ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
УЧРЕЖДЕНИЕ
САНКТ-ПЕТЕРБУРГСКОЕ ОТДЕЛЕНИЕ
МАТЕМАТИЧЕСКОГО ИНСТИТУТА ИМ. В. А. СТЕКЛОВА РАН

На правах рукописи

Соколов Дмитрий Олегович

Сложность решения задачи выполнимости булевых формул алгоритмами, основанными на расщеплении

01.01.06 — математическая логика, алгебра и теория чисел
01.01.09 — дискретная математика и математическая кибернетика

Диссертация
на соискание ученой степени
кандидата физико-математических наук

Научные руководители:
доцент, д.ф.-м.н. Э. А. Гирш
к.ф.-м.н. Д. М. Ицыксон

Санкт-Петербург
2014

Оглавление

Введение **4**

Системы доказательств 5

DPLL-алгоритмы 8

Экспандеры 13

Односторонние функции 14

Структура диссертации 17

1 Основные понятия **19**

1.1 Системы доказательств 19

1.2 DPLL-алгоритмы 22

1.2.1 Связь с системами доказательств 23

1.2.2 “Пьяные” и “близорукие” алгоритмы 24

1.3 Функция Голдрейха 25

1.3.1 Формулы, построенные по функции Голдрейха . . . 26

1.4 Экспандеры 27

1.5 Распределения на входах 28

2 Нижние оценки на сложность обращения функции Голдрейха близорукими DPLL-алгоритмами **30**

2.1 Почти биективная функция Голдрейха 31

2.1.1 Линейная функция 31

2.1.2 Немного нелинейная функция Голдрейха 32

2.2 Нижняя оценка времени работы на невыполнимых формулах 33

2.3 Нижняя оценка времени работы на выполнимых формулах 35

2.3.1 k-замыкание 35

2.3.2 “Умный” близорукий алгоритм 38

3 Нижние оценки на близорукие DPLL-алгоритмы с эвристикой отсечения 44

3.1 DPLL-алгоритмы с эвристикой отсечения 45
3.2 Система близоруких копий 47
3.3 Граничный экспандер и линейное замыкание 50
3.4 Очищенное дерева поиска 51
3.4.1 Универсальное распределение 56
3.4.2 Нижняя оценка на выполнимых формулах 57
3.5 Конструкция экспандера 57

4 Нижние оценки для расщепления по линейным комбинациям 60

4.1 Деревья расщеплений по линейным формам 61
4.2 Верхние оценки 63
4.3 Нижняя оценка для 2-кратных цейтинских формул 65
4.3.1 Коммуникационный протокол из дерева линейных расщеплений 65
4.3.2 Нижняя оценка на коммуникационную сложность . 68
4.4 Нижняя оценка для принципа Дирихле 69
4.5 Системы доказательств Res-Lin и Sem-Lin 71
4.5.1 Древовидная Res-Lin и деревья линейных расщеплений 76
4.5.2 Импликативная полнота Res-Lin 78
4.5.3 Моделирование Res-Lin в $R(lin)$ 80

Литература 83

Введение

Задача выполнимости булевых (пропозициональных) формул (SAT) — это задача нахождения по булевой формуле такой подстановки значений переменным, что при применении данной подстановки формула обращается в тождественную истину. Если такая подстановка существует, то формула называется выполнимой; если же нет, то функция, задаваемая данной формулой, является тождественной ложью, и формула является невыполнимой.

SAT — одна из первых задач, для которых была доказана **NP**-полнота (теорема Кука-Левина [1, 2]). Это означает, что любая задача из класса **NP**, который включает в себя широкий круг естественных задач, возникающих на практике, сводится к задаче выполнимости булевых формул. Таким образом, существование эффективного алгоритма для SAT эквивалентно одной из центральных задач теории сложности о равенстве между классами **P** и **NP**, и таких алгоритмов в настоящее время не известно.

Несмотря на это, формулы, возникающие на практике, успешно решаются при помощи эвристических SAT-солверов (программ для решения задачи выполнимости). В данной работе мы рассмотрим общую схему работы большинства современных SAT-солверов — алгоритмы расщепления, идея которых восходит к работам [3, 4]. Также мы рассмотрим криптографический аспект, а именно, возможности алгоритмов расщепления по взлому функции Голдрейха [5] — кандидата на роль односторонней функции. Также мы рассмотрим системы доказательств, связанные с алгоритмами расщепления.

Классическими примерами формул, которые сложны для алгорит-

мов расщепления, являются цейтинские формулы [6, 7, 8] и их обобщения, представляющие собой линейные системы уравнений по модулю 2. Однако такие формулы легко решаются методом Гаусса. В данной диссертации рассматривается обобщение классических алгоритмов расщепления и доказывается ряд верхних оценок на такие формулы. Также строятся другие, нелинейные, трудные примеры и доказываются нижние оценки на время работы этих алгоритмов на таких примерах и нижние оценки для связанных с этими алгоритмами систем доказательств. Следует отметить, что трудные варианты функции Голдрейха, рассматриваемые в данной диссертации, также являются нелинейными.

Системы доказательств

Систематическое изучение вопросов, связанных с системами доказательств для пропозициональной логики, в частности, вопроса о длине минимального доказательства в различных системах, началось с работы [9]. Интерес к этим вопросам обусловлен, в частности, тем, что пропозициональная система доказательств — это недетерминированный алгоритм, который определяет, является ли булева формула тавтологией (или невыполнимой формулой). Из гипотезы о неравенстве классов $\mathbf{NP} \neq \mathbf{co\text{-}NP}$ следует существование трудных примеров для всех систем доказательств.

Дадим формальное определение.

Определение 1. ([9], [10]) Системой доказательств для языка L называется полиномиальный по времени алгоритм $\Pi : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$, для которого выполнены следующие свойства:

- (полнота) если $x \in L$, то существует такая строка y, что $P(x,y) = 1$;
- (корректность) если существует такая строка y, что $P(x,y) = 1$, то $x \in L$.

Соответствующую строку y мы будем называть доказательством того, что x принадлежит языку L. В дальнейшем мы будем рассматривать

только пропозициональные системы доказательств, т.е. системы доказательств для языка пропозициональных тавтологий (TAUT) или, что аналогично, для языка невыполнимых формул UNSAT.

Длина доказательства — это длина его битовой записи. Мы можем сравнивать системы доказательств "по силе", т.е. система Π_1 моделирует систему Π_2, если для любой формулы в системе Π_1 найдется доказательство не сильно длиннее (в полином раз), чем любое доказательство в системе Π_2. Вопрос существования "самой сильной" системы доказательств является открытым, в случае отрицательного ответа последовало бы, что $\mathbf{NP} \neq \mathbf{co\text{-}NP}$, и, в частности, $\mathbf{P} \neq \mathbf{NP}$.

Следующий план иногда называют программой Кука [9, 11].

Программа Кука по разделению классов NP и co-NP: доказывать суперполиномиальные нижние оценки для все более сильных систем доказательств, пока не удастся развить методы, позволяющие обобщить результаты на произвольную систему доказательств.

Резолюционная система доказательств. Резолюционная система доказательств является одной из самых простых. Невыполнимая формула представляется в виде множества дизъюнктов (клозов). Доказательство получается путем добавления новых клозов из уже имеющихся при помощи правил вывода, пока мы не сможем получить пустой (тождественно ложный) клоз. Резолюция содержит два правила вывода: из клоза C можно вывести любой клоз $(C \vee x)$, где x — произвольный литерал; правило резолюции: из клозов $(x \vee D)$ и $(\neg x \vee D')$ можно вывести клоз $D \vee D'$.

Первые суперполиномиальные оценки на резолюционную систему были получены Цейтиным в 1968 году [6]. В 1985 году Хакен построил примеры формул, на которых достигается экспоненциальная нижняя оценка [12]. Вскоре были построены и другие трудные для резолюций примеры [7, 13], основанные на примере Хакена. В 2001 году Бен-Сассон и Вигдерсон [8] предложили один из наиболее общих методов доказательства нижних оценок на резолюционную систему доказательства —

метод ширины.

В главе 4 мы рассмотрим обобщение резолюционной системы доказательств, которое работает с дизъюнкциями линейных уравнений над полем $\mathbb{F}_2$. У системы Res-Lin есть два правила вывода: ослабление и резолюция. Также мы рассмотрим семантический аналог данной системы — Sem-Lin, у которой второе правило вывода заменено на семантическое, то есть из двух дизъюнкций линейных уравнений можно вывести любую, которая следует из них. Мы докажем, что эти системы эквиваленты, т.е. моделируют друг друга (более того, доказательство в одной из них можно за полиномиальное время перестроить в доказательство в другой системе). Также мы докажем свойство "импликационной полноты" данных систем: если формула $\phi \to \psi$ является тавтологией, то в данных системах из формулы ϕ можно получить формулу ψ. Мы докажем ряд экспоненциальных нижних оценок на длину доказательств в древовидном случае.

Система доказательств $R(lin)$. Раз и Цамерет в работе [14] предложили систему доказательств $R(lin)$. Эта система оперирует линейными уравнениями с целыми коэффициентами и пропозициональными переменными. Целочисленный линейный дизъюнкт — это дизъюнкция $\bigvee_i (\sum_j a_{i,j} x_j = b_i)$, где $a_{i,j}$ и b_j целые. Уравнения в дизъюнкте не повторяются.

Система доказательств $R(lin)$ содержит аксиомы $(x = 0) \vee (x = 1)$ для всех переменных x и следующие правила вывода:

- правило сечения позволяет получить дизъюнкты $A \vee B \vee (F_1 + F_2 = a_1 + a_2)$ и $A \vee B \vee (F_1 - F_2 = a_1 - a_2)$ из $A \vee (F_1 = a_1)$ и $B \vee (F_2 = a_2)$;
- синтаксическое ослабление позволяет получить дизъюнкт $A \vee (F = a)$ из A для любого целочисленного линейного уравнения $F = a$;
- правило упрощения позволяет получить дизъюнкт B из $B \vee (0 = c)$, где c — ненулевая константа.

В диссертации мы покажем, что система $R(lin)$ p-моделирует системы Res-Lin и Sem-Lin, однако на систему $R(lin)$ не известно нижних оценок, поэтому данный факт не позволяет получить примеры трудных формул для систем Res-Lin и Sem-Lin.

DPLL-алгоритмы

Одним из основных подходов к решению задачи выполнимости пропозициональных формул являются DPLL-алгоритмы (названы в честь авторов: Davis, Putnam, Logemann и Loveland). DPLL-алгоритм — рекурсивный алгоритм, который на вход получает формулу ϕ. Сначала данный алгоритм запускает процедуру **A**, которая выбирает переменную x. После этого алгоритм запускает процедуру **B** для выбора константы c, затем рекурсивно вызывает себя на формуле $\Phi[x := c]$. Если был найден выполняющий набор, то алгоритм выдает его, иначе возвращает результат запуска алгоритма на формуле $\Phi[x := 1 - c]$. Рекурсивные вызовы прекращаются, когда формула становится тривиальной.

Оценки на время работы DPLL-алгоритмов

Известны экспоненциальные нижние оценки на время работы DPLL-алгоритмов на невыполнимых формулах, в частности, такие оценки следуют из оценок на размер резолюционных доказательств [7], [6], [15]. В случае выполнимых формул суперполиномиальные нижние оценки на время работы всех возможных DPLL-алгоритмов повлекли бы за собой неравенство $\mathbf{P} \neq \mathbf{NP}$, так как при отсутствии ограничений процедура **B** может выбирать корректное значение для переменной. Экспериментальные данные показывают [16], что современные `SAT`-солверы могут выдавать корректный результат за приемлемое время на значительно больших выполнимых формулах, чем на невыполнимых. Несмотря на важность задачи существует не так много работ, в которых доказываются нижние оценки на время работы DPLL-алгоритмов на выполнимых формулах. В следующих работах [17, 18] рассматриваются специ-

фические эвристики локального поиска, также некоторые оценки были доказаны в работах [19, 20, 21].

В работе [22] дается экспоненциальная нижняя оценка для двух достаточно больших классов DPLL-алгоритмов — "близоруких" и "пьяных". В случае близоруких алгоритмов процедуры **A** и **B** могут видеть формулу со стертыми знаками отрицания, они могут запросить знаки отрицания на каждом шаге в $K = n^{1-\epsilon}$ дизъюнктах. В случае пьяных алгоритмов процедура **A** может быть произвольной, а процедура **B** выбирает значение случайным образом. В работах [23, 24] представлен "криптографический" взгляд на рассматриваемую проблему.

Все описанные нижние оценки на выполнимых формулах основаны на факте, что после нескольких шагов алгоритма формула станет трудной невыполнимой, и алгоритм обойдет все дерево расщепления для данной невыполнимой формулы. В главе 3 мы расширим класс DPLL-алгоритмов, добавив эвристику отсечения, которая может решить, что ветвь дерева расщепления "бесперспективная", и не стоит ее просматривать. Точнее, перед каждым рекурсивным вызовом алгоритм вызывает эвристику **C**, в зависимости от результата которой вызов выполняется или нет. DPLL-алгоритмы с эвристикой отсечения всегда выдают корректный результат на невыполнимых формулах, однако могут ошибаться на выполнимых. С другой стороны, если использование данной эвристики сильно сокращает время работы алгоритма, но при этом контрпримеры найти трудно, то данные алгоритмы могут быть полезны на практике.

В главе 3 мы покажем, что возможно построить за полиномиальное время семейство таких невыполнимых формул $\Phi^{(n)}$, что для любых детерминированных эвристик **A** и **C** найдется такой полиномиально моделируемый ансамбль распределений R_n, что DPLL-алгоритм, основанный на эвристиках **A**, **B** и **C** для некоторой эвристики **B**, либо ошибается на 99% формул, сгенерированных согласно распределению R_n, либо работает экспоненциальное время на формуле $\Phi^{(n)}$. В случае, если **A** и **C** не ограничены, мы покажем, что подобное утверждение эквивалентно

неравенству $\mathbf{P} \neq \mathbf{NP}$. В случае рандомизированых эвристик $\mathbf{A}$ и $\mathbf{C}$ вопрос остается открытым.

Теорема 1. Существуют такой полиномиальный алгоритм, который выдает по n невыполнимую формулу $\Phi^{(n)}$, и такая константа $\delta > 0$, что для любого близорукого алгоритма с полиномиальными эвристиками $\mathbf{A}$ и $\mathbf{C}$ найдется такой полиномиально моделируемый ансамбль распределений R_n на выполнимых формулах, что если для некоторой эвристики $\mathbf{B}$ и некоторого $\epsilon > 0$ неравенство $\Pr\limits_{\phi \leftarrow R_n}[\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\phi) = 1] \geq 1 - \epsilon$ выполнено, то время работы алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\Phi^{(n)})$ не менее $(1-\epsilon)2^N$, где $N = min\{n^\delta, r/K\}$ и $r = \Omega(n)$.

Для доказательства данного результата мы будем использовать технику замыканий для экспандеров, предложенную Алехновичем. Мы представим конструктивный вариант замыкания и применим его для построения ансамбля распределений R_n.

Мы также дадим похожие оценки для выполнимых формул, однако в этом случае невозможно доказать в точности такой же результат, так как близорукий алгоритм может подставлять значения переменным, соответствующие выполняющему набору конкретной формулы Φ. Тем не менее, в выполнимом случае возможно построить семейство формул $\Phi^{(n)}$, зависящее от эвристик $\mathbf{A}, \mathbf{B}$, в предположениях, что $\mathbf{B}$ близорука.

В работе [25] Ицыксоном был построе универсальный ансамбль распределений Q_n, который доминирует все распределения R_n. В частности, из данной конструкции следует, что близорукий DPLL-алгоритм с эвристикой отсечения, ошибающийся с вероятностью $o(1)$ на случайной формуле из ансамбля Q_n, работает экспоненциальное время на формуле $\Phi^{(n)}$.

Обобщение на расщепление по линейным комбинациям

Естественным обобщением DPLL-алгоритмов являются алгоритмы, в которых расщепление происходит по значению некоторой формулы. В

главе 4 мы рассмотрим обобщение, где алгоритм может проводить расщепление по линейной комбинации переменных над полем $\mathbb{F}_2$. Существует полиномиальный алгоритм, который проверяет, имеет ли система линейных уравнений решение, и не противоречит ли данная система какому-либо дизъюнкту формулы. Таким образом, время работы алгоритма, использующего расщепление по линейным комбинациям (в отличие от произвольных функций) не более, чем полиномиальным сомножителем от размера дерева.

Формулы, которые кодируют невыполнимые системы линейных уравнений, сложны для резолюций [6], [8], а следовательно сложны и для DPLL-алгоритмов. Выполнимые формулы, которые кодируют системы линейных уравнений, являются сложными примерами для близоруких и пьяных DPLL-алгоритмов [22], [24]. Примеры трудных формул из главы 3 для DPLL-алгоритмов с эвристикой отсечения также основаны на системах линейных уравнений. В главе 4 мы покажем, что расщепление по линейным комбинациям переменных помогает решить формулы, кодирующие линейные системы уравнений над полем $\mathbb{F}_2$, за полиномиальное время.

Для любой формулы ϕ в КНФ обозначим за $\phi^{\oplus}$ формулу в КНФ, полученную из ϕ путем подстановки $x_1 \oplus x_2$ вместо каждой переменной x (здесь x_1, x_2 — новые переменные). Уркарт [26] показал, что для невыполнимой формулы ϕ время работы любого DPLL-алгоритма на формуле $\phi^{\oplus}$ не менее $2^{d(\phi)}$, где $d(\phi)$ — минимальная глубина дерева рекурсивных вызовов DPLL-алгоритма на формуле ϕ. Уркарт также дал пример такой противоречивой формулы $Peb(G_n)$, что $d(Peb(G_n)) = \Omega(n/\log n)$, но существует DPLL-алгоритм, который решает $Peb(G_n)$ за $O(n)$ шагов. Таким образом, $Peb^{\oplus}(G_n)$ является еще одним примером формул, являющимся трудными для DPLL-алгоритмов, но простыми для расщеплений по линейным комбинациям.

Недавний алгоритм Сето и Тамаки [27], который решает задачу выполнимости формул над полным базисом, использует расщепление по линейным комбинациям переменных. Похожая идея была использована

Деменковым и Куликовым для упрощения нижней оценки $3n - o(n)$ для булевых схем над полным базисом [28]. Общая идея [27] и [28] состоит в устранении подформулы, выражающей линейную функцию.

Одним из результатов главы 4 является доказательство нижних оценок на время работы алгоритмов, использующих расщепление по линейным комбинациям. Мы докажем экспоненциальную нижнюю оценку на размер дерева расщеплений по линейным комбинациям для 2-кратных цейтинских формул (TS^2) [29], которые могут быть получены из обычных цейтинских путем подстановки вместо каждой переменной конъюнкции двух новых.

Теорема 2. За полиномиальное от n время можно построить граф $G(V, E)$ на n вершинах с максимальной степенью, ограниченной константой, и такую функцию $c : V \to \mathbb{F}_2$, что $TS^2_{G,c}$ является невыполнимой и размер любого дерева линейных расщеплений $TS^2_{(G,c)}$ составляет по крайней мере $\Omega\left(2^{n^{1/3}/\log^3(n)}\right)$.

Доказательство данной теоремы использует связь нижних оценок на деревья расщепления и коммуникационной сложности задачи, связанной с 2-кратными цейтинскими формулами.

В главе 4 мы также приведем краткое доказательство нижней оценки на размер деревьев расщеплений по линейным формулам для принципа Дирихле (PHP^m_n), полученное Ицыксоном в работе [30]. Данное доказательство не использует технику коммуникационной сложности.

С другой стороны, в работе [30] Ицыксоном было показано существование полиномиального дерева расщеплений по линейным комбинациям для формул $PM(K_{n,n+1})$, которые кодируют существование совершенного паросочетания в полном двудольном графе $K_{n,n+1}$. Формулы $PM(K_{n,n+1})$ сложны для резолюций [31].

Экспандеры

Существует несколько определений графов-экспандеров, нам потребуются комбинаторные двудольные экспандеры.

Определение 2 ([32])**.** Двудольный граф $G = (X, Y, E)$ называется (r, d, c)-экспандером, если

- степени всех вершин из множества Y равняются d;
- для любого множества $A \subseteq Y, |A| \leq r$ выполняется $\Gamma(A) \geq c|A|$.

Граф G называется (r, d, c)-граничным экспандером, если второе условие формулируется так:

- для любого множества $A \subseteq Y, |A| \leq r$ выполняется $\delta(A) \geq c|A|$.

Графы-экспандеры находят применение в различных областях теории сложности, дискретной математики, а также в практических приложениях, перечислим некоторые из них [33, 32]:

- построение кодов, исправляющих ошибки с линейным временем декодирования;
- проверка принадлежности набора чисел конечному подмножеству натуральных чисел с малым числом запросов;
- помехоустойчивое распределенное хранение данных;
- дерандомизация или частичная дерандомизация вероятностных алгоритмов;
- построение трудных примеров для различных систем доказательств.

Несложно показать, что случайный граф, с вероятностью почти 1, является экспандером для $d = O(\log |X|)$, $c = \Omega(d)$ и $r = \Omega(n)$ [32], однако для большинства из указанных выше задач требуются явные конструкции таких графов, либо параметры, которые дают случайные конструкции недостаточны для поставленных задач. Существует множество

работ, в которых приводятся различные явные конструкции графов-экспандеров [33, 34, 35, 36]. В диссертации мы построим по произвольной явной конструкции графа-экспандера экспандер, обладающий дополнительными свойствами: полный ранг матрицы смежности над полем $\mathbb{F}_2$, степень всех вершин ограничена константой. Воспользовавшись конструкцией из работы [33] мы докажем следующую лемму.

Лемма 1. Для любого достаточно большого d и любого n существует явная конструкция $(r, d, 0.75d)$-экспандера с размерами долей $|X| = |Y| = n$, $r = \Omega(n)$ и со степенью вершин из доли X не более $20kd$, где k — достаточно большая константа.

Данную конструкцию мы применим для построения сложных в среднем примеров для DPLL-алгоритмов с эвристикой отсечения ветвей (теорема 1).

