Кибернетический сборник

НОВАЯ СЕРИЯ

ВЫПУСК

21

Сборник переводов

ПОД РЕДАКЦИЕЙ
О. Б. ЛУПАНОВА

МОСКВА «МИР» 1984

ББК 32.81
К 38
УДК 519.95

Научный совет по кибернетике
Академии наук СССР

К 38 **Кибернетический сборник.** Новая серия. Вып. 21. Сб. статей: Пер. с англ. — М.: Мир, 1984, 264 с., ил.

Продолжение серии, начатой издательством «Мир» в 1965 г. В выпуске содержатся обзорные статьи и оригинальные работы известных зарубежных ученых по наиболее актуальным проблемам теоретической кибернетики. Большой интерес представляют статьи Д. Плейстида по автоматическому доказательству теорем и И. Вегенера по монотонной сложности булевых функций.

Для научных работников, инженеров-исследователей, аспирантов и студентов, занимающихся и интересующихся теоретической кибернетикой и ее приложениями.

К $\frac{1502000000-448}{041(01)-84}$ 5—84, ч. 1

ББК 32.81
519.95

Редакция литературы по математическим наукам

Нижние оценки сложности схем из функциональных элементов (обзор)

В. М. Храпченко

ВВЕДЕНИЕ

Схемы из функциональных элементов [9], известные также под названием комбинационные схемы [73], являются довольно точной моделью электронных логических схем без обратной связи, которые составной частью входят в многие устройства вычислительной техники, автоматики, телефонии и т. д. Одновременно с этим схемы из функциональных элементов представляют собой весьма содержательный объект с математической точки зрения. Поэтому их исследованию посвящено огромное число работ, которое продолжает быстро расти.

Настоящий обзор является попыткой систематизировать факты, накопленные в одном из наиболее актуальных направлений исследования схем из функциональных элементов. При работе над обзором автор существенно опирался на монографии Дж. Сэвиджа [82] и Р. Г. Нигматуллина [23], в которых можно найти подробное изложение многих результатов, упоминающихся в обзоре, и на обзорные статьи О. Б. Лупанова [11] и Р. Г. Нигматуллина [22], очень близкие по теме к данному обзору и наложившие на него свой идейный отпечаток.

Понятие схемы из функциональных элементов фактически совпадает с понятием *схемы вычисления*, которую обычно понимают как последовательность равенств:

$$\begin{aligned} z_1 &= \varphi_1(y_{11}, \ldots, y_{1r_1}),\\ &\ldots\ldots\ldots\\ z_i &= \varphi_i(y_{i1}, \ldots, y_{ir_i}),\\ &\ldots\ldots\ldots\\ z_l &= \varphi_l(y_{l1}, \ldots, y_{lr_l}), \end{aligned}$$

где все переменные $z_1, \ldots, z_l$ различны, каждая переменная y_{ij} $(i = 1, \ldots, l;\ j = 1, \ldots, r_i)$ — это либо одна из *входных* (независимых) переменных $x_1, \ldots, x_n$, либо одна из *внутренних* (зависимых) переменных $z_1, \ldots, z_{i-1}$, вычисленных на предыдущих шагах, а $\varphi_1, \ldots, \varphi_l$ — некоторые функции. При этом каждая из переменных $z_1, \ldots, z_l$ (а также каждая из переменных $x_1, \ldots, x_n$) может быть выражена в виде функции от перемен-

ных $x_1, \ldots, x_n$. Считается, что схема вычисления реализует любую такую функцию.

