinations // Mathematical Foundations of Computer Science 2014 / Ed. by E. Csuhaj-Varjú, M. Dietzfelbinger, Z. Ésik. Springer Berlin Heidelberg, 2014. Vol. 8635 of *Lecture Notes in Computer Science*. P. 372–383.

[26] Iwama K., Miyazaki S. Tree-like resolution is superpolynomially slower than dag-like resolution for the pigeonhole principle // International

Symposium on Algorithms and Computation / Springer Berlin Heidelberg. 1999. P. 133–142.

[27] Mitchell D. G. The Resolution Complexity of Constraint Satisfaction. 2002.

[28] Ben-Sasson E., Wigderson A. Short proofs are narrow — resolution made simple // Journal of ACM. 2001. Vol. 48, no. 2. P. 149–169.

[29] Linear gaps between degrees for the polynomial calculus modulo distinct primes / S. Buss, D. Grigoriev, R. Impagliazzo et al. // Proceedings of the thirty-first annual ACM symposium on Theory of computing / ACM. 1999. P. 547–556.

[30] Aho A. V., Corasick M. J. Efficient string matching: an aid to bibliographic search // Communications of the ACM. 1975. Vol. 18, no. 6. P. 333–340.

[31] Lothaire M. Algebraic combinatorics on words. No. 90. Cambridge University Press, 2002.

[32] On the complexity of deciding avoidability of sets of partial words / B. Blakeley, F. Blanchet-Sadri, J. Gunter et al. // International Conference on Developments in Language Theory / Springer. 2009. P. 113–124.

[33] Blanchet-Sadri F., Jungers R. M., Palumbo J. Testing avoidability on sets of partial words is hard // Theoretical Computer Science. 2009. Vol. 410, no. 8. P. 968–972.

[34] Itsykson D., Okhotin A., Oparin V. Computational and Proof Complexity of Partial String Avoidability // 41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016, August 22-26, 2016 - Kraków, Poland. Vol. 58 of *Leibniz International Proceedings in Informatics*. 2016. P. 51:1–51:13. URL: http://dx.doi.org/10.4230/LIPIcs.MFCS.2016.51.

[35] Itsykson D., Oparin V. Graph Expansion, Tseitin Formulas and Resolution Proofs for CSP // Computer Science - Theory and Applications - 8th International Computer Science Symposium in Russia, CSR 2013, Ekaterinburg, Russia, June 25-29, 2013. Proceedings. Vol. 7913 of *Lecture Notes in Computer Science*. 2013. P. 162–173.

[36] Oparin V. Tight Upper Bound on Splitting by Linear Combinations for Pigeonhole Principle // Theory and Applications of Satisfiability Testing – SAT 2016: 19th International Conference, Bordeaux, France, July 5-8, 2016, Proceedings / Ed. by N. Creignou, D. Le Berre. Cham: Springer International Publishing, 2016. Vol. 9710 of *Lecture Notes in Computer Science*. P. 77–84.

[37] Tight Lower Bounds on the Resolution Complexity of Perfect Matching Principles / D. Itsykson, V. Oparin, M. Slabodkin et al. // Fundamenta Informaticae. 2016. Vol. 145, no. 3. P. 229–242.

[38] Krajíček J. Proof Complexity // European Congress of Mathematics (ECM). 2005. P. 221–231.

[39] Jukna S. Boolean function complexity: advances and frontiers. Springer Science & Business Media, 2012. Vol. 27.

[40] Tutte W. The factors of graphs // Canad. J. Math. 1952. Vol. 4, no. 3. P. 314–328.

[41] Buss S. R., Pitassi T. Resolution and the Weak Pigeonhole Principle // Computer Science Logic, 11th International Workshop, CSL '97, Annual Conference of the EACSL, Aarhus, Denmark, August 23-29, 1997, Selected Papers. 1997. P. 149–156. URL: http://dx.doi.org/10.1007/BFb0028012.

[42] Pudlák P. Lower Bounds for Resolution and Cutting Plane Proofs and Monotone Computations. 1997.

[43] Razborov A. Lower bounds for the polynomial calculus // computational complexity. 1998. Vol. 7, no. 4. P. 291–324.

[44] Impagliazzo R., Pudlák P., Sgall J. Lower bounds for the polynomial calculus and the Gröbner basis algorithm // computational complexity. 1999. Vol. 8, no. 2. P. 127–144.

[45] Beame P., Pitassi T., Segerlind N. Lower bounds for Lovász-Schrijver systems and beyond follow from multiparty communication complexity // SIAM Journal on Computing. 2007. Vol. 37, no. 3. P. 845–869.