Односторонние функции

Мы называем функцию f односторонней, если ее легко вычислить, но трудно обратить. Обычно принято считать, что функция легко вычислима, если она вычислима за полиномиальное время (из этого следует, что задача обращения функции лежит в классе **NP**). Есть несколько подходов для определения трудности обращения функции.

1. *Односторонняя в наихудшем случае функция.* Полиномиально вычислимая функция называется односторонней в наихудшем случае, если обратная функция не вычислима за полиномиальное время. Основной недостаток этого определения заключается в том, что трудных для обращения входов может быть очень мало, и на самом деле функция может быть легко обращена почти на всех входах. Существование односторонних в наихудшем случае функций эквивалентно $\mathbf{P} \neq \mathbf{NP}$.

2. *Криптографически односторонняя функция.* Полиномиально вычислимая функция называется криптографически слабо односторонней, если для некоторой положительной константы c каждый вероятностный полиномиальный по времени алгоритм ошибается при обращении этой функции с вероятностью (по входам и случайным битам алгоритма) не менее $\frac{1}{n^c}$ для всех достаточно больших длин входов. Не известно никаких разумных предположений о сложностных классах, из которых следовало бы существование криптографически односторонних функций.

Под односторонними функциями мы будем подразумевать криптографические.

В 2000 году Голдрейх в своей работе [5] предложил в качестве кандидата на роль односторонней функции конструкцию основанную на графах-экспандерах. Его конструкция имеет два параметра: двудольный граф с n вершинами в каждой доле и степенью d каждой вершины в правой доле (где d — некоторая константа, не зависящая от n, или функция, растущая не быстрее $O(\log(n))$), а также предикат $P : \{0,1\}^d \to \{0,1\}$. Для того, чтобы вычислить функцию на входе $x \in \{0,1\}^n$, сопоставим биты входа вершинам из левой доли графа, после чего пометим каждую вершину в правой доле значением предиката P, примененного к соседям вершины. Значением функции будет последовательность пометок вершин правой доли.

Функция Голдрейха и криптография в $\mathbf{NC^0}$. Функция вычислима в классе $\mathbf{NC^0}$ тогда и только тогда, когда каждый бит выхода зависит только от константного числа входных битов. Несложно заметить, что любая $\mathbf{NC^0}$-вычислимая функция является обобщением функции Голдрейха, где в разных вершинах правой доли применяются различные предикаты.

Крайан и Милтерсен в работе [37] впервые поставили вопрос о существовании криптографических примитивов (например, псевдослучай-

ных генераторов), вычислимых в классе $\mathbf{NC^0}$. Моссел, Шпилка и Тревисан в работе [38] предъявили функцию $f : \{0,1\}^n \to \{0,1\}^{cn}$, основанную на двудольном графе и фиксированном предикате $P(x_1, \ldots, x_5) := x_1 \oplus x_2 \oplus x_3 \oplus (x_4 \wedge x_5)$, и показали, что данная функция является ϵ-смещенным генератором, то есть данная функция является хорошим кандидатом на роль односторонней функции.

Эплбаум, Ишай и Кушелевиц [39], [40] показали, что в стандартных предположениях существует односторонняя функция, вычислимая в $\mathbf{NC^0}$.

Функция Голдрейха и DPLL-алгоритмы. В работе [22] доказывается экспоненциальная нижняя оценка на время работы двух классов DPLL-алгоритмов на выполнимых формулах: "близоруких" и "пьяных" (см. раздел 1.2). В работе [23] был впервые предложен криптографический взгляд на [22]. Именно, в [23] было замечено, что нижняя оценка для близоруких алгоритмов была доказана на формулах, которые кодируют задачу обращения функции Голдрейха с линейным предикатом. Однако линейная функция Голдрейха неинтересна с криптографической точки зрения, так как она быстро обращается при помощи метода Гаусса. Мотивацией в работе [23] было доказательство нижней оценки для функции, которую действительно трудно обратить. В [23] были рассмотрены функции Голдрейха, основанные на случайном графе (который является экспандером с большой вероятностью) и на предикате $x_1 + x_2 + \cdots + x_{d-2} + x_{d-1}x_d$. Была доказана экспоненциальная нижняя оценка для ослабленного варианта близоруких алгоритмов (по сравнению с [22] эти алгоритмы не могли использовать правила упрощения "чистые литералы", и на каждом шаге алгоритмам разрешалось читать только константное число дизъюнктов). В работах [24] и [41] была доказана экспоненциальная нижняя оценка на среднюю сложность обращения функций Голдрейха, основанных на случайном графе и предикате вида $x_1 + x_2 + \cdots + x_{d-k} + Q(x_{d-k+1}, \ldots, x_d)$, где Q — произвольный k-местный предикат, $k < d/4$, "пьяными" алгоритмами. На самом деле,

доказательство [23] работает и для предикатов такого вида. В работе [42] была доказана нижняя оценка на среднюю сложность обращения функций Голдрейха, основанных на случайном графе и обширном классе предикатов, включающем случайные предикаты.

Во всех перечисленных работах [22, 23, 24, 42] функция Голдрейха строилась неявно, конструкция графа зависимостей была вероятностной. В главе 2 мы предлагаем явную конструкцию функции Голдрейха, основанную на экспандерах из [33]. На таких функциях дословно работают доказательства из [24, 42] экспоненциальных нижних оценок для "пьяных" и "близоруких" алгоритмов. В разделе 2.3 приводится доказательство экспоненциальной нижней оценки для "близоруких" алгоритмов, поскольку предлагаемое доказательство технически гораздо проще, чем доказательство в [22], [23] и в [42]. Однако, в отличие от работ [23, 24, 42] для полученной функции известен алгоритм обращения за время $2^{o(n)}$.

Основным результатом главы 2 является теорема:

Теорема 3. Для любого $\epsilon > 0$ существует такая явная конструкция нелинейной функции Голдрейха f, что для любого "близорукого" алгоритма A выполнено $\Pr_{y,s}[t_A(\Phi_{f(x)=f(y)}) \geq 2^{\Omega(n)}] \geq 1 - 2^{-\Omega(\frac{n}{K})}$, где $t_A(x)$ — время работы алгоритма A на входе x, s — строка случайных битов, которые использует алгоритм A, и $K = n^{1-\epsilon}$.

Структура диссертации

В главе 1 приводятся определения основных понятий, используемых в диссертации. В главе 2 предъявляется явная конструкция функции Голдрейха, для которого доказывается нижняя оценка на время обращения близорукими DPLL-алгоритмами. В главе 3 рассматриваются DPLL-алгоритмы с ошибкой, и строится семейство трудных примеров для них. В главе 4 рассматриваются алгоритмы расщепления по линейным комбинациям переменных и доказываются верхние и нижние оценки на данные алгоритмы. Также в главе 4 вводятся системы доказательств Res-Lin и

Sem-Lin и доказываются оценки на длины доказательств в данных системах.

Глава 1

Основные понятия

Под языком будем подразумевать множество конечных слов над бинарным алфавитом. Также отождествим язык и его характеристическую функцию.

1. **P** — класс, состоящий из таких языков L, для которых существует детерминированная такая машина Тьюринга M, что $\forall x\ L(x) = M(x)$.

2. **NP** — класс, состоящий из таких языков L, для которых существует недетерминированная такая машина Тьюринга M, что $\forall x\ L(x) = M(x)$.

1.1 Системы доказательств

В данном разделе мы введем определение систем доказательств.

Определение 1.1. ([9], [10], [43]) Системой доказательств для языка L называется полиномиальный по времени алгоритм $\Pi : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}$, для которого выполнены следующие свойства:

- (полнота) если $x \in L$, то существует такая строка y, что $\Pi(x, y) = 1$;
- (корректность) если существует такая строка y, что $\Pi(x, y) = 1$, то $x \in L$.

Соответствующую сроку y мы будем называть доказательством того, что x принадлежит языку L. В приведенном определении мы никак не ограничивали длину доказательства, теперь формализуем понятие *короткого* доказательства.

Определение 1.2. ([9]) Система доказательств Π называется полиномиально ограниченной, если для любого $x \in L$ существует такая строка $y \in \{0,1\}^{p(|x|)}$, что $\Pi(x, y) = 1$.

Если говорить неформально, то полиномиальная ограниченность системы доказательств означает, что для каждого x размер доказательства может быть ограничен полиномом от размера самого x.

Теорема 1.1. ([9]) *Язык* L принадлежит классу **NP** тогда и только тогда, когда для него существует полиномиально ограниченная система доказательств.

Следующая теорема описывает одну из центральных проблем теории сложности, о равенстве классов **NP** и **co-NP**, в терминах систем доказательств. Для формулировки данной теоремы нам потребуется язык `TAUT`, который состоит из пропозициональных тавтологий.

Теорема 1.2. ([9]) **NP** = **co-NP** тогда и только тогда, когда существует полиномиально ограниченная система доказательств для языка `TAUT`.

Так как класс **P** замкнут относительно дополнения, т.е. **P** = **co-P**, то из приведенной теоремы следует, что если для языка `TAUT` нет полиномиально ограниченной системы доказательств, то **P** $\neq$ **NP**.

Теперь научимся сравнивать системы доказательств в терминах сложности доказательств.

Определение 1.3. ([9], [44]) Пусть Π_1 и Π_2 — системы доказательств для языка L. Будем говорить, что система Π_1 моделирует систему Π_2, если существует такой полином p, что для любого $x \in L$ выполнено $|y_{\Pi_1}| \leq p(|y_{\Pi_2}|)$, где y_{Π_1}, y_{Π_2} — кратчайшее доказательство для x в системах Π_1 и Π_2 соответственно.

Также рассмотрим конструктивный аналог сравнения.

Определение 1.4. ([9], [44]) Пусть Π_1 и Π_2 — системы доказательств для языка L. Будем говорить, что система Π_1 p-моделирует систему Π_2, если существует такой полиномиальный по времени алгоритм A, что для любой строки $x \in L$ и любой такой строки y, что $\Pi_2(x, y) = 1$, выполнено $\Pi_1(x, A(x, y)) = 1$.

Резолюционная система доказательств. Резолюционная система доказательств предназначена для доказательства того, что булева формула в КНФ невыполнима (или, что тоже самое, формула в ДНФ принадлежит языку былевых тавтологий `TAUT`). Данная система имеет два правила:

- правило ослабления: из дизъюнкта C можно вывести любой дизъюнкт $(C \vee x)$, где x — произвольный литерал;
- правило резолюции: из дизъюнктов $(x \vee D)$ и $(\neg x \vee D')$ можно вывести дизъюнкт $D \vee D'$.

Вывод дизъюнкта C из КНФ формулы ϕ в резолюционной системе — это последовательность дизъюнктов, которая заканчивается дизъюнктом C, и каждый дизъюнкт которой является либо дизъюнктом формулы ϕ, либо получен из предыдущих дизъюнктов при помощи правил вывода. Доказательство невыполнимости формулы в КНФ — это вывод пустого дизъюнкта. Размером доказательства называется количество элементов в указанной последовательности. Доказательство называется древовидным, если каждый выведенный дизъюнкт используется как посылка правила не более одного раза.

Теорема 1.3. ([6], [12], [7]) Существует такое явное семейство невыполнимых формул в КНФ ϕ_n полиномиального от n размера, что размер любого доказательства невыполнимости указанных формул — суперполиномиальный (экспоненциальный) от n.

Опишем также способ доказательства нижних оценок, предложенный в работе [8], который нам потребуется в дальнейшем. Бен-Сассон и Вигдерсон вводят понятие ширины. Ширина дизъюнкта — это количество литералов в нем. Ширина формулы в КНФ — это ширина самого широкого дизъюнкта. Ширина резолюционного доказательства формулы — это ширина самого широкого дизъюнкта, входящего в него.

Теорема 1.4. ([8]) Размер древовидного резолюционного доказательства формулы φ не меньше, чем 2^{w-w_φ}, где w — это минимальная ширина резолюционного доказательства φ, а w_φ — это ширина φ.

1.2 DPLL-алгоритмы

Мы рассмотрим широкий класс алгоритмов поиска выполняющего набора — алгоритмы, основанные на расщеплении (DPLL-алгоритмы). Впервые данные алгоритмы были определены в работах [3] и [4]. Алгоритм расщепления параметризован двумя *эвристиками* (процедурами):

- процедура **A**, которая по формуле в КНФ выдает переменную из этой формулы. Это переменная, по которой будет проводиться расщепление;

- процедура **B**, которая по формуле в КНФ и ее переменной выдает значение из $\{0, 1\}$. Это значение, которое будет подставляться при расщеплении первым.

Алгоритм может также использовать некоторые синтаксические *правила упрощения*. Такие правила могут заменять формулу на эквивалентную ей в смысле выполнимости.

Алгоритм расщепления — это рекурсивный алгоритм, который получает на вход формулу φ и частичную подстановку ρ и работает следующим образом.

Алгоритм 1.1. Вход: формула φ и подстановка ρ

- Упростить φ с помощью правил упрощения (считаем, что правила упрощения меняют φ и ρ, причем все переменные, подставленные подстановкой ρ, удаляются из формулы φ).

- Если формула стала пустой (т.е. все ее дизъюнкты выполнены подстановкой ρ), то выдать ρ. Если формула содержит пустой дизъюнкт (заведомо невыполненный), то выдать «формула невыполнима».

- $j := \mathbf{A}(\varphi)$.

- $c := \mathbf{B}(\varphi, j)$.

- Запустить алгоритм рекурсивно на $(\varphi[x_j := c], \rho \cup \{x_j := c\})$; если алгоритм выдал подстановку, то выдать ее, в противном случае рекурсивно запустить на $(\varphi[x_j := 1 - c], \rho \cup \{x_j := 1 - c\})$; если рекурсивный вызов выдал подстановку, то выдать ее, иначе выдать «формула невыполнима».

Временем работы алгоритма расщепления будем считать количество рекурсивных вызовов.

1.2.1 Связь с системами доказательств

Поведение алгоритмов расщепления, не использующих правила упрощения, на невыполнимых формулах тесно связано с резолюционной системой доказательств.

По работе любого алгоритма алгоритма расщепления, не использующего правила упрощения, на невыполнимой формуле можно построить дерево расщепления. Каждая вершина дерева (кроме листьев) помечена переменной, по которой производится расщепление в этой вершине, из каждой вершины (кроме листьев) исходит два ребра, одно из них помечено значением 0, другое значением 1. В каждом из листьев опровергается как минимум один из дизъюнктов исходной формулы. Размер дерева расщеплений служит нижней оценкой на время работы алгоритма расщепления. Индукцией по размеру дерева легко показать, что ес-

ли у невыполнимой формулы φ есть дерево расщеплений размера k, то можно построить древовидное резолюционное доказательство φ размера k. База индукции очевидна, так как, если дерево содержит одну вершину, то формула обязана содержать пустой дизъюнкт. Для индукционного перехода заметим, что дерево обязано содержать два листа u и v, у которых есть общий родитель w. Пусть расщепление в вершине w проводилось по переменной x_i, и лист u соответствовал присваиванию $x_i := 1$, а лист v — присваиванию $x_i := 0$. Тогда дизъюнкты, которые опровергаются в вершинах v и u, содержат переменную x_i с разными знаками, а их резольвента (результат применения правила резолюции) C обязана опровергаться в вершине w. Составим новое дерево расщеплений следующим образом: от старого отрежем листья u и v, а в вершину w запишем дизъюнкт C. Получилось корректное дерево расщеплений уже для другой формулы, которая получается из исходной добавлением резольвенты для двух дизъюнктов. К получившемуся дереву можно применить индукционное предположение (число узлов дерева уменьшилось). Таким образом, мы получаем следующее утверждение:

Предложение 1.1. Время работы любого алгоритма расщепления на невыполнимой формуле не меньше, чем размер (количество дизъюнктов) кратчайшего древовидного резолюционного доказательства.

1.2.2 "Пьяные" и "близорукие" алгоритмы

Определение 1.5. ([22]) Близорукие алгоритмы — это такие алгоритмы расщепления, в которых обе эвристики **A** и **B** имеют следующие ограничения:

- они видят формулу, в которой все знаки отрицаний стерты;
- для каждой переменной им известно количество положительных и отрицательных ее вхождений в формулу;
- они могут запросить $K(n) = o(n)$ дизъюнктов, которые можно прочитать полностью (со знаками отрицаний).

Правила упрощения:

- *Правило удаления единичного дизъюнкта*: если формула содержит дизъюнкт, в котором ровно один литерал, то сделать такое присваивание переменной этого литерала, которое выполнит этот дизъюнкт.
- *Правило чистых литералов*: если формула содержит переменную, которая входит только с одним знаком, то присвоить ей такое значение, которое выполнит все дизъюнкты, в которые эта переменная входит.

Определение 1.6. ([22]) Пьяные алгоритмы — это такие алгоритмы расщепления, в которых эвристики **A** и **B** имеют следующие ограничения:

- **B** выдает произвольное значение с вероятностью $\frac{1}{2}$;
- **A** — любая (возможно, даже невычислимая);
- правила упрощения — правило удаления единичного дизъюнкта и правило чистых литералов.

1.3 Функция Голдрейха

Голдрейх в работе [5] предложил общую конструкцию функции $f : \{0,1\}^n \to \{0,1\}^n$, заданной с помощью двудольного графа G и предиката $P : \{0,1\}^d \to \{0,1\}$. Каждая строка из $\{0,1\}^n$ задает некоторое значение на множестве переменных $X = \{x_1, x_2, \ldots, x_n\}$, отождествленном с вершинами левой доли графа G, значение $(f(x))_j$ (j-й символ строки $f(x)$) рассчитывается следующим образом: пусть в вершину y_j правой доли входят ребра из вершин $x_{j_1}, x_{j_2}, \ldots, x_{j_d}$, тогда $(f(x))_j = P(x_{j_1}, x_{j_2}, \ldots, x_{j_d})$.

1.3.1 Формулы, построенные по функции Голдрейха

Закодируем задачу обращения функции Голдрейха в виде задачи выполнимости булевой формулы.

Пусть $g : \{0,1\}^\ell \to \{0,1\}$, канонической записью g в КНФ мы называем такую формулу: для каждого набора $c \in \{0,1\}^\ell$, для которого $g(c) = 0$, пишется дизъюнкт $x_1^{c_1} \vee x_2^{c_2} \vee \cdots \vee x_\ell^{c_\ell}$, где $x_i^0 = x_i$, а $x_i^1 = \neg x_i$.

Пусть f — это функция Голдрейха, построенная по предикату P и графу G. Уравнение $f(x) = b$ мы представляем следующим образом: для каждой вершины $y_j \in Y$, которая соединена с вершинами $x_{j_1}, x_{j_2}, \ldots, x_{j_d}$, мы записываем каноническую формулу в КНФ от переменных $x_{j_1}, x_{j_2}, \ldots, x_{j_d}$, которая кодирует равенство $b_j = P(x_{j_1}, x_{j_2}, \ldots, x_{j_d})$. Полученную формулу будем обозначать $\Phi_{f(x)=b}$. Часть формулы, которая соответствует вершинам из множества $A \subseteq Y$, будем обозначать $\Phi^A_{f(x)=b}$.

Лемма 1.1. Если функция $g : \{0,1\}^\ell \to \{0,1\}$ линейна как минимум по двум переменным, то в канонической записи g в виде КНФ ровно $2^{\ell-1}$ дизъюнктов, а каждая переменная имеет одинаковое количество положительных и отрицательных вхождений.

Доказательство. Пусть g в поле $\mathbb{F}_2$ имеет вид $x_1 + x_2 + h(x_3, \ldots, x_\ell)$. Обозначим $T_0 = h^{-1}(0)$, $T_1 = h^{-1}(1)$. Тогда $g^{-1}(0) = \{00y \mid y \in T_0\} \cup \{11y \mid y \in T_0\} \cup \{01x \mid y \in T_1\} \cup \{10y \mid y \in T_1\}$. Отсюда видно, что $|g^{-1}(0)| = 2^{l-1}$, и каждая переменная будет иметь поровну положительных и отрицательных вхождений. □

Под канонической записью уравнения будем подразумевать каноническую запись булевой фукнции, которая по подстановке значений переменным возвращает 1 тогда и только тогда, когда при применении данной подстановке уравнение обращается в истину. Из леммы 1.1 следует, что близорукий алгоритм, не видя знаков отрицаний, не отличит равенство $g(x) = 0$ от $g(x) = 1$, если g линейно как минимум по 2

переменным. Также нетрудно видеть, что после подстановки значения переменной, каноническая форма перейдет в каноническую форму.

1.4 Экспандеры

Мы рассматриваем двудольные графы. В каждой из долей находится по n вершин, первую долю мы обозначаем $X = \{x_1, x_2, \ldots, x_n\}$, вторую $Y = \{y_1, y_2, \ldots, y_n\}$. Для каждой вершины доли Y есть упорядоченный список ее соседей из доли X, причем повторения в этом списке (кратные ребра) разрешаются. Все рассматриваемые нами графы будут d-регулярными, т.е. степень любой вершины множества Y равняется d, где d — это константа.

Каждому графу мы ставим в соответствие матрицу $n \times n$ над полем $\mathbb{F}_2$. Строки этой матрицы соответствуют вершинам множества Y, а столбцы — вершинам множества X. Число, стоящее в клетке с координатами (y, x) соответствует четности числа ребер между вершинами y и x. Такую матрицу будем называть матрицей смежности графа.

Для $A \subseteq Y$ обозначим через $\Gamma(A)$ (множество соседей A) множество вершин из X, соединенных как минимум с одной вершиной из A, а через $\delta(A)$ (граница A) — множество вершин из X, которые соединены ровно с одной вершиной из A одним ребром.

Определение 1.7. Двудольный граф G называется (r, d, c)-экспандером, если

- степени всех вершин из множества Y равняются d;
- для любого множества $A \subseteq Y, |A| \leq r$ выполняется $\Gamma(A) \geq c|A|$.

Граф G называется (r, d, c)-граничным экспандером, если второе условие формулируется так:

- для любого множества $A \subseteq Y, |A| \leq r$ выполняется $\delta(A) \geq c|A|$.