Если исходить из того, что в этих равенствах каждая операция φ_i $(i = 1, \ldots, l)$ выполняется функциональным элементом, то мы приходим к схеме из функциональных элементов. В этой схеме n входов, которым приписаны переменные $x_1, \ldots, x_n$, и l функциональных элементов, причем i-й элемент имеет r_i перенумерованных входов и один выход, переменная z_i, приписанная его выходу, есть функция φ_i от переменных, приписанных его входам, а соединения осуществляются так, как это диктуется равенствами. В схеме из функциональных элементов еще указываются ее *выходы*, которыми могут быть выходы элементов или входы схемы. Выходы схемы нумеруются числами $1, 2, \ldots, m$ (вообще говоря, один выход может получить несколько номеров) и считается, что схема из функциональных элементов реализует упорядоченный набор из m функций, каждая из которых выражает переменную, приписанную выходу с соответствующим номером, через переменные $x_1, \ldots, x_n$. Такой набор из m функций называется *(n, m)-функцией* или *вектор-функцией*. В случае $m = 1$ вектор-функция является функцией в обычном смысле.

Схему из функциональных элементов удобно представлять в виде ориентированного графа, в котором вершинами являются входы схемы с приписанными им переменными $x_1, \ldots, x_n$ и элементы схемы с приписанными им переменными $z_1, \ldots, z_l$ и функциями $\varphi_1, \ldots, \varphi_l$ (т. е. элемент и его выход не отделяются друг от друга). Ребра этого графа соответствуют входам элементов, а именно: для каждого входа каждого элемента схемы в графе имеется ребро с началом в той вершине, которой приписана та же переменная, что и этому входу элемента, и концом в вершине, отождествляемой с данным элементом, причем ребра нумеруются так же, как соответствующие входы элементов. Наконец, в соответствии с нумерацией выходов схемы вершинам графа приписываются числа $1, 2, \ldots, m$.

Обычно функции $\varphi_1, \ldots, \varphi_l$ можно выбирать лишь из некоторого заранее заданного множества Б, называемого базисом. Отсюда возникает понятие «схема из функциональных элементов в базисе Б (над базисом Б)».

Если некоторую вектор-функцию можно реализовать схемой в базисе Б, то, как правило, ее можно реализовать многими различными способами. Для класса $\mathcal{K}_Б$ таких вектор-функций возникает задача синтеза оптимальной (в каком-то смысле) схемы в базисе Б, реализующей заданную вектор-функцию. Одним из возможных критериев оптимальности схемы является число ее элементов (в общем случае различные элементы могут учитываться со своими весами — см. работу [9]). Приняв

этот критерий, для каждой вектор-функции f из класса $\mathcal{K}_Б$ определим *сложность* $L_Б(f)$ как минимум числа элементов в схемах над базисом Б, реализующих f, а любую схему, на которой достигается этот минимум, назовем *минимальной* (схемой в базисе Б для вектор-функции f). Теперь задача синтеза получает точную формулировку: для заданной вектор-функции f из класса $\mathcal{K}_Б$ построить минимальную схему в базисе Б.

Если базис Б является конечным, то эта задача имеет тривиальное решение, которое состоит в следующем. Сначала строятся все схемы в базисе Б, содержащие 1 элемент, и проверяется, есть ли среди них схема, реализующая f. Затем строятся все схемы в базисе Б, содержащие 2 элемента, и делается та же проверка и т. д. до тех пор, пока не встретится схема, реализующая f. Совершенно ясно, что она и является минимальной.

Такое решение является безукоризненным, если подходить к задаче синтеза формально. Реально воспользоваться им чаще всего невозможно. Дело в том, что с ростом числа элементов в схемах количество схем растет очень быстро. В невырожденном случае, когда базис Б содержит функцию от двух или более переменных, число схем в базисе Б, состоящих из l элементов, не меньше $(l!)^2$ (чуть медленнее растет число неизоморфных схем, но это уже совсем другой вопрос). Поэтому если сложность вектор-функции f превышает небольшой порог (величиной всего в несколько единиц), то тривиальный метод решения становится практически неосуществимым, причем использование самых быстрых вычислительных машин лишь несущественно расширяет границы его применимости (увеличение скорости в 100 раз в лучшем случае лишь на единицу увеличивает число элементов в схемах, которые еще можно просмотреть).