Лемма 1.2 ([22])**.** Каждый (r, d, c)-экспандер является граничным $(r, d, 2c - d)$-экспандером.

Нам понадобятся граничные экспандеры, для этого достаточно иметь экспандеры с константой $c > d/2$. К примеру, случайный граф является подходящим экспандером:

Лемма 1.3 ([32])**.** При $d \geq 32$ для достаточно больших n случайный двудольный d-регулярный граф, в котором в долях X и Y по n вершин и для каждой вершины из Y выбираются случайным образом равновероятно независимо d ребер (повторения разрешаются), является $(\frac{n}{10d}, d, \frac{5}{8}d)$-экспандером с вероятностью не менее 0.9.

Следствие 1.1. В условиях леммы этот граф является граничным $(\frac{n}{10d}, d, \frac{1}{4}d)$-экспандером.

Доказательство. Следует из леммы 1.2. □

Существуют также и явные конструкции таких экспандеров:

Лемма 1.4 ([33])**.** Для каждой константы $\epsilon > 0$ существует такая константа d, что за полиномиальное от n время можно построить (r, d, c)-экспандер, в котором $c = (1 - \epsilon)d$, $r = \Omega(n/d)$.

Следствие 1.2. Такой граф будет граничным $(\Omega(n/d), d, (1 - 2\epsilon)d)$-экспандером.

1.5 Распределения на входах

В теории сложности в среднем случае вычислительные задачи рассматриваются в совокупности с распределением на входах. Одним из основных подходов, описанным в работах [45] и [46], является рассмотрение семейств вероятностных мер с конечным носителем.

Определение 1.8. Ансамбль распределений D — это семейство $\{D_n\}_{n=1}^{\infty}$, где $D_n : \{0,1\}^n \to [0,1]$ и $\sum\limits_{x \in \{0,1\}^n} D_n(x) = 1$. Множество $supp\ D_n = \{x \in \{0,1\} \mid D_n(x) > 0\}$ называется носителем D_n. Носителем ансамбля распределений называется объединение носителей: $supp\ D = \bigcup\limits_{n=1}^{\infty} supp\ D_n$.

Ансамбль D_n называется полиномиально моделируемым, если существует такой полином p и такой полиномиальный детерминированный алгоритм S (сэмплер), что если x равномерно распределен на множестве $\{0,1\}^{p(n)}$, то выход алгоритма $S(1^n, x)$ распределен согласно распределению D_n. Множество всех полиномиально моделируемых распределений будем обозначать **PSamp**.

Глава 2

Нижние оценки на сложность обращения функции Голдрейха близорукими DPLL-алгоритмами

В данной главе мы рассмотрим явную конструкцию функции Голдрейха, основанную на экспандерах из [33]. На таких функциях дословно работает доказательство из [24] об экспоненциальных нижних оценках для "пьяных" алгоритмов. Мы продемонстрируем экспоненциальную оценку для близоруких алгоритмов, а кроме того докажем нижнюю оценку для общего варианта близоруких алгоритмов, а не для ослабленного, как это делается в [23].

Предлагаемая функция Голдрейха имеет следующую структуру: она состоит из суммы линейной функции Голдрейха и нелинейной. Линейная функция нужна для того, чтобы DPLL-алгоритмам было сложно ее обратить, а нелинейная нужна для того, чтобы получившаяся функция не была бы тривиально обратимой. В основе линейной функции лежат экспандеры из [33], а нелинейная может быть фактически произвольной, единственное ограничение — она может зависеть только от $n^{\varepsilon/2}$ перемен-

ных, а не от всех. На самом деле, нам неважно, что в нелинейной части используется один и тот же предикат, доказательство нижней оценки проходит для произвольной нелинейной части.

План доказательства следующий: сначала мы немного модифицируем экспандер, чтобы он остался экспандером, а ранг его матрицы смежности стал бы большим. Благодаря тому, что нелинейная часть функции зависит лишь от малого числа переменных, мы доказываем, что построенная нами функция Голдрейха является почти биекцией. Сначала мы доказываем нижнюю оценку для невыполнимых формул, пользуясь техникой, разработанной для метода резолюций [8]. Далее, пользуясь похожестью нашей функции на линейную и биективную, мы доказываем, что с большой вероятностью за первые несколько шагов текущая формула станет невыполнимой, и применяем доказанную оценку для невыполнимых формул.

Результаты данной главы опубликованы в работах [47, 48].

2.1 Почти биективная функция Голдрейха

Будем рассматривать такие двудольные графы (X, Y, E), что $|X| = |Y|$.

2.1.1 Линейная функция

Пусть G_1 — d_1-регулярный граф (X, Y, E_1), а G_2 — d_2-регулярный граф (X, Y, E_2). Определим граф $G_1 + G_2$, это будет $(d_1 + d_2)$-регулярный граф $(X, Y, E_1 \cup E_2)$, где под объединением мы понимаем следующую операцию: списки соседей для каждой вершины множества Y получаются в результате добавления списка из графа G_2 в конец списка из графа G_1. Матрица смежности графа $G_1 + G_2$ равняется сумме матриц смежностей для графов G_1 и G_2.

Предложение 2.1. Если граф G является (r, d, c)-экспандером, а G' — просто d'-регулярный, то $G + G'$ является $(r, d + d', c)$-экспандером.

Теорема 2.1. По любому графу G можно за полиномиальное от n время построить такой 1-регулярный граф T, что ранг матрицы смежности $G + T$ не менее $n - 1$.

Сначала докажем вспомогательную лемму.

Лемма 2.1. Пусть $a = (\alpha_1, \ldots, \alpha_n) \in \mathbb{F}_2^n$. Тогда из векторов $b_i = (\alpha_1, \ldots, \alpha_{i-1}, \alpha_i + 1, \alpha_{i+1}, \ldots, \alpha_n)$ можно выбрать как минимум $n - 1$ линейно независимых.

Доказательство. Составим матрицу A размера $n \times n$, все столбцы которой совпадают с вектором a. Вектора b_i являются столбцами матрицы $A + E$, где E — единичная матрица. Поскольку ранг суммы матриц не превосходит суммы рангов, то $n = \operatorname{rk} E \leq \operatorname{rk}(A + E) + \operatorname{rk} A$. Поскольку все столбцы матрицы A одинаковые, то $\operatorname{rk} A \leq 1$, отсюда $\operatorname{rk}(A + E) \geq n - 1$. □

Доказательство теоремы 2.1. Опишем построение графа T, начнем с пустого множества ребер и будем добавлять ребра по одному. На i-ом шаге при $1 \leq i \leq n - 1$ будем добавлять соседа к вершине $y_i \in Y$ так, чтобы первые i строчек матрицы смежности $G + T$ были бы линейно независимы. Это всегда можно сделать по лемме 2.1 (роль вектора a играет i-я строка матрицы смежности графа G). Для вершины y_n добавим соседа произвольным образом. □

Следствие 2.1. Если граф G является (r, d, c)-экспандером, то построенный в теореме граф $G + T$ будет $(r, d + 1, c)$-экспандером, а функция Голдрейха f, построенная по $G + T$ и линейному предикату будет обладать свойством: для каждого $b \in \{0, 1\}^n$ размер множества $f^{-1}(b)$ не превосходит 2.

2.1.2 Немного нелинейная функция Голдрейха

Пусть $R \subseteq X$ — некоторое подмножество X. Будем называть R-графом такой регулярный граф, в котором все вершины из $X \setminus R$ имеют степень 0.

Лемма 2.2. Пусть G — $(d-k)$-регулярный граф, матрица смежности которого имеет ранг $n-1$, а H — k-регулярный R-граф. Пусть f — функция Голдрейха, построенная по $G+H$ и предикату $x_1+x_2+\cdots+x_{d-k}+Q(x_{d-k+1},\ldots,x_d)$, где Q — произвольный k-местный предикат. Тогда для каждого $b\in\{0,1\}^n$ размер множества $f^{-1}(b)$ не превосходит $2^{|R|+1}$.

Доказательство. Рассмотрим систему уравнений $f(x)=b$, зафиксируем значения всех переменных из множества R. Получим линейную систему уравнений, которая совпадает с линейной системой, которую можно получить следующим образом: рассмотреть линейную систему с матрицей графа G и подставить в нее значения для переменных множества R. Поскольку матрица G имеет ранг $n-1$, то получившаяся линейная система имеет не более двух решений. Значит, исходная система $f(x)=b$ имеет не более $2^{|R|+1}$ решений. □

2.2 Нижняя оценка времени работы на невыполнимых формулах

В данном разделе мы приведем пример невыполнимых формул, на которых любой близорукий DPLL-алгоритм работает экспоненциальное время.

Пусть G — это граничный (r,d,c)-расширитель. С каждой вершиной множества X отождествим пропозициональную переменную. Пусть каждой вершине $y_j\in Y$ сопоставлена формула в КНФ от переменных, соответствующих смежных с y_j вершинам. Обозначим формулу, стоящую в вершине y_j, через φ_j. Очевидно, что ширина каждого дизъюнкта формулы φ_j не превосходит d. Формулу, соответствующую всему множеству Y, обозначим за Φ. Для множества $A\subseteq Y$ конъюнкцию формул, соответствующих вершинам из множества A, будем обозначать Φ^A.

Будем говорить, что переменная чувствительна для формулы, если заменой значения этой переменной можно поменять значение формулы

на противоположное при любом фиксированном значении остальных переменных.

Теорема 2.2 ([24])**.** Пусть каждая формула φ_j содержит не более k нечувствительных переменных, ρ — такая частичная подстановка переменным из X, что

- формула $\Phi|_\rho$ невыполнима;
- для любого множества вершин $A \subseteq Y$, такого, что $|A| < \frac{r}{2}$, формула $\Phi|_\rho^A$ выполнима.

Тогда любое резолюционное доказательство формулы $\Phi|_\rho$ имеет ширину как минимум $\frac{(c-k)r}{4} - |\rho|$.

Доказательство. Рассмотрим меру Бен-Сасона-Вигдерсона, которая определена на дизъюнктах резолюционного доказательства формулы $\Phi|_\rho$. Пусть $\mu(D)$ — это размер минимального числа таких вершин A, что из формулы $\Phi^A|_\rho$ семантически следует дизъюнкт D (это означает, что любой выполняющий набор формулы $\Phi^A|_\rho$ является выполняющим набором для дизъюнкта D). Мера μ обладает полуаддитивностью, т.е., если дизъюнкт D получается применением правила резолюции из дизъюнктов C_1 и C_2, то $\mu(D) \leq \mu(C_1) + \mu(C_2)$. Поскольку для любого $A \subseteq Y$, для которого $|A| < \frac{r}{2}$, формула $\Phi|_\rho^A$ выполнима, то мера пустого дизъюнкта не меньше $\frac{r}{2}$. Из полуаддитивности следует, что в доказательстве существует дизъюнкт C, для которого выполняется $\frac{r}{2} > \mu(C) \geq \frac{r}{4}$. Пусть A — это такое минимальное множество вершин, что из формулы $\Phi|_\rho^A$ семантически следует дизъюнкт C, т.е. $|A| = \mu(C) \geq \frac{r}{4}$. В графе G выполняется $\delta(A) \geq c|A|$. Заметим, что $\delta(A)$ — это множество переменных, которые входят ровно в одну формулу из множества A. Из них не менее $(c-k)|A|$ переменных, которые являются чувствительными для хотя бы одной из вершин множества A. В формулу $\Phi|_\rho^A$ входит как минимум $(c-k)|A| - |\rho|$ чувствительных переменных. Покажем, что в дизъюнкт C входят все чувствительные переменные. Действительно, если это не так, то найдется переменная x_j, чувствительная для вершины

$v \in A$, которая не входит в дизъюнкт C. Рассмотрим множество $A \setminus \{v\}$, из него семантически не следует дизъюнкт C, значит существует такое значение переменных, при которых все формулы из $A \setminus \{v\}$ истинны, а C ложно. Но заменой значения переменной x_j легко сделать, чтобы все формулы из A были бы истинны, а C ложно, противоречие с тем, что C семантически следует из A. □

Следствие 2.2 ([24])**.** Размер дерева расщеплений для формулы $\Phi|_\rho$ не меньше $2^{\frac{(c-k)r}{4}-|\rho|-d}$.

Доказательство. Следует из теоремы 2.2, теоремы 1.4 и предложения 1.1. □

2.3 Нижняя оценка времени работы на выполнимых формулах

2.3.1 k-замыкание

Пусть граф G — граничный (r, d, c)-экспандер. Пусть $0 < k < c - 2$. Предикат $P(x_1, \ldots, x_d) = x_1 + \cdots + x_{d-k} + Q(x_{d-k+1}, \ldots, x_d)$; функция Голдрейха f задана графом G и предикатом P.

Определение 2.1 ([22])**.** Пусть $J \subseteq X$, множество вершин $I \subseteq Y$ будем называть k-замыканием множества J, если существует такая конечная последовательность множеств $I_1, I_2, \ldots, I_m$ (обозначим $C_\ell = \bigcup_{1 \le i \le \ell} I_i$, $C_0 = \emptyset$), что выполняются следующие свойства:

- $I_\ell \subseteq Y$ и $0 < |I_\ell| \le \frac{r}{2}$ для всех $1 \le \ell \le m$;
- $I_i \cap I_j = \emptyset$ для всех $1 \le i, j \le m$;
- $|\delta(I_\ell) \setminus (\Gamma(C_{\ell-1}) \cup J)| \le (1 + k)|I_\ell|$ для всех $1 \le \ell \le m$;
- для всех $I' \subseteq Y \setminus C_m$, если $0 < |I'| \le \frac{r}{2}$, то $|\delta(I') \setminus (\Gamma(C_m) \cup J)| > (1 + k)|I'|$;

- $I = C_m$.

Множество всех k-замыканий множества J будем обозначать $Cl^k(J)$.

Лемма 2.3 ([22], [24]). 1. Для любого множества $J \subseteq X$ существует k-замыкание.

2. Пусть $J \subseteq J^*$, тогда для любого $I \in Cl^k(J)$ существует $I^* \in Cl^k(J^*)$, что $I \subseteq I^*$.

Доказательство. 1. k-замыкание может быть получено как результат работы следующего алгоритма $\mathcal{C}$ на входе $(J, \emptyset)$.

Алгоритм 2.1. Алгоритм $\mathcal{C}(J, I_0)$

(a) $I := I_0$ (переменная I обозначает некоторое подмножество Y).

(b) Пока существует такое множество $I' \subseteq Y \setminus I$, что $0 < |I'| \leq \frac{r}{2}$, $|\delta(I') \setminus (\Gamma(I) \cup J)| \leq (1 + k)|I'|$

- $I := I \cup I'$.

(c) Выдать I.

2. Пусть $I \in Cl^k(J)$, обозначим через I^* результат работы работы алгоритма $\mathcal{C}$ на входе (J, I). Нетрудно видеть, что $I \subseteq I^*$. Покажем, что $I^* \in Cl^k(J)$. Действительно, последовательность $I_1, \ldots, I_m$ из определения 2.1 получается объединением соответствующей последовательности для множества I и последовательности значений переменной I', которые получаются в процессе работы алгоритма $\mathcal{C}$.

□

Лемма 2.4 ([22]). Пусть $|J| < \frac{(c-k-1)r}{2}$, тогда для любого множества $I \in Cl^k(J)$ выполняется неравенство $|I| \leq (c - k - 1)^{-1}|J|$.

Доказательство. От противного. Пусть $I_1, I_2, \ldots, I_m$ — это последовательность, соответствующая k-замыканию I, $C_\ell = \bigcup_{1 \leq i \leq \ell} I_\ell$. Пусть t — это наименьшее число, при котором выполняется неравенство $|C_t| >$

$(c-k-1)^{-1}|J|$, тогда $|C_t| \leq (c-k-1)^{-1}|J| + \frac{r}{2} \leq r$. Тогда $|\delta(C_t)| \geq c|C_t| > |J| + (k+1)|C_t|$. Индукцией по ℓ можно показать

$$|\delta(C_\ell) \setminus J| \leq (k+1)|C_\ell|, \tag{2.1}$$

что противоречит полученному ранее неравенству при $l = t$. При $l = 1$ неравенство (2.1) следует из того, что $|\delta I_1 \setminus J| \leq (k+1)|I_1|$. $|\delta(C_\ell) \setminus J| \leq |\delta(I_1 \cup \cdots \cup I_{\ell-1}) \setminus J| + |\delta(I_\ell) \setminus (J \cup \Gamma(C_{\ell-1}))| \leq (k+1)(|C_{\ell-1}|) + (k+1)|I_\ell|$. □

Определение 2.2 ([22], [24])**.** Пусть $f : \{0,1\}^n \to \{0,1\}^n$ — функция Голдрейха, построенная по графу G и предикату P, $b \in \{0,1\}$. Частичная подстановка ρ называется локально корректной для уравнения $f(x) = b$, если существует такая строка $z \in \{0,1\}^n$, согласованная с ρ и $I \in Cl^k(Vars(\rho))$, что выполняется равенство $f(z)|_I = b|_I$.

Лемма 2.5 ([22])**.** Если частичная подстановка ρ локально корректна для уравнения $f(x) = b$, то для любого $Z \subseteq X$, $|Z| \leq \frac{r}{2}$ существует такая строка $z \in \{0,1\}^n$, согласованная с ρ, что выполняется равенство $f(z)|_Z = b|_Z$.

Доказательство. От противного. Возьмем минимальное по размеру $Z \subseteq Y$ такое, что $|Z| \leq \frac{r}{2}$ и при всех z, согласованных с ρ выполняется $f(z)|_Z \neq b|_Z$. Пусть $I \in Cl^k(Vars(\rho))$ из определения 2.2. Поскольку ρ локально корректна, то $Z \setminus I \neq \emptyset$.

Из определения замыкания следует, что $|\delta(Z \setminus I) \setminus (\Gamma(I) \cup Vars(\rho))| > (k+1)|Z \setminus I|$, следовательно существует такая вершина $y \in Z \setminus I$, что не менее, чем $k+1$ граничная переменная множества Z (не соединенная с замыканием и не являющаяся подставленной) соединена с y. Из минимальности Z следует, что существует такой $z \in \{0,1\}^n$, согласованный с ρ, что $f(z)|_{Z \setminus \{y\}} = b|_{Z \setminus \{y\}}$. Но путем инвертирования бита в z, соответствующего линейной граничной переменной, связанной с y, можно удовлетворить уравнение, соответствующее y. Таким образом, существует такой z', согласованный с ρ, что $f(z')|_Z = b|_Z$. Противоречие. □

2.3.2 “Умный” близорукий алгоритм

Опишем “умный” близорукий алгоритм. Такому алгоритму будет разрешено читать больше дизъюнктов (или, что тоже самое, открывать больше битов выхода). Кроме того, умный близорукий алгоритм не будет делать некоторые подстановки, которые заведомо приводят к невыполнимой формуле. Нетрудно видеть, что достаточно доказать нижнюю оценку на время работы умных близоруких алгоритмов, оценка для всех близоруких алгоритмов будет являться следствием.

Опишем поведение умного близорукого алгоритма формально. Алгоритм хранит текущую частичную подстановку ρ и множество $I \in Cl^k(Vars(\rho))$. В самом начале $\rho = \emptyset$, $I = \emptyset$. На очередном шаге алгоритм упрощает входную формулу, возможно, увеличивая ρ (при этом расширяет множество I, чтобы выполнялось $I \in Cl^k(Vars(\rho))$).

Если умный алгоритм запросил дизъюнкт, который соответствует вершине $y_j \in Y$, то мы будем говорить, что алгоритм открыл j-й бит выхода. Будем считать, что умному алгоритму бесплатно выдаются все остальные дизъюнкты, которые соответствуют $y_j \in Y$.

Пусть эвристика A выбирает переменную x для расщепления. Пусть Z — множество уже открытых алгоритмом битов правой части (в частности сюда входит K битов, открытых перед выбором переменной x). Алгоритм расширяет множество I до элемента из $Cl^k(Vars(\rho) \cup \{x\})$. $Z := Z \cup I$. После этого алгоритм выбирает значение для переменной x так, чтобы формула, соответствующая битам Z, была бы выполнима (если это возможно). Применения правил упрощения также соответствуют расширению ρ и I.

Лемма 2.6 ([22])**.** Для любого умного близорукого алгоритма A найдется другой умный близорукий алгоритм B, который не использует правило чистых литералов и единичных дизъюнктов, и при этом время его работы ограничено полиномом от времени работы алгоритма A.

Доказательство. Если текущий предикат (с учетом сделанной подстановки ρ) в вершине $y \in Y$ линеен хотя бы по двум переменным, то из

леммы 1.1 следует, что никакой из литералов, участвующих в формуле, соответствующей вершине y, не является чистым. Если же литерал является чистым, то для всех вершин, в которых он появляется, осталось менее двух линейных переменных. Все такие вершины обязаны содержаться в $I \in Cl^k(Vars(\rho))$, а значит, все соответствующие биты выхода уже открыты, и алгоритм самостоятельно может сделать подстановку, выполняющую чистые литералы. Аналогично, если формула содержит единичный дизъюнкт, то соответствующая ему вершина попадает в множество I, а это значит, что умный алгоритм сам может сделать правильную подстановку. □

Далее считаем, что умный алгоритм не использует правил упрощения.

Обозначим $N = \lfloor \frac{(c-k-1)r}{4dK} \rfloor$.

Лемма 2.7 ([22])**.** После N шагов умного близорукого алгоритма будет открыто не более чем $\frac{r}{2}$ выходных битов.

Доказательство. Количество открытых битов не более $K\frac{(c-k-1)r}{4dK} + |Cl^k(Vars(\rho))|$, где ρ — это текущая подстановка. По лемме 2.4 $|Cl^k(Vars(\rho)| \leq \frac{|Vars(\rho)|}{c-k-1}$. При этом $|Vars(\rho))| \leq \frac{(c-k-1)r}{4}$, тогда $K\frac{(c-k-1)r}{4dK} + |Cl^k(Z)| \leq \frac{(c-k-1)r}{4d} + \frac{r}{4} \leq \frac{r}{2}$ □

Следствие 2.3. В течение первых N шагов умный близорукий алгоритм не делает возвратов (возврат соответствует листу дерева расщеплений).

Доказательство. В течение N шагов количество открытых битов не превосходит $\frac{r}{2}$. По индукции можно доказать, что текущая подстановка в течение этого количества шагов является локально корректной. База индукции очевидна, а переход следует из того, что подстановка значения переменной выбирается так, чтобы сужение формулы на I было выполнено, это по лемме 2.5 возможно благодаря локальной корректности подстановки ρ на предыдущем шаге. □

Ближайшая цель — доказать, через N шагов с большой вероятностью текущая формула будет невыполнимой.

Далее мы считаем, что граф G имеет вид $G_L + H$, где G_L — это $(d-k)$-регулярный, а H — это k-регулярный R-граф, ранг матрицы смежности графа G_L не менее $n-1$. В этом случае функция Голдрейха f, построенная по G и P линейная по переменным $X \setminus R$.

Лемма 2.8. Пусть $b \in \{0,1\}^n$, а множество $J \subseteq X$. Пусть $y \in \{0,1\}^n$, а $Z \subseteq Y$, определим $X_y = \{x \in \{0,1\}^n \mid \forall j \in (X \setminus J)\ x_j = y_j\}$, $S_y = \{x \in X_y \mid f(x)|_Z = b|_Z\}$. Тогда либо $|S_y| \geq 2^{|J|-|Z|-|J\cap R|}$, либо $|S_y| = 0$.

Доказательство. Нам нужно оценить число подстановок $x \in X_y$, которые удовлетворяют системе уравнений $f(x)|_Z = b|_Z$. Если мы зафиксируем значения для переменных x_j при $j \in J \cap R$, то эта система станет линейной относительно переменных x_j при $j \in (J \setminus R)$. Ранг этой системы не превосходит $|Z|$, а количество переменных не менее $|J| - |J \cap R|$ (не обязательно, что все эти переменные явно входят в систему). Таким образом, если решения существуют, то размерность пространства решений не менее $|J| - |J \cap R| - |Z|$; поскольку система над полем $\mathbb{F}_2$, то число решений не меньше, чем $2^{|J|-|Z|-|J\cap R|}$ даже при фиксированных значениях x_j при $j \in J \cap R$. □

Пусть Z — это множество открытых битов b в уравнениии $f(x) = b$, ρ — некоторая частичная подстановка. Обозначим через $C_{\rho,Z,b}$ множество подстановок переменных $x \in \{0,1\}^n$, которые продолжают ρ и являются решением уравнения $f(x)|_Z = b|_Z$. Формально $C_{\rho,Z,y} = \{x \mid f(x)|_Z = b|_Z, x \sim \rho\}$.

Лемма 2.9. Пусть $Z \subseteq Y$, $|Z| < \frac{r}{2}$, $J \subseteq X$. Тогда для любых двух локально корректных подстановок ρ_1, ρ_2 с $Vars(\rho_1) = Vars(\rho_2) = J$ и для любого $b \in \{0,1\}^n$ выполняется $\frac{|C_{\rho_1,Z,b}|}{|C_{\rho_2,Z,b}|} \leq 2^{|R|}$.

Доказательство.

$$\frac{|C_{\rho_1,Z,b}|}{|C_{\rho_2,Z,b}|} = \frac{\sum\limits_{\sigma} |C_{\rho_1\cup\sigma,Z,b}|}{\sum\limits_{\sigma} |C_{\rho_2\cup\sigma,Z,b}|},$$

где суммирование в обоих случаях ведется по частичным подстановкам σ с $Vars(\sigma) = R \setminus J$.

Покажем, что размер множества $C_{\rho_i\cup\sigma,Z,b}$ либо 0, либо какой-то фиксированный, не зависящий от σ и $i \in \{1, 2\}$.

Размер множества $C_{\rho_i\cup\sigma,Z,b}$ равняется числу решений системы уравнений $f(x)|_Z = b|_Z$ при условии, что некоторые биты x зафиксированны подстановкой $\rho_i \cup \sigma$. При такой фиксации система становится линейной. Заметим, что ранг такой системы не зависит от подстановок ρ_i и σ (поскольку они влияют только на столбец правой части), поэтому если такая система имеет решение, то их число не зависит от i и σ.

Поскольку ρ_i локально корректна, а $|Z| < \frac{r}{2}$. Тогда по лемме 2.5 следует, что найдется такая подстановка σ_i с $Vars(\sigma_i) = R \setminus J$, что $C_{\rho_i\cup\sigma_i,Z,b} \neq \emptyset$. Тем самым

$$\frac{|C_{\rho_1,Z,b}|}{|C_{\rho_2,Z,b}|} \leq \frac{2^{|R|}|C_{\rho_1\cup\sigma_1,Z,b}|}{|C_{\rho_2\cup\sigma_2,Z,b}|} = 2^{|R|}.$$

□

Теорема 2.3. Пусть $|R| = o(\frac{n}{K})$, пусть ρ — это текущая подстановка после N шагов умного близорукого алгоритма, запущенного на формуле, которая кодирует $f(x) = b$ для некоторого $b \in f(\{0,1\}^n)$, а Z — это множество открытых битов. Тогда $\Pr_{y\leftarrow U(\{0,1\}^n)}[\exists x : x \sim \rho,\ f(x) = f(y) \mid f(y)|_Z = b|_Z] \leq 2^{-\Omega(\frac{n}{K})}$.

Доказательство. По следствию 2.3 на протяжении указанного числа шагов алгоритм не делал возвратов, следовательно $|\rho| = \frac{(c-k-1)r}{4dK}$.

Применим лемму 2.8 для $J = Vars(\rho)$, а $Z = I$, где $I \in Cl^k(Vars(\rho))$ из определения умного близорукого алгоритма после шага N. Поскольку $b \in f(\{0,1\}^n)$, то найдется $y \in \{0,1\}^n$, для которого $S_y \neq \emptyset$ (S_y определено в лемме 2.8), для этого y выполняется $|S_y| \geq 2^{|Vars(\rho)|-|I|-|R|}$.

Поэтому не менее $2^{|Vars(\rho)|-|I|-|R|}$ подстановок с носителем $Vars(\rho)$ являются локально корректными. Заметим, что

$$\begin{aligned}
&\Pr_{y\leftarrow U(\{0,1\}^n)}[\exists x : x \sim \rho,\ f(x) = f(y) \mid f(y)|_Z = b|_Z] \\
&\quad = \Pr_{y\leftarrow U(\{0,1\}^n)}[f^{-1}(f(y)) \cap C_{\rho,Z,b} \neq \emptyset \mid f(y)|_Z = b|_Z] \\
&\quad \leq \max_y |f^{-1}(f(y))| \cdot \Pr_{y\leftarrow U(\{0,1\}^n)}[y \in C_{\rho,Z,b} \mid f(y)|_Z = b|_Z]
\end{aligned}$$

По лемме 2.2 первый сомножитель оценивается $\max_y |f^{-1}(f(y))| \leq 2^{|R|+1}$. Оценим второй сомножитель: $\Pr_{y\leftarrow U(\{0,1\}^n)}[y \in C_{\rho,Z,b} \mid f(y)|_Z = b|_Z] \leq \frac{\max_\sigma |C_{\sigma,Z,b}|}{\sum_\sigma |C_{\sigma,Z,b}|}$, где σ пробегает все локально корректные подстановки с носителем $Vars(\rho)$. По лемме 2.9 $\frac{\max_\sigma |C_{\sigma,Z,b}|}{\sum_\sigma |C_{\sigma,Z,b}|} \leq 2^{|R|}\frac{\min_\sigma |C_{\sigma,Z,b}|}{2^{|Vars(\rho)|-|I|-|R|}\min_\sigma |C_{\sigma,Z,b}|} = 2^{2|R|+|I|-|Vars(\rho)|}$.

Итого:

$$\Pr_{y\leftarrow U(\{0,1\}^n)}[\exists x : x \sim \rho,\ f(x) = f(y) \mid f(y)|_Z = b|_Z] \leq 2^{3|R|+|I|-|Vars(\rho)|+1}.$$

Поскольку $I \in Cl^k(Vars(\rho))$, то из леммы 2.4 следует, что $|I| \leq (c-k-1)^{-1}|Vars(\rho)|$. Утверждение теоремы следует из того, что $Vars(\rho) = \Omega(\frac{n}{K})$ и $c > k+2$. □

Теорема 2.4. Пусть $|R| = o(\frac{n}{K})$, тогда для любого "близорукого" алгоритма A выполнено $\Pr_{y,s}[t_A(\Phi_{f(x)=f(y)}) \geq 2^{\Omega(n)}] \geq 1 - 2^{-\Omega(\frac{n}{K})}$, где $t_A(x)$ — время работы алгоритма A на входе x, а s — строка случайных битов, которые использует алгоритм A.

Доказательство. Из леммы 2.6 следует, что достаточно доказать теорему для умных близоруких алгоритмов, которые не используют правил упрощения.

Зафиксируем последовательность случайных битов s и докажем, что для алгоритмов с такой последовательностью случайных битов выполняется $\Pr_y[t_A(\Phi_{f(x)=f(y)}) \geq 2^{\Omega(n)}] \geq 1 - 2^{-\Omega(\frac{n}{K})}$. Из этого будет следовать теорема.

Рассмотрим умный близорукий алгоритм после N шагов на формуле $\Phi_{f(x)=f(y)}$. Пусть Z_y — это множество открытых битов правой части к этому моменту. Заметим, что для данной последовательности случайных битов s поведение алгоритма в первые N шагов будет одинаковым для всех $y' \in \{0,1\}^n$, для которых $f(y')|_{Z_y} = f(y)|_{Z_y}$ (при этом $Z_{y'} = Z_y$). Таким образом, множество всех $y \in \{0,1\}^n$ можно разбить на конечное число классов эквивалентности $S_1, S_2, \ldots, S_m$, так что для всех y из одного класса S_y совпадают значения Z_y и $f(y)|_{Z_y}$, а для представителей разных классов такие значения не совпадают.

$$\Pr_y[t_A(\Phi_{f(x)=f(y)}) \geq 2^{\Omega(n)}] = \sum_{i=1}^{m} \Pr_y[t_A(\Phi_{f(x)=f(y)}) \geq 2^{\Omega(n)} \mid y \in S_i] \Pr_y[y \in S_i].$$

По теореме 2.3 после N шагов умного близорукого алгоритма

$$\Pr_y[\Phi_{f(x)=f(y)}|_\rho \text{ невыполнима } \mid y \in S_i] \geq 1 - 2^{-\Omega(\frac{n}{K})},$$

где ρ — текущая подстановка, которая является локально корректнтной, тогда по теореме 2.2 $\Pr_y[t_A(\Phi_{f(x)=f(y)}) \geq 2^{\Omega(n)} \mid y \in S_i] \geq 1 - 2^{-\Omega(\frac{n}{K})}$. Из последнего неравенства следует теорема. □

Приведем конструкцию функции Голдрейха, на которой достигается доказанная нижняя оценка.

Выберем $\epsilon = \frac{1}{4k}$ и построим для такого ϵ экспандер H на n вершинах по лемме 1.4. Константа d будет удовлетворять неравенству $d \geq 4k$. Добавим к построенному графу 1-регулярный граф T, чтобы получился граф $H + T$ с рангом матрицы смежности $n - 1$. Полученный граф будет $(r, d+1, (1-\frac{1}{4k})d)$-экспандером. Выберем подмножество $R \subseteq X$ размера $o(n/K)$ и k-регулярный R-граф F. Положим $G = H + T + F$, построенный граф будет $(r, d+1+k, (1-\frac{1}{4k})d)$-экспандером, или граничным $(r, d+1+k, d(1-\frac{1}{4k}) - k - 1)$-экспандером. При $k > 1$ выполняется $d(1-\frac{1}{4k}) - k - 1 > k + 2$. Для такого графа подойдет любой предикат вида $x_1 + \cdots + x_{d-k} + Q(x_{d_{k+1}}, \ldots, x_d)$, где Q — произвольный k-местный предикат. Можно заметить, что нигде не используется, что предикат Q одинаковый для всех вершин множества Y.

Глава 3

Нижние оценки на близорукие DPLL-алгоритмы с эвристикой отсечения

В данной главе мы расширим класс DPLL-алгоритмов, добавив эвристику отсечения, которая может решить, что ветвь дерева расщепления “бесперспективная”, и не стоит ее просматривать. Точнее, перед каждым рекурсивным вызовом алгоритм вызывает эвристику $\mathbf{C}$, в зависимости от результата которой вызов выполняется или нет.

Мы покажем, что возможно построить за полиномиальное время семейство таких невыполнимых формул $\Phi^{(n)}$, что для любых детерминированных эвристик $\mathbf{A}$ и $\mathbf{C}$ найдется такой полиномиально моделируемый ансамбль распределений R_n, что DPLL-алгоритм, основанный на эвристиках $\mathbf{A}, \mathbf{B}$ и $\mathbf{C}$ для некоторой эвристики $\mathbf{B}$ либо ошибается на 99% формул, сгенерированных согласно распределению R_n, либо работает экспоненциальное время на формуле $\Phi^{(n)}$. В случае, если $\mathbf{A}$ и $\mathbf{C}$ не ограничены, мы покажем, что подобное утверждение эквивалентно неравенству $\mathbf{P} \neq \mathbf{NP}$.

Для доказательства основного результата мы будем использовать технику замыканий для экспандеров, предложенную Алехновичем. Мы

представим конструктивный вариант замыкания и применим его для построения ансамбля распределений R_n.

Мы также дадим похожие оценки для выполнимых формул, однако в этом случае невозможно доказать в точности такой же результат, так как близорукий алгоритм может подставлять значения переменным, соответствующие выполняющему набору конкретной формулы Φ. Тем не менее, в выполнимом случае возможно построить семейство формул $\Phi^{(n)}$, зависящее от эвристик $\mathbf{A}, \mathbf{B}$, в предположениях, что $\mathbf{B}$ близорука.

Мы построим распределение R_n зависящее от эвристик $\mathbf{A}$ и $\mathbf{C}$. Мы также построим универсальный ансамбль распределений Q_n, который доминирует все распределения R_N. В частности, мы докажем, что близорукий DPLL-алгоритм с эвристикой отсечения, ошибающийся с вероятностью $o(1)$ на случайной формуле из ансамбля Q_n, работает экспоненциальное время на формуле $\Phi^{(n)}$.

Результаты данной главы опубликованы в работе [25].

3.1 DPLL-алгоритмы с эвристикой отсечения

Определим семейство алгоритмов для решения задачи выполнимости формулы в КНФ. DPLL-алгоритмы с эвристикой отсечения параметризуются тремя эвристиками: $\mathbf{A}, \mathbf{B}$ и $\mathbf{C}$. Эвристика $\mathbf{A}$ получает на вход формулу и возвращает переменную для расщепления, эвристика $\mathbf{B}$ получает на вход формулу и переменную и возвращает значение, которое должно быть подставлено на место переменной первым, эвристика $\mathbf{C}$ получает формулу и переменную и значение и определяет, стоит ли осуществлять данную подстановку.

Формально алгоритм $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ определяется следующим образом.

Алгоритм 3.1. Алгоритм $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$.

- Вход: формула φ
- Если φ не содержит дизъюнктов, то вернуть 1.

- Если φ содержит пустой, то вернуть 0.
- $j := \mathbf{A}(\varphi)$.
- $b := \mathbf{B}(\varphi, j)$.
- Если $\mathbf{C}(\varphi, j, b) = 1$, то если $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\varphi[x_j := b]) = 1$, вернуть 1.
- Если $\mathbf{C}(\varphi, x, \neg b) = 1$, то если $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\varphi[x_j := \neg b]) = 1$, вернуть 1.
- Вернуть 0.

Пусть $\mathbf{1}$ обозначает константную единичную функцию. Алгоритм $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{1}}$ очевидно корректно решает задачу выполнимости. Для любых эвристик $\mathbf{A}, \mathbf{B}, \mathbf{C}$ алгоритм $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{1}}$ возвращает корректное значение на невыполнимой формуле, однако на выполнимой может ошибиться.

Для любой формулы φ выполнение алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ соответствует дереву расщепления. Вершины дерева расщепления соответствуют рекурсивным вызовам, ребра соответствуют подстановкам значения переменной. Каждый лист дерева расщепления соответствует одной из трех ситуаций: 1) один из дизъюнктов стал невыполнимым, 2) подстановка выполнила все дизъюнкты, 3) процедура $\mathbf{C}$ вернула 0 дважды. Несложно понять, что если эвристики $\mathbf{A}$ и $\mathbf{B}$ детерминированы, то дерево расщепления алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ является поддеревом дерева расщепления алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{1}}$, если на вход подана невыполнимая формула, то достаточно детерминированности только эвристики $\mathbf{A}$.

Под временем работы алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ будем подразумевать количество рекурсивных вызовов (количество вершин в дереве расщепления).

Наша главная цель — доказательство нижней оценки для почти корректных DPLL-алгоритмов с эвристикой отсечения. Мы построим полиномиально моделируемый ансамбль распределений R_n на выполнимых формулах и такую последовательность невыполнимых формул $\Phi^{(n)}$, что если $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ с большой вероятностью корректно решает задачу выполнимости на формулах из распределения R_n, то алгоритм $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ будет

работать экспоненциальное время на формуле $\Phi^{(n)}$. Без ограничений на эвристики подобный результат влек бы неравенство $\mathbf{P} \neq \mathbf{NP}$, так как в противном случае полиномиальная эвристика $\mathbf{C}$ могла бы отсекать невыполнимые ветви. Мы рассмотрим близорукие ограничения.

Близорукая эвристика имеет доступ к формуле со стертыми знаками отрицания. Для каждой переменной она также имеет доступ к количеству вхождений данной переменной с положительным и отрицательным знаком. Близорукая эвристика также может прочесть $K = n^{1-\varepsilon}$ дизъюнктов целиком (со знаками отрицаний).

Однако недостаточно ограничить только эвристику $\mathbf{C}$, так как эвристика $\mathbf{A}$ может передать информацию эвристике $\mathbf{C}$. Например, $\mathbf{A}$ может возвращать лексикографически минимальные переменные для выполнимых формул и максимальные для невыполнимых. Мы также сделаем эвристику $\mathbf{A}$ близорукой. Нам также понадобится детерминированность эвристик $\mathbf{A}, \mathbf{C}$.

3.2 Система близоруких копий

Далее будем предполагать, что эвристики $\mathbf{A}$ и $\mathbf{C}$ являются детерминированными и полиномиальными по времени близорукими алгоритмами.

Пусть Φ — невыполнимая формула, и T_Φ — дерево расщепления для $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{1}}(\Phi)$. Скажем, что множество вершин $\{v_1, v_2, \ldots, v_S\}$ дерева расщепления образуют *систему близоруких копий* формулы Φ, если выполняются следующие условия:

- для любых $i, j \in \{1, \ldots, S\}$ вершина v_i не является потомком вершины v_j;
- для каждой вершины v_i существует выполнимая формула Φ_i с одним выполняющим набором, и данный набор удовлетворяет частичной подстановке, сделанной на пути от корня дерева T_Φ к вершине v_i;
- близорукие алгоритмы $\mathbf{A}$ и $\mathbf{C}$ не могут различить формулы Φ_i и Φ на пути от корня дерева T_Φ к вершине v_i. Формально это озна-

чает, что для любой подстановки ρ, соответствующей пути в дереве расщеплений для формулы Φ, нет разницы с формулой Φ_i, то есть эти формулы со стертыми знаками отрицания одинаковы, каждая переменная имеет одинаковое число вхождений с положительным и отрицательным знаком, и на каждом шаге эвристики **A** и **C** видят одинаковые знаки отрицаний в прочитанных дизъюнктах для формул Φ_i и Φ.

Назовем формулы $\{\Phi_1, \Phi_2, \ldots, \Phi_S\}$ *близорукими представителями* формулы Φ.

Лемма 3.1. Пусть Φ — невыполнимая формула, и $\{v_1, v_2, \ldots, v_S\}$ — система близоруких копий формулы Φ с близорукими представителями $\{\Phi_1, \Phi_2, \ldots, \Phi_S\}$. Пусть U_Φ — равномерное распределение на формулах $\{\Phi_1, \Phi_2, \ldots, \Phi_S\}$. Предположим, что $\Pr_{\phi \leftarrow U_\Phi}[\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\phi) = 1] \geq 1-\epsilon$, тогда время работы $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\Phi)$ не менее $(1-\epsilon)S$. (Вероятность берется также и по случайным битам эвристики **B**).

Доказательство. Заметим, что использование случайных битов эвристикой **B** не оказывает влияние на время работы алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ на невыполнимой формуле. Зафиксируем последовательность случайных битов, для которой выполняется $\Pr_{\phi \leftarrow U_\Phi}[\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\phi) = 1] \geq 1 - \epsilon$. Теперь предполагаем, что B использует только эту последовательность случайных битов.

Рассмотрим такую из формул Φ_j, что $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\Phi_j) = 1$. Заметим, что дерево расщепления алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ на входе Φ_j содержит путь от корня до вершины v_j из дерева T_Φ. При этом единственный выполняющий набор формулы Φ_j является продолжением частичной подстановки, сделанной на этом пути. На протяжении этого пути эвристика **A** выбирает те же переменные для расщепления что и на формуле Φ, так как не может различить формулы Φ и Φ_j. Также эвристика **C** не может различить данные формулы на протяжении указанного пути. Следовательно, не менее $(1-\epsilon)S$ вершин из множества $\{v_1, \ldots, v_S\}$ должны быть посещены алгоритмом $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ на входе Φ. □

Итак, теперь наша цель — построение семейства формул $\Phi^{(n)}$, для каждой формулы из которого найдется система близоруких копий экспоненциального размера. И более того, равномерное распределение на близоруких представителях должно быть полиномиально моделируемым.

Рассмотрим формулы, которые кодируют систему линейных уравнений $Ax = b$ над полем $\mathbb{F}_2$, где A — матрица $n \times n$.