Такая большая трудоемкость объясняется не малой эффективностью тривиального алгоритма, а трудностью решения задачи синтеза в общем виде и присуща всем алгоритмам, предназначенным для ее решения, — к этому выводу одним из первых пришел С. В. Яблонский в своей работе [48], рассматривая данную задачу для контактных схем. С тех пор эта точка зрения стала общепринятой, получив много косвенных подтверждений своей справедливости.

В этих условиях бессмысленно продолжать пытаться решать задачу синтеза в общем виде. Выход состоит в том, чтобы перейти к рассмотрению конкретных наиболее интересных функций (например, часто встречающихся в математике или технике) или, скорее, классов функций и решать задачу синтеза именно для них. Кроме того, задачу можно несколько ослабить, не требуя, чтобы построенная схема была непременно минимальной, считая приемлемой достаточно близкую к ней схему.

При этом задача синтеза естественным образом распадается на две в определенной степени самостоятельные задачи:

1) построение для заданной конкретной вектор-функции f схемы, содержащей возможно меньше элементов, и получение тем самым верхней оценки для $L_{\text{Б}}(f)$;

2) получение возможно лучшей нижней оценки для $L_{\text{Б}}(f)$, которая подтверждала бы, что построенная схема достаточно близка к минимальной.

Среди этих задач первая обычно бывает легче, так как для ее решения достаточно в конце концов построить одну схему, в то время как для решения второй требуется в каком-то смысле просмотр всех схем, реализующих вектор-функцию f. По этой причине возникло целое направление, связанное с поиском методов получения нижних оценок сложности схем для конкретных вектор-функций.

Этому направлению и посвящен обзор. В нем рассматривается главным образом двузначный случай, когда переменные принимают лишь два значения 0 и 1, а базис Б состоит из булевых функций. Ставилось целью в двузначном случае как можно полнее охватить методы получения нижних оценок сложности, причем не только для схем произвольного вида, но и для схем с некоторыми, правда, лишь наиболее естественными ограничениями. Насколько это удалось, судить читателям.

Что касается применения конкретного метода к различным функциям, то здесь возможности настолько велики, что все такие результаты перечислить довольно трудно, да и, пожалуй, нет смысла. Тем не менее многие из них в обзоре упоминаются. Верхние оценки сложности схем в обзоре приводятся лишь в виде исключения, если они служат лучшему пониманию метода.

Нижние оценки в общем случае

Наиболее универсальными, но зато и наиболее слабыми являются следующие две нижние оценки, которые доказываются очень просто и поэтому считаются тривиальными.

Первая относится к (n, m)-функции, у которой все компоненты $f_1, \ldots, f_m$ различны и отличны от $x_1, \ldots, x_n$. Она имеет вид

$$L_{\text{Б}}(f_1, \ldots, f_m) \geqslant \min_{1 \leqslant i \leqslant m} L_{\text{Б}}(f_i) + m - 1.$$

Вторая оценка относится к функции f, существенно зависящей от n переменных, и имеет вид

$$L_{\text{Б}}(f) \geqslant \frac{n-1}{r-1},$$

где r — максимум числа переменных у функций базиса Б.

Первая из этих оценок, по-видимому, не требует пояснений, а вторую поясним следующим образом: для функции, существенно зависящей от n переменных, схема (а по-другому граф) содержит по крайней мере по одной цепи, соединяющей каждый из n ее входов с выходом, при этом в подграфе, образованном этими цепями, в каждом элементе сходится не более r цепей, а выходит из такого элемента (если он не является для схемы выходным) не менее 1 цепи.

Перейдем к рассмотрению двузначного случая. Будем считать, что базис является конечным и полным, т. е. позволяет реализовать любую булеву функцию. Судя по всему, это самый естественный класс базисов.