Нам потребуется, чтобы в каждой строке матрицы A число единиц было ограничено сверху константой d. В случае линейных систем это означает, что система содержит уравнения вида $x_{i_1} + x_{i_2} + \cdots + x_{i_s} = b_i$, где $s \leq d$. Такие уравнения могут быть записаны в виде d-КНФ формулы, содержащей не более 2^d дизъюнктов. Заметим, что в зависимости от правой части уравнения в записи в КНФ меняются только знаки отрицаний, а количество вхождений каждой переменной с положительным и отрицательным знаками одинаково. Из сказанного следует, что близорукая эвристика не может отличить 0 и 1 в правой части уравнения, пока не прочтет один из дизъюнктов со знаками отрицания. Будем говорить, что близорукая эвристика открывает биты вектора b и за один запрос она может открыть один бит.

Выберем невырожденную матрицу A, система $Ax = b$ имеет единственное решение. Это означает, что переменная x_1 не может принять значение $\alpha \in \{0, 1\}$. Формула Φ будет кодировать систему $Ax = b$ после подстановки $x_1 := \alpha$. Φ невыполнима по выбору α.

Теперь опишем неформально, как мы будем выбирать систему близоруких копий для формулы Φ. Для вершины v_i из дерева расщеплений алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{1}}$ мы найдем такое $b' \in \{0, 1\}^n$, что решение системы будет согласовано с частичной подстановкой, сделанной на пути от корня к v_i, и с подстановкой $x_1 := \alpha$. Неочевидно, почему такой выбор всегда возможен, однако мы сможем построить указанную систему при помощи выбора матрицы A.

3.3 Граничный экспандер и линейное замыкание

В данном разделе мы рассмотрим упрощенную версию замыкания для случая $k = 0$ из раздела 2.3. Данное упрощение позволит нам конструктивно искать надмножество замыкания небольшого размера (лемма 3.5)

Рассмотрим двудольный граф G (с кратными ребрами), обозначим доли как X и Y. В данной главе разрешим (r, d, c)-(граничным) экспандерам иметь степени вершин из множества Y не более d (а не ровно d).

Пусть G — (r, d, c)-граничный экспандер. Пусть $J \subseteq X$, тогда *замыканием* множества J назовем максимальное подмножество $I \subseteq Y$, которое удовлетворяет следующим условиям: 1) $|I| \leq r$; 2) $\delta(I) \subseteq J$. Замыкание может быть определено неоднозначно, будем подразумевать, что берется лексикографически наименьшее из замыканий. Будем использовать обозначение $Cl(J)$ для данного замыкания.

Предложение 3.1. Пусть G — (r, d, c)-граничный экспандер и $c \geq 1$, тогда выполняются следующие свойства: 1) Если I — замыкание множества J, тогда $|I| \leq \frac{|J|}{c}$. 2) Если $|J| < r/2$, то существует единственное замыкание множества J. 3) Пусть $J \subseteq J'$ и $|J'| < r/2$, то $Cl(J) \subseteq Cl(J')$.

Доказательство. 1) Так как $|I| \leq r$ и $\delta(I) \subseteq J$, получаем $|J| \geq \delta(I) \geq c|I|$. 2) Пусть I_1, I_2 — два замыкания множества J. По первому пункту получаем $|I_j| \leq \frac{|J|}{c} < r/2$. Несложно видеть $\delta(I_1 \cup I_2) \subseteq \delta(I_1) \cup \delta(I_2) \subseteq J$ и $|I_1 \cup I_2| \leq r$. Из максимальности замыкания получаем $I_1 = I_2$. 3) Так как $|J| \leq |J'| < r/2$, существуют единственные замыкания для J и J' по пункту 2. Рассмотрим множество $Cl(J) \cup Cl(J')$, очевидно что $\delta(Cl(J) \cup Cl(J')) \subseteq J'$, однако по пункту 1 $|Cl(J) \cup Cl(J')| \leq 2|J'|/c < r$. Следовательно, $Cl(J') = Cl(J) \cup Cl(J')$. □

Для любого двудольного графа G с частями X и Y определим матрицу A размера $|Y| \times |X|$. Элементы X соответствуют столбцам, а элементы Y — строкам. Обозначим $A_{y,x}$ количество ребер между x и y

по модулю 2. Пусть G — (r, d, c)-граничный экспандер и $c \geq 1$. Будем называть матрицу A матрицей смежности графа G. Рассмотрим линейную систему $Ax = b$, где x вектор неизвестных размера $|X|$. Пусть ρ — частичная подстановка переменных. Частичная подстановка ρ называется *локально корректной*, если $|\rho| < r/2$ и система уравнений $Ax|_{Cl(vars(\rho))} = b|_{Cl(vars(\rho))}$ имеет решение согласованное с ρ. Будем обозначать проекцию вектора z на координаты из множества E как $z|_E$.

Лемма 3.2 ([22])**.** Если частичная подстановка ρ локально корректна, то для любого $I \subseteq Y, |I| \leq r/2$ система $Ax|_I = b|_I$ имеет решение, согласованное с ρ.

Доказательство. Рассмотрим такое минимальное множество I, $|I| \leq r/2$, что условие леммы не выполнено. Допустим, что $\delta(I) \not\subseteq vars(\rho)$, пусть $y \in I$ такое, что для некоторой вершины $z \in X \setminus vars(\rho)$ существует точно одно ребро из z в I и оно соединяет z с y. Из минимальности I получаем, что $I' = I \setminus \{y\}$ удовлетворяет утверждению леммы, т.е. найдется такой $x \in \{0,1\}^{|X|}$, что x согласован с ρ и $Ax|_{I'} = b|_{I'}$. Но мы можем удовлетворить уравнение, соответствующее вершине y (возможно, мы должны изменить значение переменной z в подстановке x), противоречие. Итак, $\delta(I) \subseteq vars(\rho)$, таким образом $\delta(I \cup Cl(vars(\rho))) \subseteq vars(\rho)$ и $|I \cup Cl(vars(\rho))| \leq r$; следовательно, по пункту 1 утверждения 3.1 $|Cl(vars(\rho))| \leq |\rho| < r/2$. Максимальность $Cl(vars(\rho))$ влечет $I \subseteq Cl(vars(\rho))$. □

3.4 Очищенное дерева поиска

Следующую лемму мы докажем в разделе 3.5.

Лемма 3.3. Существует алгоритм, который на вход получает $n \in \mathbb{N}$ и за полиномиальное время возвращает (r, d, c)-граничный экспандер G с n вершинами в каждой доле и невырожденной матрицей смежности, где $r = \Omega(n)$, $c > 2$, d — константа, и степень всех вершин из доли X ограничена константой D.

Пусть G — граф из леммы 3.3, и A — его матрица смежности. Пусть $x_1 := \alpha$ — подстановка, которая делает систему $Ax = b$ невыполнимой. Пусть формула Φ кодирует систему $Ax = b$ после применения подстановки $x_1 := \alpha$.

Рассмотрим дерево расщепления T_Φ, которое создает алгоритм $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{1}}$ на входе Φ. Пусть каждая вершина T_Φ содержит частичную подстановку, содержащую подстановки, которые делал алгоритм на пути от корня к данной вершине, и подстановку $x_1 := \alpha$.

Обойдем вершины дерева T_Φ в порядке увеличения глубины. Некоторые вершины будем удалять из дерева вместе с поддеревьями. Если вершина удалена, то больше ее не рассматриваем. Удалять вершину будем в том и только в том случае, если частичная подстановка, соответствующая данной вершине, не была локально корректной. Получившееся дерево будет обозначать T'_Φ.

Лемма 3.4. 1. Длина каждого пути от корня до листа в T'_Φ составляет не менее $r/2 - 2$.

2. По крайней мере половина из первых $r/2 - 3$ вершин на каждом пути от корня до листа T'_Φ являются точками расщепления (имеют ровно двух потомков).

Доказательство. 1) Из того, что $c > 2$, следует, что $Cl(\{x_1\}) = \emptyset$ и подстановка $\{x_1 := \alpha\}$ локально корректна. Пусть ρ — локально корректная подстановка соответствующая вершине v в дереве T_Φ. Пусть x_v — переменная для расщепления в вершине v, и $|\rho| < r/2 - 1$; пункт 1 из утверждения 3.1 влечет неравенство $Cl(vars(\rho) \cup \{x_v\}) < r/2$. Из того, что ρ локально корректна, следует существование такого α_v, что $\rho \cup \{x_v := \alpha_v\}$ — также локально корректная подстановка. В завершение отметим, что лист дерева T_Φ не может соответствовать локально корректной подстановке, т.к. формула Φ невыполнима.

2) Пусть v — вершина в дереве T'_Φ, которая находится на расстоянии от корня не более $r/2 - 3$, и u — вершина, которая находится на

пути от корня до листа, содержащего вершину v, и длина пути от корня до u равна $\lceil r/2 \rceil - 1$. Пусть $|\rho_v| \leq r/2 - 2$, и ρ_v локально корректна. Пусть $\rho'_v = \rho_v \cup \{x_v := \alpha_v\}$ не является локально корректной частичной подстановкой. Последнее означает, что значение x_v следует из подстановки ρ_v и уравнений, соответствующих $Cl(vars(\rho'_v))$. Третий пункт утверждения 3.1 обеспечивает тот факт, что $Cl(vars(\rho'_v)) \subseteq Cl(vars(\rho_u))$. Представим множество $vars(\rho_u)$ в виде дизъюнктного объединения двух множеств $P \cup Q$, где P соответствует переменным в вершинах с двумя потомками, а Q — соответствует остальным переменным. Значения переменных из множества Q следуют из значений для переменных из P и уравнений, соответствующих множеству $Cl(vars(\rho_u))$. Заметим, что уравнения соответствующие частичной подстановке ρ_u линейно независимы. Таким образом, $|Q| \leq |Cl(vars(\rho_u))| \leq \frac{|vars(\rho_u)|}{c} < \frac{|vars(\rho_u)|}{2}$. □

Следствие 3.1. Размер дерева T'_Φ (и дерева T_Φ) составляет не менее $2^{r/4-2}$.

Пусть K — верхняя оценка на количество битов, которое эвристики **A** и **C** могут открыть за один шаг. Построим систему близоруких копий для формулы Φ. Рассмотрим дерево T'_Φ, на каждом пути от корня к листу выберем такую вершину, что среди ее предков ровно N точек ветвления, где $N \leq \frac{r/4-2}{K}$. Лемма 3.4 доказывает, что расстояние до корня от выбранных вершин не более $2N$, и количество выбранных вершин равно 2^N. Обозначим множество выбранных вершин как $\{v_1, v_2, \ldots, v_{2^N}\}$. Рассмотрим работу алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ на пути от корня к вершине v_i. На протяжении этого пути эвристики **A** и **C** открывают не более $2NK$ битов вектора b. Таким образом, $2NK < r/2$ и подстановка ρ_{v_i} является локально корректной, следовательно, существует такой вектор b', что система $Ax = b'$ имеет решение, согласованное с ρ_{v_i}, b и b' одинаковы на всех открытых битах на пути. Пусть Φ_i — формула, которая кодирует линейную систему $Ax = b'$ после подстановки $x_1 := \alpha$.

Необходимо доказать, что равномерное распределение на формулах Φ_i является полиномиально моделируемым. Пусть существует алгоритм,

который по вершине дерева T_Φ определяет количество потомков у нее в дереве T'_Φ. В этом предположении мы можем генерировать равномерное распределение на множестве $\{v_1, v_2, \ldots, v_{2^N}\}$. Точнее, мы моделируем работу алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{1}}$ в следующем смысле: эвристика **A** выбирает переменную для расщепления. Если текущая вершина является точкой расщепления, то мы выбираем случайное значение для подстановки; иначе выбираем то единственное, которое обеспечит корректность локальной подстановки. Останавливаемся после N точек ветвления. Пусть мы остановились на вершине v_i. К этому моменту мы знаем биты, открытые эвристиками **A** и **C**. Основываясь на этой информации нетрудно восстановить формулу Φ_i при помощи метода Гаусса.

Теперь опишем способ определения по вершине, является ли она точкой расщепления, и если не является — то способ нахождения правильной подстановки. К сожалению, нет очевидного полиномиального по времени алгоритма для подсчета замыкания. Другая идея — нахождение минимально невыполнимой подсистемы, однако данная задача **NP**-трудна и лучшее известное приближенное решение ([49]) недостаточно для решения поставленной задачи.

Однако лемма 3.2 обеспечивает, что подстановка локально корректна, если совместно любое множество уравнений размера не более $r/2$. Если подстановка не является локально корректной, то противоречивым является множество уравнений, содержащее замыкание. Мы построим за полиномиальное время надмножество замыкания размера не более $r/2$. Для того, чтобы сделать это, мы покажем, что замыкание множества J содержится в окрестности множества соседей множества J. И данная окрестность будет искомым надмножеством.

Пусть $\Gamma^k(J)$ — множество вершин, удаленных от J на расстояние не более k. Следующая лемма доказана Ицыксоном, для полноты изложения приведем ее доказательство.

Лемма 3.5 ([25])**.** Если $|J| < r/2$, то $Cl(J) \subseteq \Gamma^k(J)$, где $k = O(\log |J|)$.

Доказательство. Пусть $I = Cl(J)$. Разделим множество I на непересе-

кающиеся части $I_1 = I \cap \Gamma^1(J)$, $I_{j+1} = \left(I \setminus \bigcup_{i=1}^{j} I_i\right) \cap \Gamma^{2j+1}(J)$; пусть I_ℓ — последнее не пустое множество. Введем обозначение $S_i = \cup_{j=i}^{\ell} I_j$. Утверждение 3.1 влечет тот факт, что $|S_i| \leq |I| < r/2$. Так как $\delta(I) \subseteq J$, то для $i \geq 2$ выполнено свойство: $\delta(S_i) \subseteq \Gamma(I_{i-1})$. Поэтому $c|S_i| \leq d|I_{i-1}|$. Последнее неравенство обеспечивает, что $|S_i|\frac{c}{d} \leq |I_{i-1}|$, поэтому $|S_i|(1 + \frac{c}{d}) \leq |S_{i-1}|$. И $1 \leq |I_\ell| = |S_\ell| \leq \frac{|J|}{(1+\frac{c}{d})^{\ell-1}}$, поэтому $\ell \leq \log_{1+c/d} |J| + 1$. □

Из леммы 3.3 степени всех вершин ограничены константой, отсюда вытекает следствие.

Следствие 3.2. Существует такое $\delta > 0$, что если $|J| < n^\delta$, то неравенство $|\Gamma^k(J)| < r/2$ выполнено.

Из вышесказанного вытекает следующая лемма.

Лемма 3.6. Существует такой полиномиальный по времени алгоритм, который по $n \in \mathbb{N}$ возвращает невыполнимую формулу Φ; существует такая константа δ, что для любого близорукого алгоритма с полиномиальными детерминированными эвристиками $\mathbf{A}$ и $\mathbf{C}$ найдется система близоруких копий для Φ размера 2^N с близорукими представителями $\Phi_1, \Phi_2, \ldots, \Phi_{2^N}$ в дереве расщепления алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{1}}$ для любой эвристики $\mathbf{B}$, где $N = min\{n^\delta, \frac{r/4-2}{K}\}$ и K — количество битов, которые могут открыть эвристики $\mathbf{A}$ и $\mathbf{C}$ за один шаг. Более того, равномерное распределение на множестве $\{\Phi_1, \Phi_2, \ldots, \Phi_{2^N}\}$ является полиномиально моделируемым.

Лемма 3.1 и лемма 3.6 доказывают следующую теорему:

Теорема 3.1. Существуют такой полиномиальный алгоритм, который выдает по n невыполнимую формулу $\Phi^{(n)}$, и такая константа $\delta > 0$, что для любого близорукого алгоритма с полиномиальными эвристиками $\mathbf{A}$ и $\mathbf{C}$ найдется такой полиномиально моделируемый ансамбль распределений R_n на выполнимых формулах, что если для некоторой эвристики

B и некоторого $\epsilon > 0$ неравенство $\Pr_{\phi \leftarrow R_n}[\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\phi) = 1] \geq 1 - \epsilon$ выполнено, то время работы алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\Phi^{(n)})$ не менее $(1-\epsilon)2^N$, где $N = min\{n^\delta, r/K\}$ и $r = \Omega(n)$.

3.4.1 Универсальное распределение

Основываясь на том факте, что формула $\Phi^{(n)}$ не зависит от эвристик **A** и **C**, мы построим универсальное распределение, которое доминирует остальные. Пусть $(\mathcal{R}^{(i)}, c_i)$ — нумерация сэмплеров $\mathcal{R}^{(i)}$, ограниченных по времени полиномом n^{c_i}, для всех распределений из теоремы 3.1 для всех DPLL-алгоритмов, основанных на близоруких полиномиальных эвристиках **A** и **C**. Мы построим сэмплер $\mathcal{Q}$, который определяет ансамбль распределений Q_n. $\mathcal{Q}$ на входе 1^n с вероятностью $\frac{1}{2^i}$ возвращает ответ сэмплера $\mathcal{R}^{(i)}(1^{\lfloor n^{1/c_i} \rfloor})$, запущенного на не более, чем n^{c_i} шагов, для всех $i < n$, и с вероятностью $\frac{1}{2^n}$ выдается фиксированная выполнимая формула ϕ_0. Следующая теорема доказана Ицыксоном, для полноты изложения приведем ее доказательство.

Теорема 3.2 ([25])**.** Для любых полиномиальных эвристик **A** и **C** существует такое $\epsilon > 0$, что если для любой эвристики **B** алгоритм $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ на всех n ошибается с вероятностью не более ϵ на входах, распределенных согласно Q, то для достаточно больших n время работы алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ на входе $\Phi^{(n)}$ будет не менее 2^{N-1}, где $N = min\{n^\delta, r/K\}$ и $r = \Omega(n)$, δ — некоторая положительная константа.

Доказательство. Пусть $\mathcal{R}_i$ — сэмплер, соответствующий эвристикам **A** и **C**, и его время работы ограничено полиномом n^{c_i}. Для всех $n \geq i$ сэмплер $\mathcal{R}^{(i)}$ будет представлен в нумерации из определения ансамбля $\mathcal{Q}$. Пусть $\epsilon = \frac{1}{2^{i+1}}$; для $i \geq n$ алгоритм $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ работает корректно на случайном входе с вероятностью не менее $\frac{1}{2}$ согласно распределению $R^{(i)}_{\lfloor n^{1/c_i} \rfloor}$. Пусть $m = \lfloor n^{1/c_i} \rfloor$, тогда алгоритм $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ работает на входе Φ_m не менее 2^{N-1} шагов, где $N = min\{m^\delta, r/K(m)\}$ и $r = \Omega(m)$, $\delta > 0$ — константа из теоремы 3.1. □

3.4.2 Нижняя оценка на выполнимых формулах

Если сделать эвристику **B** близорукой, то можно доказать результат, аналогичный теореме 3.1, но для выполнимых формул $\Phi^{(n)}$.

Теорема 3.3. Существует такое $\delta > 0$, что для любых близоруких эвристик **A**, **B** и **C** найдется полиномиальный алгоритм, который по n выдает такую выполнимую формулу $\Psi^{(n)}$, что существует такой полиномиально моделируемый ансамбль распределений R_n на выполнимых формулах, что если для некоторого $\epsilon > 0$ выполнено неравенство $\Pr_{\phi \leftarrow R_n}[\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\phi) = 1] \geq 1 - \epsilon$, то время работы алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\Psi^{(n)})$ не менее $(1-\epsilon)2^N$, где $N = min\{n^\delta, r/K\}$ и $r = \Omega(n)$.

Доказательство. Рассмотрим матрицу A, которая использовалась для конструкции формулы $\Phi^{(n)}$ в доказательстве теоремы 3.1. Пусть $b_0 = 0^n$; запустим алгоритм $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ на формуле $Ax = b_0$. Рассмотрим первую подстановку, которую делает алгоритм $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$. Не умаляя общности можно полагать, что первая подстановка была $x_1 := \alpha_1$, и алгоритм открыл биты из множества Z_1. Выберем формулу $\Psi^{(n)}$ как выполнимую формулу вида $Ax = b$, где все биты b из множества Z_1 равны нулю, и формула после подстановки $x_1 := \alpha_1$ становится невыполнимой. Подстановка $x_1 := \alpha_1$, очевидно, локально корректная, так как $c > 2$, таким образом, такая выполнимая формула $\Psi^{(n)}$ существует.

Теперь возьмем распределение R_n из теоремы 3.1 для формулы $\Phi^{(n)}$, которая соответствует системе $Ax = b$ с подстановкой $x_1 := \alpha_1$. В завершение заметим, что время работы алгоритма $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ на входе $\Psi^{(n)}$ не менее времени работы данного алгоритма на входе $\Phi^{(n)}$, так как первая подстановка алгоритма — это $x_1 := \alpha_1$. □

3.5 Конструкция экспандера

В данном разделе мы построим явную конструкцию (r, d, c)-граничного экспандера с n вершинами в каждой доле, где $r = \Omega(n)$, $c > 2$, и следующими дополнительными свойствами:

- степени всех вершин ограничены константами;
- матрица смежности графа невырождена.

Назовем семейство графов G_n явным, если существует алгоритм, который по n выдает граф G_n за полиномиальное от n время.

Теорема 3.4. [33] Для любого $\epsilon > 0$ найдутся $k \geq 1$ и явная конструкция $(r, d, (1-\epsilon)d)$-экспандера с размерами долей $|X| = m$ и $|Y| = n$, где $r = \frac{n}{kd}$, $d = polylog(\frac{n}{m})$.

Следствие 3.3. Для любого достаточно большого d и любого n существует явная конструкция $(\frac{n}{kd}, d, 0.95d)$-экспандера с размерами долей $|X| = n$, $|Y| = 2n$, где — k некоторая константа.

Лемма 3.7. Для любого достаточно большого d и любого n существует явная конструкция $(r, d, 0.75d)$-экспандера с размерами долей $|X| = |Y| = n$, $r = \Omega(n)$ и со степенью вершин из доли X не более $20kd$.

Доказательство. Построим граф G по следствию 3.3. Найдем множество вершин $T \subseteq X$ со степенями большими $20kd$. Поскольку всего ребер не более $2nd$, получим, что $|T| \leq \frac{2nd}{20kd} = \frac{n}{10k}$. Если $|T| < \lfloor \frac{n}{10k} \rfloor$, добавим в T несколько вершин, чтобы было верно равенство $|T| = \lfloor \frac{n}{10k} \rfloor$. Пусть Z — множество вершин из доли Y, которые соединены с множеством T по крайней мере $\frac{d}{5}$ ребрами. Пусть $|Z| \geq \frac{n}{kd}$ и $Z' \subseteq Z$, $|Z'| = \lfloor \frac{n}{kd} \rfloor$. Существует по крайней мере $\frac{1}{5}|Z'|d$ ребер, соединенных с Z' и T. Следовательно, $|\Gamma(Z')| \leq |T| + \frac{4}{5}d|Z'| \leq 0.9\frac{n}{k} < 0.95d|Z'|$ для достаточно больших n; это противоречит свойству расширения. Таким образом, для достаточно больших n мы получаем, что $|Z| < \frac{n}{kd}$. Удалим из графа множества T и Z и несколько вершин из Y, чтобы сделать размеры X и Y равными $\lceil n(1 - 1/kd) \rceil$. Поскольку мы удалили несколько ребер, смежных с вершинами из Y, получающийся граф будет $(\frac{n}{kd}, d, 0.75d)$-экспандером. □

Лемма 3.8. Для любого достаточно большого d и любого n существует явная конструкция $(r, d+2, 0.75d)$-экспандера с размерами долей $|X| =$

$|Y| = n$, $r = \Omega(n)$, степени всех вершин из X не более $20kd+2$, и матрица смежности графа G невырождена над полем $\mathbb{F}_2$.

Доказательство. Для начала докажем вспомогательное утверждение

Предложение 3.2. Пусть $a = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \{0,1\}^n$ и $\sum_{i=1}^{n} \alpha_i = 0 \bmod 2$. Тогда вектора $b_1 = a + (1, 0, \ldots, 0), b_2 = a + (0, 1, 0, \ldots, 0), \ldots, b_n = a + (0, \ldots, 0, 1)$ линейно независимы над $\mathbb{F}_2$.

Доказательство. Сумма элементов любого вектора b_i нечетна, таким образом сумма нечетного числа b_i не может быть равна нулю. Пусть e_i — i-ый вектор стандартного базиса (единица только в позиции i). $b_{i_1} + \cdots + b_{i_{2\ell}} = 2\ell \cdot a + e_{i_1} + \cdots + e_{i_{2\ell}} = e_{i_1} + \cdots + e_{i_{2\ell}} \neq 0$, таким образом, сумма базисных векторов не может быть равна нулю. □

Построим граф G по лемме 3.8. Рассмотрим некоторое паросочетание между вершинами из X и Y (необязательно чтобы сопоставленные вершины были соединены ребром). Для каждой вершины из Y с нечетной степенью добавим ребро в ее пару. Заметим, что мы увеличили степень вершин не более, чем на 1, и степени всех вершин из Y стали четными.

Рассмотрим матрицу смежности получившегося двудольного графа. Пусть k — ранг данной матрицы, и $k < n$. Рассмотрим k линейно независимых строк данной матрицы. Будем выполнять следующую операцию для оставшихся $n - k$ строк: добавим 1 в одну позицию, так чтобы ранг увеличился на единицу (это возможно по утверждению 3.2). В конце получится, что мы добавили $n - k$ единиц (в каждую строку не более одной) так, чтобы ранг стал равен n. Последнее означает, что единицы были добавлены в разные столбцы. □

Следствие 3.4. Граф из условия леммы по лемме 1.2 является $(r, d + 2, \frac{1}{2}d - 2)$-граничным экспандером.

Глава 4

Нижние оценки для расщепления по линейным комбинациям

В данной главе мы рассмотрим обобщение DPLL-алгоритмов, которые могут расщепляться по линейным комбинациям переменных над полем $\mathbb{F}_2$. Мы докажем экспоненциальную нижнюю оценку на размер дерева расщеплений указанных алгоритмов для 2-цейтинских формул, которые могут быть получены из обычных цейтинских путем замены каждой переменной на конъюнкцию двух новых. План доказательства следующий: для каждой невыполнимой формулы ϕ мы определим задачу поиска $Search_\phi$ — по подстановке значений переменным найти опровергнутый дизъюнкт. Затем мы продемонстрируем перестройку дерева расщепления T в вероятностный коммуникационный протокол для задачи $Search_\phi$ глубины $O(\log |T| \log\log |T|)$ (значение некоторых переменных знает Алиса, а некоторых Боб). И наконец, мы применим нижнюю оценку на размер коммуникационного протокола для задачи $Search_{TS^2_{G,c}}$, где $TS^2_{G,c}$ — 2-кратная цейтинская формула. Данная оценка следует из работ [50] и [29].

Также в данной главе мы дадим простое доказательство нижней оценки $2^{\frac{n-1}{2}}$ на размер дерева линейных расщеплений для формул PHP^m_n, которые кодируют принцип Дирихле. Мы также покажем, что

для формул $PM(K_{n,n+1})$, которые кодируют существование совершенного паросочетания в полном двудольном графе $K_{n,n+1}$, существует полиномиальное по размеру дерево расщеплений по линейным комбинациям.

В разделе 4.5 мы рассмотрим обобщение резолюционной системы доказательств, которое работает с дизъюнкциями линейных уравнений над полем $\mathbb{F}_2$. У системы Res-Lin есть два правила вывода: ослабление и резолюция. Также мы рассмотрим семантический аналог данной системы — Sem-Lin, у которой второе правило вывода заменено на семантическое. Мы покажем, что дерево линейных расщеплений для невыполнимой формулы ϕ можно перестроить в древовидное доказательство в указанных системах, и наоборот. Таким образом, мы докажем нижнюю оценку на древесную версию указанных систем. В разделе 4.5.3 мы докажем, что система доказательств $R(lin)$, определенная в [14], p-моделирует наши системы доказательств.

Результаты данной главы опубликованы в работе [30].

4.1 Деревья расщеплений по линейным формам

Рассмотрим бинарное дерево T, в котором каждое ребро помечено линейным уравнением. Для каждой вершины v дерева T за Φ_v^T мы обозначим систему линейных уравнений, которая состоит из уравнений, написанных на пути от корня дерева T до вершины v.

Определение 4.1. Дерево линейных расщеплений для формулы в КНФ ϕ — это бинарное дерево, для которого выполнены следующие свойства:

- каждая внутренняя вершина v данного дерева помечена линейной формой f_v;
- ребра, исходящие из внутренней вершины v, помечены линейными уравнениями: $f_v = 0$ и $f_v = 1$;
- для каждого листа v верно ровно одно из следующих свойств:

– система Φ_v не имеет решений (будем называть такие листья вырожденными);

– система Φ_v имеет решения, но противоречит некоторому дизъюнкту C исходной формулы ϕ (будем говорить, что лист v опровергает дизъюнкт C);

– система Φ_v имеет единственное решение, которое выполняет формулу ϕ (будем называть такой лист выполняющим).

Можно рассматривать дерево линейных расщеплений как дерево рекурсивных вызовов алгоритма, который ищет выполняющий набор КНФ формулы ϕ. Алгоритм получает на вход КНФ формулу ϕ и линейную систему уравнений Φ, цель алгоритма — найти выполняющий набор для $\phi \wedge \Phi$. Изначально $\Phi = \mathit{True}$, на каждом шаге алгоритм как-то выбирает линейную форму f и значение $\alpha \in \{0, 1\}$, после чего алгоритм делает два рекурсивных вызова с параметрами $(\phi, \Phi \wedge \{f = \alpha\})$ и $(\phi, \Phi \wedge \{f = 1 - \alpha\})$. Алгоритм выполняет возврат в одном из трех случаев:

- система Φ не имеет решений;
- система Φ противоречит некоторому дизъюнкту C из формулы ϕ (система противоречит дизъюнкту $(l_1 \vee l_2 \vee \cdots \vee l_k)$ тогда и только тогда, когда для любого $i \in [k]$ система $\Phi \wedge \{l_i = 1\}$ не имеет решений, следовательно данное условие может быть проверено за полиномиальное время);
- система имеет единственное решение, которое выполняет формулу ϕ.

Заметим, что если достаточно найти один выполняющий набор, то алгоритм может остановится в первом выполняющем листе. Однако на невыполнимой формуле алгоритм обязан пройти полное дерево.

Предложение 4.1. По любому дереву линейных расщеплений T для формулы ϕ возможно построить дерево линейных расщеплений T_0 без

вырожденных листьев. При этом количество листьев в дереве T_0 не превосходит количество листьев в дереве T.

Доказательство. Пусть дерево T содержит такие вершины v (необязательно листья), что система Φ_v^T не имеет решений. Пусть w — ближайшая из них к корню. Вершина w не является корнем, поскольку система уравнений, соответствующая корню, пустая, а следовательно выполнимая. Пусть s — родитель вершин w и u (брат вершины w). Система Φ_s^T выполнима по выбору вершины w. Построим дерево T' при помощи удаления вершины w и стягивания ребра (s, u). T' — корректное дерево расщепления, поскольку для всех вершин v дерева T' система $\Phi_v^{T'}$ эквивалентна соответствующей системе дерева T. Продолжим применять данную операцию удаления, пока в дереве остаются вырожденные вершины. □

4.2 Верхние оценки

Предложение 4.2. Пусть ϕ — формула в КНФ, кодирующая невыполнимую систему линейных уравнений $\bigwedge_{i=1}^{m}(f_i = \beta_i)$ над полем $\mathbb{F}_2$, i-ое уравнение представлено в виде КНФ формулы ϕ_i и $\phi = \bigwedge_{i=1}^{m} \phi_i$ (возможно, что некоторые дизъюнкты встречаются более одного раза). Тогда существует дерево линейных расщеплений для формулы ϕ размера $O(|\phi|)$.

Доказательство. Опишем дерево T, в котором есть путь от корня до листа, помеченный уравнениями $(f_1 = \beta_1), (f_2 = \beta_2), \ldots, (f_m = \beta_m)$; данный лист является вырожденным, поскольку система не имеет решений. Для всех $i \in [m]$ i-ая вершина пути имеет ребенка u_i, к которому ведет ребро, помеченное $f_i = 1 - \beta_i$. Теперь опишем поддерево T_{u_i} с корнем в вершине u_i: это полное бинарное дерево, в котором происходит последовательное расщепление по всем переменным, входящим в формулу ϕ_i. Мы знаем, что $\Phi_{u_i}^T$ противоречит формуле ϕ_i, таким образом, каждый лист дерева T_{u_i} либо вырожденный, либо противоречит одному

из дизъюнктов ϕ_i. T_{u_i} имеет 2^k листьев, где k — количество переменных в формуле ϕ_i, однако хорошо известен факт, что для кодировки линейного уравнения из k — переменных в виде формулы в КНФ необходимо не менее 2^{k-1} дизъюнктов. Таким образом, размер дерева равен $O(|\phi|)$. $\square$

Для любого графа $G(V, E)$ определим формулу $PM(G)$, которая кодирует существование совершенного паросочетания в графе G. Каждому ребру $e \in E$ соответствует переменная x_e. Для каждой вершины $v \in V$ формула $PM(G)$ содержит дизъюнкт $\bigvee\limits_{(v,u)\in E} x_{(v,u)}$, и для каждой пары ребер $(v, u), (v, w) \in E$ данная формула содержит дизъюнкт $\neg x_{(v,u)} \vee \neg x_{(v,w)}$. Формула $PM(G)$ выполнима тогда и только тогда, когда в графе G существует совершенное паросочетание. Следующее предложение доказано Ицыксоном, для полноты изложения приведем его доказательство.

Предложение 4.3 ([30])**.** Если в графе $G(V, E)$ нечетное число вершин, то существует дерево линейных расщеплений полиномиального размера для формулы $PM(G)$.

Доказательство. Опишем бинарное дерево T, которое имеет путь от корня до листа, помеченный уравнениями $\sum\limits_{(v,u)\in E} x_{(v,u)} = 1$ для всех вершин v. Данный лист является вырожденным, поскольку в графе нечетное число вершин, и сумма всех указанных уравнений приведет к противоречию $0 = 1$. Для каждого i, i-ая вершина на пути имеет также ребенка u_i, ребро к которому помечено уравнением $\sum\limits_{(v_i,u)\in E} x_{(v_i,u)} = 0$. Теперь опишем поддерево T_{u_i} с корнем в вершине u_i. Это дерево расщепления по всем переменным $x_{v_i,u} \in E$. Заметим, что как только мы подставим двум переменным значение 1, то мы сразу получим противоречие с дизъюнктом $\neg x_{(v_i,w_1)} \vee \neg x_{(v_i,w_2)}$. Лист, соответствующий подстановкам только одной переменной значение 1, является вырожденным, так как данная подстановка противоречит уравнению $\sum\limits_{(v_i,u)\in E} x_{(v_i,u)} = 0$. А лист, соответствующий только нулевым подстановкам противоречит

уравнению $\sum_{(v_i,u)\in E} x_{(v_i,u)} = 1$. Таким образом, размер T_{u_i} равен $O(|V|^2)$, следовательно размер дерева T равен $O(|V|^3)$. □

4.3 Нижняя оценка для 2-кратных цейтинских формул

В данном разделе мы докажем нижнюю оценку на размер дерева линейных расщеплений. Доказательство будет состоять из двух частей. В первой части мы по доказательству построим коммуникационный протокол, а во второй мы докажем нижнюю оценку на указанный протокол.

4.3.1 Коммуникационный протокол из дерева линейных расщеплений

Пусть ϕ — невыполнимая формула в КНФ. Для каждой подстановки значений переменным найдется такой дизъюнкт из формулы ϕ, который опровергается данной подстановкой. Обозначим $Search_\phi$ следующую задачу поиска: по подстановке значений переменным формулы ϕ найти опровергнутый дизъюнкт.

Рассмотрим функцию или задачу поиска f со входом $\{0,1\}^n$, разобьем множество $[n]$ на две непересекающиеся части X и Y. Алиса знает биты входа, соответствующие части X, а Боб — части Y. Вероятностный коммуникационный протокол с публичными случайными битами и вероятностью ошибки ϵ — это такое бинарное дерево, что каждая внутренняя вершина v помечена функцией одного из двух типов: $a_v : \{0,1\}^X \times \{0,1\}^R \to \{0,1\}$ или $b_v : \{0,1\}^Y \times \{0,1\}^R \to \{0,1\}$, где R — целое число, которое обозначает количество случайных битов, используемых протоколом. Один ребенок каждой внутренней вершины помечен 0, а другой 1, листья помечены строками (ответами протокола). Предположим, что Алиса знает $x \in \{0,1\}^X$, а Боб знает $y \in \{0,1\}^Y$, и оба они знают строку $r \in \{0,1\}^R$. Алиса и Боб общаются согласно протоколу следующим образом. Изначально они кладут фишку в корень дерева.

На каждом шаге, когда фишка лежит в вершине, помеченной функцией a_v, Алиса вычисляет значение $a_v(x, r)$ и посылает результат Бобу; если вершина помечена b_v, то Боб вычисляет значение $b_v(y, r)$ и посылает результат Алисе. После этого игроки перемещают фишку с ребенка, соответствующего переданному биту. Коммуникация останавливается, когда игроки перемещают фишку в лист, метка данного листа является ответом. Необходимо, чтобы с вероятностью $1-\epsilon$ по случайным битам ответ протокола совпадал со значением функции f. Сложностью коммуникационного протокола называется глубина дерева или, что эквивалентно, количество переданных между игроками битов в худшем случае. Будем обозначать $R_\epsilon^{pub}(f)$ сложность вероятностного коммуникационного протокола для функции f с ошибкой ϵ.

Пусть $EQ : \{0,1\}^{2n} \to \{0,1\}$ и $EQ(x, y) = 1 \Leftrightarrow x = y$. Будем считать, что Алиса знает x, а Боб знает y.

Следующая лемма доказана в [51]. Приведем ее доказательство, чтобы в дальнейшем использовать приведенный в нем протокол.

Лемма 4.1 ([51]). $R_\delta^{pub}(EQ) \leq \lceil \log(\frac{1}{\delta}) \rceil + 1$.

Доказательство. Рассмотрим следующий протокол глубины 2: Алиса посылает Бобу внутреннее произведение $< x, r >= \sum_{i=1}^{n} x_i r_i \ mod\ 2$, где $r \in \{0,1\}^n$ — строка случайных битов. Боб вычисляет значение внутреннего произведения $< y, r >$ и посылает 1 тогда и только тогда, когда его результат и результат Алисы одинаковы. Результат протокола — бит переданный Бобом. Если $x = y$, то данный протокол всегда выдает корректный ответ. Если $x \neq y$, то по принципу случайных подсумм результат будет корректным с вероятностью $\frac{1}{2}$. Для уменьшения ошибки до δ, Алиса последовательно шлет Бобу $\lceil \log(\frac{1}{\delta}) \rceil$ внутренних произведений x с независимыми случайными строками $r_1, r_2, \ldots, r_{\lceil \log(\frac{1}{\delta}) \rceil} \in \{0,1\}^n$. Боб проверяет, что его внутренние произведения равны произведениям Алисы, и посылает 1 если равны 0 иначе. Если $x = y$, то данный протокол всегда выдает корректный ответ. Если $x \neq y$, то результат будет корректным с вероятностью $1 - 2^{\lceil \log(\frac{1}{\delta}) \rceil} \geq 1 - \delta$. □

Лемма 4.2. Рассмотрим несколько линейных уравнений над $\mathbb{F}_2$. Пусть Алиса знает значение для некоторых переменных, а Боб знает значение остальных. Тогда существует вероятностный коммуникационный протокол с публичными битами и ошибкой δ, который использует $O(\log(\frac{1}{\delta}))$ коммуникации.

Доказательство. Пусть мы имеем t уравнений. Когда мы хотим проверить i-ое уравнение, то Алиса должна посчитать сумму значений ее переменных, а Боб — сумму значений своих, после чего они должны убедиться, что сумма их значений равна правой части уравнения α_i. Пусть сумма значений Алисы переменных i-ого уравнения плюс α_i равна z_i, а сумма значений Боба переменных i-ого уравнения равна y_i. Все уравнения будут верны тогда и только тогда, когда $EQ(z_1 z_2 \dots z_t, y_1 y_2 \dots y_t) = 1$. Для подсчета EQ будем использовать протокол из леммы 4.1. □

Теорема 4.1. Пусть ϕ — невыполнимая формула в КНФ, и T — дерево линейных расщеплений для ϕ. Тогда для любого распределения переменных между Алисой и Бобом $R^{pub}_{\frac{1}{3}}(Search_\phi) = O(\log(|T|) \log\log(|T|))$.

Доказательство. Построим коммуникационный протокол по дереву T без вырожденных листьев. Алиса и Боб вместе знают подстановку π переменным формулы ϕ (часть подстановки знает Алиса, а часть — Боб). Подстановка π определяет путь ℓ_π в дереве T, который соответствует ребрам, помеченным уравнениями, которые выполняет π. Данный путь содержит лист, помеченный дизъюнктом C_π формулы ϕ, который опровергается подстановкой π. Протокол, который мы описываем, с большой вероятностью будет выдавать данный дизъюнкт.

Протокол состоит из $O(\log(|T|))$ раундов. При анализе раунда мы будем полагать, что все предыдущие раунды завершились без ошибок. Таким образом, вероятность ошибки в протоколе может быть ограничена суммой вероятностей ошибки в каждом раунде. В начале i-ого раунда Алиса и Боб знают дерево T_i — связный подграф исходного дерева T, $T_1 = T$. Так как T_i — связный подграф, то корень данного дерева является предком всех остальных вершин T_i. В предположении, что предыду-

щие раунды закончились без ошибок, мы будем считать, что T_i содержит часть пути ℓ_π, который идет от корня T_i к листу, в котором опровергается C_π. Мы также будем поддерживать неравенство $|T_{i+1}| \leq \frac{2}{3}|T_i|$. Таким образом, если T_i состоит из одной вершины, то это должен быть лист, помеченный дизъюнктом C_π.

Пусть $|T_i| > 1$, тогда существует такая вершина v в дереве T_i, что размер поддерева с корнем в вершине v (обозначим его $T_i^{(v)}$) не менее $\frac{1}{3}|T_i|$ и не более $\frac{2}{3}|T_i|$. Дерево T_{i+1} равно дереву $T_i^{(v)}$, если $v \in \ell_\pi$, и равно $T \setminus T_i^{(v)}$ иначе. Алиса и Боб используют одинаковый алгоритм для поиска вершины v, теперь они должны проверить, лежит ли v на пути ℓ_π. Вершина v лежит на пути ℓ_π тогда и только тогда, когда π выполняет все уравнения на пути от корня T_i до вершины v. Алиса и Боб проверяют данные уравнения, пользуясь методом, описанным в лемме 4.2 с $\delta = \frac{1}{3\lceil \log_{\frac{3}{2}} |T| \rceil}$. Так как количество раундов не превосходит $\lceil \log_{\frac{3}{2}} |T| \rceil$, то суммарная ошибка протокола не более $\frac{1}{3}$. Глубина данного протокола равна количеству раундов $\lceil \log_{\frac{3}{2}} |T| \rceil$, умноженному на глубину протокола для функции EQ — $O(\log\log(|T|))$. □

4.3.2 Нижняя оценка на коммуникационную сложность

Цейтинская формула $TS_{G,c}$ может быть построена из произвольного графа $G(V, E)$ и функции $c : V \to \mathbb{F}_2$, переменные формулы $TS_{G,c}$ соответствуют ребрам графа G. Формула $TS_{G,c}$ — конъюнкция следующих условий для каждой вершины $v \in V$, записанных в КНФ: четность числа ребер, инцидентных вершине v, которые имеют значение 1, равна $c(v)$. Хорошо известно, что формула $TS_{G,c}$ невыполнима тогда, и только тогда, когда существует такая компонента связности S графа G, что $\sum\limits_{v \in S} c(v) = 1$.

k-кратная цейтинская формула $TS^k_{G,c}$ [29] может быть получена из цейтинской формулы $TS_{G,c}$ путем подстановки вместо каждой переменной x_i конъюнкции k новых переменных $(z_{i1} \wedge z_{i2} \wedge \cdots \wedge z_{ik})$ и записи

полученной формулы в КНФ. Заметим, что если степень графа G ограничена константой, то для каждого фиксированного k формула $TS^k_{G,c}$ имеет полиномиальный размер относительно $|V|$.

Теорема 4.2. За полиномиальное от n время можно построить граф $G(V,E)$ на n вершинах с максимальной степенью, ограниченной константой, и такую функцию $c : V \to \mathbb{F}_2$, что $TS^2_{G,c}$ является невыполнимой, и $R^{pub}_{\frac{1}{3}} = \Omega(\frac{n^{1/3}}{(\log(n)\log\log(n))^2})$.

Следствие 4.1. В условиях теоремы 4.2 размер любого дерева линейных расщеплений $TS^2_{(G,c)}$ составляет по крайней мере $\Omega\left(2^{n^{1/3}/\log^3(n)}\right)$.

Доказательство следствия 4.1. Следует из теоремы 4.2 и теоремы 4.1. □

Определим функцию $DISJ_{n,2} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}$, для всех $x, y \in \{0,1\}^n \quad DISJ_{n,2}(x,y) = 1$ тогда и только тогда, когда $x_i \wedge y_i = 0$ для всех $i \in [n]$.

Теорема 4.3. ([29]) Пусть $m = \frac{n^{1/3}}{\log(n)}$. Тогда за полиномиальное от n время можно построить граф $G(V,E)$ на n вершинах с максимальной степенью, ограниченной константой, и такую функцию $c : V \to \mathbb{F}_2$, что $TS^2_{(G,c)}$ невыполнима, и $R^{pub}_{\epsilon}(DISJ_{m,2}) = O\left(R^{pub}_{\epsilon}(Search_{TS^2_{(G,c)}})\log(n)(\log\log(n))^2\right)$.

Лемма 4.3. ([50]) $R^{pub}_{1/3}(DISJ_{n,2}) = \Omega(n)$.

Доказательство теоремы 4.2. Пусть $m = \frac{n^{1/3}}{\log(n)}$. Из леммы 4.3 следует, что $R^{pub}_{\epsilon}(DISJ_{m,2}) = \Omega(m)$, таким образом по теореме 4.3 возможно построить такой граф G и такую функцию c, что $R^{pub}_{1/3}\left(Search_{TS^2_{(G,c)}}\right) = \Omega\left(\frac{n^{1/3}}{(\log(n)\log\log(n))^2}\right)$. □

4.4 Нижняя оценка для принципа Дирихле

В данном разделе мы докажем нижнюю оценку на размер дерева линейных расщеплений для формул PHP^m_n, которые кодируют принцип

Дирихле. Переменные формулы PHP_n^m — $p_{i,j}$, где $i \in [m]$, $j \in [n]$; $p_{i,j}$ показывает, содержится ли i-ый кролик в j-ой клетке. Формула состоит из дизъюнктов двух типов: 1) длинные дизъюнкты, которые кодируют, что каждый кролик находится в какой-нибудь клетке: $p_{i,1} \vee p_{i,2} \cdots \vee p_{i,n}$ для всех $i \in [m]$; 2) короткие дизъюнкты, которые кодируют, что в каждой клетке находится не более одного кролика: $\neg p_{i,k} \vee \neg p_{j,k}$ для всех $i \neq j \in [m]$ и $k \in [n]$. Если $m > n$, то формула PHP_n^m невыполнима.

Будем называть подстановку значений переменным $p_{i,j}$ *допустимой*, если она выполняет все короткие дизъюнкты. Другими словами, в любой допустимой подстановке в каждой клетке содержится не более одного кролика.

Результаты данного раздела доказаны Ицыксоном, для полноты изложения доказательство теоремы.

Лемма 4.4 ([30])**.** Пусть линейная система $Ap = b$ с переменными $p = (p_{i,j})_{i\in[m],j\in[n]}$, состоящая из не более $\frac{n-1}{2}$ уравнений, имеет допустимое решение. Тогда для любого $i \in [m]$ данная система имеет допустимое решение, которое выполняет длинный дизъюнкт $p_{i,1} \vee p_{i,2} \vee \cdots \vee p_{i,n}$.

Теорема 4.4 ([30])**.** Для всех $m > n$ все деревья линейных расщеплений для PHP_n^m имеют размер не менее $2^{\frac{n-1}{2}}$.

Доказателъство. Скажем, что уравнение $f = \alpha$ допустимо следует из линейной системы Φ, если любое допустимое решение Φ выполняет $f = \alpha$.

Рассмотрим дерево линейных расщеплений T для PHP_n^m. Удалим из дерева T все вершины v, в которых Φ_v^T не имеет допустимых решений. Получившийся граф по прежнему является деревом, так как удаляя вершину мы должны удалить и все ее поддерево. Обозначим получившееся дерево T'. Заметим, что невозможна ситуация, что лист T' не является листом T. В самом деле, предположим, что вершина v в дереве T помечена линейной формой f, тогда все допустимые подстановки, которые выполняют Φ_v^T, также выполняют одну из систем $\Phi_v^T \wedge (f = 1)$ или $\Phi_v^T \wedge (f = 0)$, таким образом, если вершина v осталась в дереве T', то

и один из ее потомков останется в дереве T'. Следовательно, в каждом листе ℓ дерева T' система Φ_ℓ^T опровергает дизъюнкт из формулы PHP_n^m. Так как существует допустимая подстановка, которая выполняет Φ_ℓ^T, то Φ_ℓ^T не может опровергать короткий дизъюнкт, следовательно в данном листе опровергается длинный дизъюнкт.

Рассмотрим вершину v дерева T' с единственным ребенком u, пусть ребро (u, v) помечено $f = \alpha$. Мы знаем, что система $\Phi_v^T \wedge (f = 1 + \alpha)$ не имеет допустимых решений. Таким образом, уравнение $f = \alpha$ допустимо следует из Φ_v^T; и множества допустимых решений $\Phi_u^{T'}$ и $\Phi_v^{T'}$ одинаковы.

Пусть T' содержит вершину v с единственным ребенком u; объединим вершины u и v в одну и удалим ребро (u, v). Будем повторять данную операцию, пока в текущем дереве остаются вершины с одним ребенком. Обозначим получившееся дерево T''. Пусть V' — множество вершин T', и V'' — множество вершин T''. Определим сюръективное отображение $\mu : V' \to V''$, которое вершинам из дерева T' сопоставляет вершины из дерева T'', в которые они были объединены. Мы знаем, что для всех $u \in T'$ множества допустимых решений $\Phi_u^{T'}$ и $\Phi_{\mu(u)}^{T''}$ совпадают.

Для каждого листа ℓ'' дерева T'' существует такой лист ℓ' дерева T', что $\mu(\ell') = \ell''$. Система $\Phi_{\ell'}^{T'}$ опровергает длинный дизъюнкт $p_{i,1} \vee \cdots \vee p_{i,n}$, таким образом система $\Phi_{\ell''}^{T''}$ не имеет допустимых решений, которые выполняют $p_{i,1} \vee \cdots \vee p_{i,n}$. Все внутренние вершины дерева T'' имеют двух сыновей. Лемма 4.4 влечет тот факт, что глубина всех листьев в дереве T'' не менее $\frac{n-1}{2}$, таким образом, размер дерева T'' составляет не менее $2^{(n-1)/2}$. □

4.5 Системы доказательств Res-Lin и Sem-Lin

Линейным дизъюнктом назовем дизъюнкцию линейных уравнений $\bigvee_{i=1}^{k}(f_i = \alpha_i)$, где f_i — линейная форма и $\alpha_i \in \mathbb{F}_2$. Мы также можем рассматривать линейный дизъюнкт как отрицание системы линейных уравнений $\neg \bigwedge_{i=1}^{n}(f_i = 1 + \alpha_i)$. Тривиальный линейный дизъюнкт — линейный

дизъюнкт, который тождественно равен истине. Дизъюнкт $\neg \bigwedge_{i=1}^{n} (f_i = \alpha_i)$ тривиален тогда и только тогда, когда система $\bigwedge_{i=1}^{n} (f_i = \alpha_i)$ не имеет решений.

Линейная формула в КНФ — конъюнкция линейных дизъюнктов. Будем говорить, что пропозициональная формула ϕ семантически следует из множества формул $\psi_1, \psi_2, \ldots, \psi_k$, если любая подстановка, которая выполняет ψ_i для всех $i \in [k]$, также выполняет ϕ.

Определим систему доказательств Res-Lin, которая может быть использована для доказательства невыполнимости линейной КНФ формулы. Данная система имеет два правила:

- правило ослабления: из линейного дизъюнкта C можно вывести любой линейный дизъюнкт D, который семантически следует из C, т.е. формула $C \to D$ является тавтологией;
- правило резолюции: из линейных дизъюнктов $(f = 0) \vee D$ и $(f = 1) \vee D'$ можно вывести дизъюнкт $D \vee D'$.

Вывод линейного дизъюнкта C из линейной формулы ϕ в КНФ в системе Res-Lin — это последовательность линейных дизъюнктов, которая заканчивается дизъюнктом C, и каждый дизъюнкт которой является либо дизъюнктом формулы ϕ, либо получен из предыдущих дизъюнктов при помощи правил вывода. Доказательство невыполнимости линейной формулы в КНФ — это вывод пустого дизъюнкта. Система доказательств Sem-Lin отличается от системы Res-Lin заменой второго правила. Данное правило заменяется на семантическое: из линейных дизъюнктов C_1, C_2 можно вывести любой линейный дизъюнкт C_0, который семантически следует из них, т.е. формула $(C_1 \wedge C_2) \to C_0$ является тавтологией.

Для того, чтобы проверить, что Sem-Lin и Res-Lin являются системами доказательств в смысле [9], мы должны проверить, что возможно проверить корректность доказательства за полиномиальное время. До-

статочно проверить корректность применения правил вывода. Корректность применения правила резолюции проверяется тривиальным образом, для проверки применения остальных правил мы докажем следующее утверждение.

Предложение 4.4. За полиномиальное время возможно проверить:

- является ли линейный дизъюнкт $C_0 = \neg \bigwedge_{i \in I}(f_i = \alpha_i)$ результатом ослабления линейного дизъюнкта $C_1 := \neg \bigwedge_{i \in J}(g_i = \beta_i)$;
- следует ли семантически линейный дизъюнкт $C_0 := \neg \bigwedge_{i \in J}(g_i = \beta_i)$ из линейных дизъюнктов $C_1 := \neg \bigwedge_{i \in J}(g_i = \beta_i)$ и $C_2 = \neg \bigwedge_{i \in K}(h_i = \gamma_i)$.

Доказательство. 1) Линейный дизъюнкт C_0 является ослаблением C_1 тогда и только тогда, когда любая выполняющая подстановка C_1 выполняет также и C_0. Покажем, что C_0 является ослаблением C_1 тогда и только тогда, когда для всех $j \in J$ система $\bigwedge\limits_{i \in I}(f_i = \alpha_i) \wedge (h_j = \beta_j + 1)$ не имеет решений. Действительно, если система имеет решения, то решение выполняет C_1 и опровергает C_0. Пусть C_0 не является ослаблением C_1, тогда существует подстановка, выполняющая C_1 и опровергающая C_0, данная подстановка выполняет уравнение $h_j = \beta_j + 1$ для некоторого $j \in J$, таким образом, данная подстановка выполняет $\bigwedge\limits_{i \in I}(f_i = \alpha_i) \wedge (h_j = \beta_j + 1)$.

Таким образом, для проверки корректности применения правила ослабления достаточно проверить, что для любого $j \in J$ соответствующая система не имеет решений.

2) Аналогично пункту 1 можно показать, что C_0 семантически следует из C_1 и C_2 тогда и только тогда, когда для любых $j \in J$ и $k \in K$ система $\bigwedge\limits_{i \in I}(f_i = \alpha_i) \wedge (g_j = \beta_j + 1) \wedge (h_k = \gamma_k + 1)$ не имеет решений. $\square$

Предложение 4.5. Правило ослабления может быть промоделировано полиномиальным числом применений чисто синтаксических правил:

- *упрощение*: позволяет вывести линейный дизъюнкт D из линейного дизъюнкта $D \vee (0 = 1)$;

- *синтаксическое ослабление*: позволяет вывести $D \vee (f = \alpha)$ из D;
- *сложение*: позволяет вывести $D \vee (f_1 = \alpha_1) \vee (f_1 + f_2 = \alpha_1 + \alpha_2 + 1)$ из $D \vee (f_1 = \alpha_1) \vee (f_2 = \alpha_2)$.

Доказательство. Будем представлять линейные дизъюнкты в виде отрицания системы уравнений. В данном представлении правило сложения позволяет прибавлять к одному уравнению системы другое, а правило упрощения позволяет убрать тривиальное уравнение $0 = 0$.

Пусть дизъюнкт $\neg \bigwedge\limits_{i \in I} (g_i = \beta_i)$ — результат ослабления дизъюнкта $\neg \bigwedge\limits_{i \in J} (f_i = \alpha_i)$.

Применив правило синтаксического ослабления несколько раз получим дизъюнкт $\neg \bigwedge\limits_{i \in J} (f_i = \alpha_i) \wedge \bigwedge\limits_{i \in I} (g_i = \beta_i)$.

Из утверждения 4.4 мы знаем, что любое уравнение $f_i = \alpha_i$ является линейной комбинацией уравнений $g_j = \beta_j$. Таким образом, можно получить уравнение $0 = 0$ из любого $f_i = \alpha_i$ путем многократного применения правила сложения. В завершении удалим уравнение $0 = 0$, пользуясь правилом упрощения. □

Покажем, что системы Sem-Lin и Res-Lin полиномиально эквивалентны. Это означает, что любое доказательство в одной системе может быть переделано в доказательство в другой за полиномиальное время. Любое доказательство в системе Res-Lin является также доказательством и в системе Sem-Lin, таким образом нужно построить преобразование в обратную сторону.

Предложение 4.6. Пусть нетривиальный линейный дизъюнкт $C_0 := \neg \bigwedge\limits_{i \in I} (f_i = \alpha_i)$ семантически следует из $C_1 := \neg \bigwedge\limits_{i \in J} (g_i = \beta_i)$ и $C_2 := \neg \bigwedge\limits_{i \in L} (h_i = \gamma_i)$. Тогда C_0 может быть получен из C_1 и C_2 путем применения одного правила резолюции и полиномиального числа правил ослабления.

Перед доказательством рассмотрим пример, который проиллюстрирует вывод линейного дизъюнкта $(x + y = 0)$ из дизъюнктов $(x = 0)$

и $(y = 0)$ в Res-Lin: 1) применим правило ослабления к $(x = 0)$ и получим $(x + y = 0) \vee (y = 1)$; 2) применим правило резолюции к $(x + y = 0) \vee (y = 1)$ и $(y = 0)$ для получения $(x + y = 0)$.

Будем использовать следующую хорошо известную лемму:

Лемма 4.5. Если для матрицы $A \in \mathbb{F}_2^{m \times n}$ и вектора $b \in \mathbb{F}_2^m$ линейная система $Ax = b$ не имеет решений, то существует такой вектор $y \in \mathbb{F}_2^m$, что $y^T A = 0$ и $y^T b = 1$. Другими словами, если линейная система над $\mathbb{F}_2$ невыполнима, то возможно сложить несколько уравнений так, чтобы получить противоречие $0 = 1$.

Доказательство утверждения 4.6. Оба C_1 и C_2 не могут быть тривиальными, иначе C_0 должен был бы быть тривиальным. Если C_i, где $i \in \{1, 2\}$, тривиален, то C_0 является ослаблением C_{3-i}. Не умаляя общности можно считать, что C_1 и C_2 нетривиальны.

Для всех $j \in J$ и $l \in L$ система $\bigwedge\limits_{i \in I}(f_i = \alpha_i) \wedge (g_j = 1 + \beta_j) \wedge (h_l = 1 + \gamma_l)$ невыполнима. Так как система $\bigwedge\limits_{i \in I}(f_i = \alpha_i)$ выполнима, то выполнено одно из следующих утверждений:

- система $\bigwedge\limits_{i \in I}(f_i = \alpha_i)$ становится невыполнимой, если мы добавим одно уравнение (например $g_j = 1 + \beta_j$). Тогда по лемме 4.5 отрицание данного уравнения может быть получено как линейная комбинация уравнений из системы $\bigwedge\limits_{i \in I}(f_i = \alpha_i)$;
- система $\bigwedge\limits_{i \in I}(f_i = \alpha_i)$ становится невыполнимой, только если добавить оба уравнения $(g_j = 1 + \beta_j) \wedge (h_l = 1 + \gamma_l)$. По лемме 4.5 уравнение $g_j + h_l = \beta_j + \gamma_l + 1$ может быть получено как линейная комбинация уравнений из системы $\bigwedge\limits_{i \in I}(f_i = \alpha_i)$.

Заметим, что если уравнения $g_j = 1 + \beta_j$ и $h_l = 1 + \gamma_l$ противоречат друг другу (т.е. $g_j = h_l$ и $\beta_j = 1 + \gamma_l$), то уравнение $g_j + h_l = \beta_j + \gamma_l + 1$ тривиально $(0 = 0)$.

Разделим множество J на две непересекающиеся части J' и J'', где $j \in J''$ тогда и только тогда, когда система $\bigwedge\limits_{i\in I}(f_i = \alpha_i) \wedge (g_j = \beta_j + 1)$ невыполнима. Аналогично определим разбиение $L = L' \cup L''$. Заметим, что если $J = J''$, то $\neg \bigwedge\limits_{i\in I}(f_i = \alpha_i)$ является ослаблением $\neg \bigwedge\limits_{j\in J}(g_j = \beta_j)$, аналогично, если $L = L''$, то $\neg \bigwedge\limits_{i\in I}(f_i = \alpha_i)$ является ослаблением $\neg \bigwedge\limits_{i\in L}(h_i = \gamma_i)$. Таким образом, предположим, что $J' \neq \emptyset$ и $L' \neq \emptyset$.

Мы получили, что C_0 является ослаблением $D := \neg(\bigwedge\limits_{i\in J''}(g_i = \beta_i) \wedge \bigwedge\limits_{i\in L''}(h_i = \gamma_i) \wedge \bigwedge\limits_{i\in J', j\in L'}(g_i + h_j = \beta_i + \gamma_j + 1)$. Осталось показать, что D может быть получено из C_1 и C_2 как результат применения одного резолюционного правила и нескольких правил ослабления.

Пусть $j_0 \in J'$ и $l_0 \in L'$.

1) Применим правило ослабления к C_1 и получим $D_1 := \neg\left((g_{j_0} = \beta_{j_0}) \wedge \bigwedge\limits_{i\in J'}(g_i + h_{l_0} = \beta_i + \gamma_{l_0} + 1) \wedge \bigwedge\limits_{i\in J''}(g_i = \beta_i)\right)$;

2) Применим правило ослабления к C_2 и получим $D_2 := \neg\left((g_{j_0} = \beta_{j_0} + 1) \wedge \bigwedge\limits_{i\in L'}(h_i + g_{j_0} = \beta_{j_0} + \gamma_{l_0} + 1) \wedge \bigwedge\limits_{i\in L''}(h_i = \gamma_i\})\right)$;

3) Применим правило резолюции к D_1 и D_2, получим

$$D_3 := \neg\left(\bigwedge_{i\in J'}(g_i + h_{l_0} = \beta_i + \gamma_{l_0} + 1) \wedge \bigwedge_{i\in L'}(h_i + g_{j_0} = \beta_{j_0} + \gamma_{l_0} + 1) \wedge \bigwedge_{i\in J''}(g_i = \beta_i) \wedge \bigwedge_{i\in L''}(h_i = \gamma_i)\right)$$

4) Применим правило ослабления к D_3 и получим D. □

4.5.1 Древовидная Res-Lin и деревья линейных расщеплений

Назовем доказательство в Res-Lin (или Sem-Lin) древовидным, если все дизъюнкты могут быть помещены в вершины корневого дерева так, что: 1) пустой дизъюнкт находится в корне; 2) дизъюнкты исходной формулы

являются листьями; 3) дизъюнкт в каждой внутренней вершине является результатом применения одного из правил к его детям.

Лемма 4.6. 1. Каждое дерево линейных расщеплений для невыполнимой формулы в КНФ может быть перестроено в древовидное Res-Lin доказательство и размер получившегося доказательства не более, чем в два раза превосходит размер дерева расщеплений.

2. Каждое древовидное Res-Lin доказательство невыполнимости формулы ϕ может быть перестроено в дерево линейных расщеплений для ϕ без увеличения размера дерева.

Доказательство. Начнем с перестройки дерева расщеплений в дерево расщеплений без вырожденных вершин методом, описанным в утверждении 4.1. Обозначим получившееся дерево T. В каждую вершину v дерева T поместим линейный дизъюнкт $\neg\Phi_v^T$. По построению каждый дизъюнкт является результатом применения правила резолюции к дизъюнктам, находящимся в детях; корень содержит пустой дизъюнкт. В каждом листе ℓ система Φ_ℓ^T опровергает некоторой линейный дизъюнкт C исходной формулы. Таким образом, $\neg T_v$ является ослаблением C. Размер получившегося доказательства не превосходит размера дерева плюс не более одного правила ослабления на каждый лист.

2) Рассмотрим дерево доказательства и стянем все ребра, соответствующие применению правила ослабления. Обозначим получившееся дерево T. Все ребра в данном дереве соответствуют применению правила резолюции. Пусть правило резолюции было применено к дизъюнктам $\neg((f = 0) \wedge D_1)$ и $\neg((f = 1) \wedge D_2)$, тогда мы пометим ребро, идущее к первому дизъюнкту, $f = 0$, а второе $f = 1$.

Покажем, что для каждой вершины v все дизъюнкты, написанные в v, противоречат системе Φ_v^T. Каждая вершина может содержать несколько дизъюнктов (так как мы стянули некоторые вершины), достаточно доказать указанный факт для самого слабого из них (т.е. для дизъюнкта, который используется в резолюции). Докажем индукцией по глубине

вершины v. В корне содержится пустой дизъюнкт, следовательно, утверждение верно для корня. Предположим, что мы доказали утверждение для вершины v, теперь докажем для ее детей u и w. Пусть $\neg(D_1 \wedge D_2)$ — дизъюнкт в v, и пусть он является результатом применения резолюционного правила к $\neg(f = 0 \wedge D_1)$ и $\neg(f = 1 \wedge D_2)$. По индукционному предположению мы знаем, что $\neg(D_1 \wedge D_2)$ противоречит системе Φ_v^T. Это означает, что отрицание каждого уравнения в D_1 противоречит системе Φ_v^T. Пусть дизъюнкт $\neg(f = 0 \wedge D_1)$ находится в вершине u, тогда $\Phi_u^T = \Phi_v^T \wedge (f = 0)$, следовательно $f = 1$ противоречит Φ_v^T; отрицание всех уравнений из D_1 противоречит Φ_v^T и таким образом противоречит Φ_u^T. Мы получили, что Φ_u^T противоречит $\neg(f = 0 \wedge D_1)$ и по аналогии это верно для $\neg(f = 1 \wedge D_2)$. Применяя утверждение к листьям, мы получаем, что каждый лист опровергает дизъюнкт формулы ϕ. □

Следствие 4.2. 1) Для всех $m > n$ каждое древовидное доказательство в Res-Lin и Sem-Lin формулы PHP_n^m имеет размер $2^{\Omega(n)}$. 2) В условиях теоремы 4.2 размер любого древовидного доказательства в Res-Lin и Sem-Lin формулы $TS^2_{(G,c)}$ имеет размер $\Omega(2^{n^{\frac{1}{3}}/\log^3(n)})$.

Доказательство. Следует из леммы 4.6, утверждения 4.6, теоремы 4.4 и теоремы 4.2. □

4.5.2 Импликативная полнота Res-Lin

В данном разделе мы докажем, что Res-Lin является импликативно полной.

Лемма 4.7. 1. Если линейный дизъюнкт D является ослаблением линейного дизъюнкта C, то для любого линейного дизъюнкта E дизъюнкт $D \vee E$ является ослаблением $C \vee E$.

2. Если линейный дизъюнкт D семантически следует из (или является результатом резолюции) C и F, то для любого линейного дизъюнкта E, дизъюнкт $D \vee E$ семантически следует из (или является результатом резолюции) $C \vee E$ и $F \vee E$.

Теорема 4.5. Если линейный дизъюнкт C_0 семантически следует из $C_1, C_2, \ldots, C_k$, то C_0 может быть выведен из $C_1, C_2, \ldots, C_k$ в Res-Lin.

Доказательство. Будем действовать по следующему плану: построим список таких линейных дизъюнктов $\mathcal{D}$, что конъюнкция всех дизъюнктов из данного списка будет невыполнима. Так как Res-Lin — полная система (это следует из того факта, что для любой линейной КНФ можно рассмотреть дерево расщеплений по всем переменным), существует вывод пустого дизъюнкта из $\mathcal{D}$. По лемме 4.7 из списка $\mathcal{D}' := \{D \vee C_0 \mid D \in \mathcal{D}\}$ возможно вывести дизъюнкт C_0. После этого мы покажем, что любой дизъюнкт из списка $\mathcal{D}'$ является ослаблением одного из дизъюнктов $C_1, C_2, \ldots, C_k$.

Будем строить список $\mathcal{D}$ шаг за шагом. Первоначально $\mathcal{D}$ состоит из дизъюнктов $C_1, C_2, \ldots, C_k$. Заметим, что если подстановка π опровергает C_0, то по условию теоремы она также опровергает один из дизъюнктов $C_1, C_2, \ldots, C_k$, таким образом, она опровергает и их конъюнкцию. Пусть $C_1 := \bigvee\limits_{i=1}^{n}(f_i = \alpha_i)$ и $C_0 := \bigvee\limits_{i=1}^{m}(g_i = \beta_i)$. Пока существует подстановка π, выполняющая C_0 и все дизъюнкты из $\mathcal{D}$, будем добавлять в $\mathcal{D}$ новый дизъюнкт C^π. Так как π выполняет C_0, существует такое i, что π выполняет $g_i = \beta_i$. Обозначим $I := \{i \mid \pi \text{ выполняет} f_i = \alpha_i\}$ и дизъюнкт $C^\pi := \bigvee\limits_{i \in I}(f_i + g_i = \alpha_i + \beta_i + 1) \vee \bigvee\limits_{i \notin I}(f_i = \alpha_i)$. Из конструкции следует, что π опровергает C^π.

Заметим, что для любой подстановки значений переменным найдется дизъюнкт в списке $\mathcal{D}$, который будет опровергнут подстановкой. Следовательно, конъюнкция дизъюнктов из $\mathcal{D}$ невыполнима. Мы должны показать, что для всех $D \in \mathcal{D}$ дизъюнкт $D \vee C_0$ является ослаблением одного из дизъюнктов $C_1, C_2, \ldots, C_k$. Если D равен одному из дизъюнктов $C_1, C_2, \ldots, C_k$, то утверждение тривиально. Пусть $D = C^\pi$, тогда $D \vee C_0$ является ослаблением $C_1 \vee C_0$ и, следовательно, C_1. $\square$

4.5.3 Моделирование Res-Lin в $R(lin)$

В данном разделе мы покажем, что система $R(lin)$ p-моделирует Res-Lin. Система $R(lin)$ оперирует линейными уравнениями с целыми коэффициентами и пропозициональными переменными. В данном разделе мы будем использовать знак $=$ для равенств в целых числах и знак $\underset{2}{\equiv}$ для равенств по модулю 2. Целочисленный линейный дизъюнкт — это дизъюнкция $\bigvee\limits_i(\sum\limits_j a_{i,j}x_j = b_i)$, где $a_{i,j}$ и b_j целые. Уравнения в дизъюнкте не повторяются.

Система доказательств $R(lin)$ содержит аксиомы $(x = 0) \vee (x = 1)$ для всех переменных x и следующие правила вывода:

- правило сечения позволяет получить дизъюнкты $A \vee B \vee (F_1 + F_2 = a_1 + a_2)$ и $A \vee B \vee (F_1 - F_2 = a_1 - a_2)$ из $A \vee (F_1 = a_1)$ и $B \vee (F_2 = a_2)$;
- синтаксическое ослабление позволяет получить дизъюнкт $A \vee (F = a)$ из A для любого целочисленного линейного уравнения $F = a$;
- правило упрощения позволяет получить дизъюнкт B из $B \vee (0 = c)$, где c — ненулевая константа.

Используя систему $R(lin)$, можно доказать, что множество целочисленных линейных уравнений $K = \{K_1, \ldots, K_m\}$ противоречиво. Доказательством в данной системе является последовательность целочисленных линейных дизъюнктов, которая заканчивается пустым дизъюнктом, и каждый дизъюнкт в данной последовательности является либо аксиомой, либо дизъюнктом из множества K, либо получен из предыдущих при помощи правил вывода.

Уравнение $x_1 + x_2 + \cdots + x_n \underset{2}{\equiv} 0$ будем представлять следующей дизъюнкцией целочисленных линейных уравнений: $(x_1 + x_2 + \cdots + x_n = 0) \vee (x_1 + x_2 + \cdots + x_n = 2) \vee \cdots \vee (x_1 + x_2 + \cdots + x_n = 2\lceil \frac{n}{2} \rceil)$, а уравнение $x_1 + x_2 + \cdots + x_n \underset{2}{\equiv} 1$ будем представлять следующим образом: $(x_1 + x_2 + \cdots + x_n = 1) \vee (x_1 + x_2 + \cdots + x_n = 3) \vee \cdots \vee (x_1 + x_2 + \cdots + x_n = 2\lceil \frac{n-1}{2} \rceil + 1)$.

Теорема 4.6. Система $R(lin)$ p-моделирует Res-Lin.

Перед доказательством теоремы докажем технические леммы.

Лемма 4.8. В системе $R(lin)$ возможно вывести $A \vee B \vee \bigvee\limits_{i\in[k],j\in[n]} (L_i \pm K_j)$ из $A \vee \bigvee\limits_{i=1}^{k} L_i$ и $B \vee \bigvee\limits_{j=1}^{n} K_j$, где L_i и K_j — уравнения с целыми коэффициентами. Утверждение верно для всех вариантов знака $\pm$.

Доказательство. Обозначим $C := \bigvee\limits_{i\in[k],j\in[n]} (L_i \pm K_j)$.

Применим несколько раз синтаксическое ослабление к дизъюнкту $B \vee \bigvee\limits_{j=1}^{n} K_j$ и получим дизъюнкт $A \vee B \vee C \vee \bigvee\limits_{j=1}^{n} K_j$. Теперь будем последовательно удалять лишние уравнения, начиная с последнего.

Предположим, что мы получили дизъюнкт $A' := A \vee B \vee C \vee \bigvee\limits_{j=1}^{\ell} K_j$, где $\ell \geq 1$.

Применим правило сечения к $A \vee \bigvee\limits_{i=1}^{k} L_i$ и A', таким образом мы получим дизъюнкт $B'' = A \vee B \vee C \vee \bigvee\limits_{j=1}^{\ell-1} K_j \vee \bigvee\limits_{i=1}^{k-1} L_i$, так как уравнение $L_k \pm K_\ell$ содержится в C, следовательно мы можем не писать его дважды.

Теперь применим правило сечения к B'' и A', таким образом, мы удалим последнее уравнение из B''. Применим правило сечения несколько раз, таким образом, мы получим $A'' = A \vee B \vee C \vee \bigvee\limits_{j=1}^{\ell-1} K_j$, таким образом, мы уменьшили число уравнений K_j в дизъюнкте A'. □

Лемма 4.9. Пусть $f(x) = a_1x_1 + a_2x_2 + \cdots + a_nx_n$, где $a_1, a_2, \ldots, a_n$ — натуральные числа, тогда дизъюнкт $(f(x) = 0) \vee \cdots \vee (f(x) = \sum\limits_i a_i)$ выводим в $R(lin)$.

Доказательство. Будем использовать a_i раз лемму 4.8 и получим $(a_ix_i = 0) \vee (a_ix_i = 1) \vee \ldots (a_ix_i = a_i)$ из аксиомы $(x_i = 0) \vee (x_i = 1)$. Применив лемму 4.8 для всех $i \in [n]$, получим искомый дизъюнкт. □

Доказательство теоремы 4.6. Из утверждения 4.5 следует, что достаточно промоделировать правило резолюции, правило упрощения, синтаксического ослабления и сложения в $R(lin)$. Моделирование правила упрощения и синтаксического ослабления тривиальна. Продемонстрируем моделирование остальных.

Правило резолюции можно промоделировать при помощи леммы 4.8 и правил упрощения. Действительно, для правила резолюции к $A \vee f = 1 \vee f = 3 \vee \ldots$ и $B \vee f = 0 \vee f = 2 \vee \ldots$, мы применим лемму 4.8 и получим $A \vee B \vee 0 = 1 \vee 0 = 3 \vee \ldots$, и, применив правило упрощения, получим $A \vee B$.

Для того, чтобы промоделировать правило сложения, мы докажем следующее утверждение: в системе $R(lin)$ за полиномиальное число шагов можно вывести дизъюнкт $A \vee (f \underset{2}{\equiv} \alpha) \vee (f+g \underset{2}{\equiv} \alpha+\beta+1)$ из дизъюнкта $A \vee (f \underset{2}{\equiv} \alpha) \vee (g \underset{2}{\equiv} \beta)$, где $\alpha, \beta \in \mathbb{F}_2$, а $f = \sum\limits_{i \in I} x_i$ и $g = \sum\limits_{j \in J} x_j$ — линейные формы.

Применим лемму 4.9 для получения дизъюнкта $(f \underset{2}{\equiv} \alpha) \vee (f \underset{2}{\equiv} \alpha + 1)$. Теперь применим лемму 4.8 к данному дизъюнкту и дизъюнкту $A \vee (f \underset{2}{\equiv} \alpha) \vee (g \underset{2}{\equiv} \beta)$ для получения $A \vee (f \underset{2}{\equiv} \alpha) \vee (f + g \underset{2}{\equiv} \alpha + \beta + 1)$. Если множества I и J не пересекаются, то мы получили то, что хотели.

Предположим, что $I \cap J \neq \emptyset$, тогда в уравнение $(f(x) + g(x) = 1 \mod 2)$ входят переменные с коэффициентом 2; рассмотрим одну из таких переменных — x_ℓ. Из аксиомы $(x_\ell = 0) \vee (x_\ell = 1)$ выведем дизъюнкт $(2x_\ell = 0) \vee (2x_\ell = 2)$ при помощи двух применений правила сечения; при помощи леммы 4.8 получим дизъюнкт $D := A \vee (f \underset{2}{\equiv} \alpha) \vee (f + g - 2x_\ell \underset{2}{\equiv} \alpha + \beta + 1) \vee (f - g - 2x_\ell = -1)$. Теперь удалим последнее уравнение из D. Для этого будем использовать лемму 4.9 для линейной формы $f + g - 2x_\ell$, и получим дизъюнкт C. Применим правило сечения для C и D и будем повторять применение правила сечения для результата и дизъюнкта D, пока не получим $A \vee (f \underset{2}{\equiv} \alpha) \vee (f + g - 2x_\ell \underset{2}{\equiv} \alpha + \beta + 1)$. Повторим указанный процесс для всех общих переменных f и g. $\square$

Литература

[1] Cook S. The Complexity of Theorem-Proving Procedures // Proceedings of the Third Annual ACM Symposium on Theory of Computing. — STOC '71. — New York, NY, USA : ACM, 1971. — P. 151–158.

[2] Левин Л. А. Универсальные задачи перебора // Проблемы передачи информации. — Vol. 9. — 1973. — P. 115–116.

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

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

[5] Goldreich O. Candidate One-Way Functions Based on Expander Graphs // Electronic Colloquium on Computational Complexity. — No. 00-090. — 2000.

[6] Цейтин Г. С. О сложности вывода в исчислении высказываний // Записки научных семинаров ЛОМИ. — 1968. — Vol. 8. — P. 234–259.

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

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

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

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

[11] Buss S. Towards NP-P via proof complexity and search. // Ann. Pure Appl. Logic. — 2012. — Vol. 163, no. 7. — P. 906–917.

[12] 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.

[13] Chvátal V., Szemerédi E. Many Hard Examples for Resolution // J. ACM. — 1988. — Vol. 35, no. 4. — P. 759–768.

[14] Raz R., Tzameret I. Resolution over linear equations and multilinear proofs // Ann. Pure Appl. Logic. — 2008. — Vol. 155, no. 3. — P. 194–224.

[15] Pudlák P., Impagliazzo R. A Lower Bound for DLL Algorithms for k-SAT (Preliminary Version) // Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms. — SODA '00. — Philadelphia, PA, USA : Society for Industrial and Applied Mathematics, 2000. — P. 128–136.

[16] Simon L., Le Berre D., Hirsch E. A. The SAT2002 Competition // Annals of Mathematics and Artificial Intelligence. — 2005. — Vol. 43, no. 1-4. — P. 307–342.

[17] Hirsch E. A. SAT Local Search Algorithms: Worst-Case Study. // J. Autom. Reasoning. — 2000. — Vol. 24, no. 1/2. — P. 127–143.

[18] Alekhnovich M., Ben-Sasson E. Linear Upper Bounds for Random Walk on Small Density Random 3-CNF // FOCS. — 2003. — P. 352–361.

[19] Nikolenko S. I. Hard satisfiable formulas for DPLL-type algorithms // CoRR. — 2003. — Vol. cs.CC/0301012.

[20] Achlioptas D., Beame P., Molloy M. A sharp threshold in proof complexity yields lower bounds for satisfiability search // Journal of Computer and System Sciences. — 2004. — Vol. 68, no. 2. — P. 238 – 268. — Special Issue on STOC 2001.

[21] Achlioptas D., Beame P., Molloy M. Exponential Bounds for DPLL Below the Satisfiability Threshold // Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms. — SODA '04. — Philadelphia, PA, USA : Society for Industrial and Applied Mathematics, 2004. — P. 139–140.

[22] Alekhnovich M., Hirsch E. A., Itsykson D. M. Exponential Lower Bounds for the Running Time of DPLL Algorithms on Satisfiable Formulas // J. Autom. Reason. — 2005. — Vol. 35, no. 1-3. — P. 51–72.

[23] Cook J., Etesami O., Miller R., Trevisan L. Goldreich's One-Way Function Candidate and Myopic Backtracking Algorithms // proceedings of TCC. — Springer-Verlag, 2009. — P. 521–538.

[24] Itsykson D. M. Lower bound on average-case complexity of inversion of Goldreich function by "drunken" backtracking algorithms // Proceedings of the 5th International Computer Science Symposium in Russia. — Vol. 6072 of Lecture Notes in Computer Science. — Springer, 2010. — P. 204–215.

[25] Itsykson D. M., Sokolov D. O. Lower Bounds for Myopic DPLL Algorithms with a Cut Heuristic // Algorithms and Computation / Ed. by Takao Asano, Shin-ichi Nakano, Yoshio Okamoto, Osamu Watanabe. — Springer Berlin Heidelberg, 2011. — Vol. 7074 of Lecture Notes in Computer Science. — P. 464–473.

[26] Urquhart A. The Depth of Resolution Proofs // Studia Logica. — 2011. — Vol. 99, no. 1–3. — P. 249–364.

[27] 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.

[28] Demenkov E., Kulikov A. S. An Elementary Proof of a 3n - o(n) Lower Bound on the Circuit Complexity of Affine Dispersers. // Mathematical Foundations of Computer Science 2011 / Ed. by Filip Murlak, Piotr Sankowski. — Vol. 6907 of Lecture Notes in Computer Science. — Springer, 2011. — P. 256–265.

[29] 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.

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

[31] Razborov A. A. Resolution lower bounds for perfect matching principles // Journal of Computer and System Sciences. — 2004. — Vol. 69, no. 1. — P. 3 – 27. — Special Issue on Computational Complexity 2002.

[32] Hoory S., Linial N., Wigderson A. Expander Graphs and Their Applications // Bulletin of the American Mathematical Society. — 2006. — Vol. 43. — P. 439–561.

[33] Capalbo M., Reingold O., Vadhan S., Wigderson A. Randomness conductors and constant-degree lossless expanders // Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing. — STOC '02. — New York, NY, USA : ACM, 2002. — P. 659–668.

[34] Reingold O., Vadhan S., Wigderson A. Entropy waves, the zig-zag graph product, and new constant-degree expanders. — 2002. — Vol. 155, no. 1. — P. 157–187.

[35] Høholdt T., Janwal H. Optimal Bipartite Ramanujan Graphs from Balanced Incomplete Block Designs: Their Characterizations and Applications to Expander/LDPC Codes // Proceedings of the 18th International Symposium on Applied Algebra, Algebraic Algorithms and Error-Correcting Codes. — AAECC-18 '09. — Berlin, Heidelberg : Springer-Verlag, 2009. — P. 53–64.

[36] Alon N., Schwartz O., Shapira A. An Elementary Construction of Constant-degree Expanders // Comb. Probab. Comput. — 2008. — Vol. 17, no. 3. — P. 319–327.

[37] Cryan M., Miltersen P. On Pseudorandom Generators in NC^0 // Mathematical Foundations of Computer Science. — 2001. — P. 272–284.

[38] Mossel E., Shpilka A., Trevisan L. On e-Biased Generators in NC^0 // Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science. — FOCS '03. — Washington, DC, USA : IEEE Computer Society, 2003. — P. 136–145.

[39] Applebaum B., Ishai Y., Kushilevitz E. Cryptography in NC^0 // SIAM J. Comput. — 2006. — Vol. 36, no. 4. — P. 845–888.

[40] Applebaum B., Ishai Y., Kushilevitz E. Computationally Private Randomizing Polynomials and Their Applications // Computational Complexity. — 2006. — Vol. 15, no. 2. — P. 115–162.

[41] Miller R. Goldreich's one-way function candidate and drunken backtracking algorithms. — 2009. — Distinguished Majors Thesis.

[42] Cook J., Etesami O., Miller R., Trevisan L. On the One-Way Function Candidate Proposed by Goldreich // ACM Trans. Comput. Theory. — 2014. — Vol. 6, no. 3. — P. 14:1–14:35.

[43] Ila C. Proof Complexity // European Congress of Mathematics. Port Askaig. — 2002.

[44] Reckhow R. On the Length of Proofs in the Propositional Calculus // PhD thesis. — 1976.

[45] Bogdanov A., Trevisan L. Average-case Complexity // Found. Trends Theor. Comput. Sci. — 2006. — Vol. 2, no. 1. — P. 1–106.

[46] Impagliazzo R. A Personal View of Average-Case Complexity // Structure in Complexity Theory Conference. — IEEE Computer Society, 1995. — P. 134–147.

[47] Ицыксон Д. М., Соколов Д. О. Сложность обращения явной функции Голдрейха DPLL алгоритмами // Записки научных семинаров ПОМИ. — 2012. — Vol. 399. — P. 88–108.

[48] Itsykson D. M., Sokolov D. O. The Complexity of Inversion of Explicit Goldreich's Function by DPLL Algorithms // Computer Science — Theory and Applications / Ed. by Alexander Kulikov, Nikolay Vereshchagin. — Springer Berlin Heidelberg, 2011. — Vol. 6651 of Lecture Notes in Computer Science. — P. 134–147.

[49] Alon N., Panigrahy R., Yekhanin S. Deterministic Approximation Algorithms for the Nearest Codeword Problem // Proceedings of the 12th International Workshop and 13th International Workshop on Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques. — APPROX '09 / RANDOM '09. — Berlin, Heidelberg : Springer-Verlag, 2009. — P. 339–351.

[50] Kalyanasundaram B., Schintger G. The Probabilistic Communication Complexity of Set Intersection // SIAM J. Discret. Math. — 1992. — Vol. 5, no. 4. — P. 545–557.

[51] Kushilevitz E., Nisan N. Communication Complexity. — New York, NY, USA : Cambridge University Press, 1997. — ISBN: 0-521-56067-5.