Оказывается, что сложность реализации функции (или вектор-функции) не так уж сильно зависит от базиса. Моделируя элементы одного базиса схемами в другом базисе, Д. Мюллер показал [73], что

для любых двух конечных полных базисов Б и Б′ *существуют такие положительные константы* c_1 *и* c_2, *что для любой булевой функции* (*вектор-функции*) f *выполняются неравенства*

$$c_1 L_{\text{Б}}(f) \leqslant L_{\text{Б}'}(f) \leqslant c_2 L_{\text{Б}}(f).$$

Таким образом, оценки для сложности функций в разных базисах отличаются лишь постоянными множителями. В связи с этим первоочередной задачей является определение главного множителя оценки — того, который не зависит от базиса. Под этим подразумевается, что вместо одной конкретной булевой функции рассматривается последовательность «аналогичных» ей функций, содержащая при каждом n чаще всего по одной функции от n переменных, а оценка ищется в виде произведения коэффициента, зависящего от базиса, на функцию от n, которая и представляет собой главный множитель.

Методом Риордана — Шеннона [81] Д. Мюллер показал [73], что существует последовательность булевых функций, для которой главный множитель в нижней оценке сложности растет как $2^n/n$, т. е. существуют очень сложные булевы функции (правда, неизвестно, как их явно описать). К сожалению, все нижние оценки, которые были до сих пор получены для конкретных функций (точнее, для последовательностей функций), являются линейными относительно n, и поэтому о поведении главного множителя они не говорят ничего нового по сравнению со второй тривиальной нижней оценкой.

Эти нижние оценки невозможно рассматривать в отрыве от базиса. Нам потребуются главным образом следующие четыре базиса: $\text{Б}_0 = \{\&, \vee, \bar{}\}$; Б_1 — базис, состоящий из всех не более чем двуместных булевых функций, кроме $\oplus$ — суммы по

модулю 2 и $\sim$ — ее отрицания; $Б_2$ — базис, состоящий из всех не более чем двуместных булевых функций; $Б_Ш$ — базис, состоящий из одной функции — штриха Шеффера $x/y = \bar{x} \vee \bar{y}$. Для каждого из этих базисов $r = 2$, и вторая тривиальная нижняя оценка принимает вид

$$L_Б(f) \geqslant n - 1,$$

где f — функция, существенно зависящая от n переменных.

Нетривиальные нижние оценки сложности для схем из функциональных элементов появились уже после того, как такие оценки были получены для контактных схем, для формул и для некоторых других объектов. Каждая из этих ранних оценок была получена для одной конкретной функции (т. е. последовательности функций), и складывалось впечатление, что нижние оценки для разных функций надо получать обязательно по-разному. Только позднее стало понятным, что в доказательстве почти каждой из полученных нижних оценок заложен метод, применимый к целому ряду функций. Сейчас дело обстоит именно так для большинства нижних оценок. При этом процесс получения нижней оценки можно разбить на два этапа:

1) выявление определенного свойства булевых функций, в которое обычно входит ряд параметров функции, и получение неравенства, оценивающего снизу сложность функции через эти параметры;

2) определение параметров конкретной булевой функции и подстановка их в неравенство, полученное на первом этапе.

Фактически первый этап — это изобретение метода получения нижних оценок, а второй этап — это применение данного метода к конкретной функции. Чаще всего второй этап не такой трудный, как первый, но и он редко бывает тривиальным. В обзоре больше внимания уделено методам, а их применение к конкретным функциям будет рассматриваться лишь для более сильных методов.

Можно ожидать (см., например, [48]), что более сложная функция, как правило, имеет больше *подфункций*, т. е. функций, получающихся из нее в результате подстановки констант вместо переменных. Поэтому естественно классифицировать булевы функции по числу имеющихся у них подфункций. Следуя [60, 82], введем семейство классов булевых функций, имеющих заданное число подфункций заданного вида. А именно, обозначим через $P^n_{p,q}$ класс булевых функций от n переменных, каждая из которых при подстановке вместо любых p ее переменных всеми возможными способами констант порождает не менее q различных подфункций. Очевидно, что для непустого класса $P^n_{p,q}$ справедливы соотношения $q \leqslant 2^p$ и $q \leqslant 2^{2^{n-p}}$. Лег-

ко проверить, что $P^n_{0,1}$ — это класс всех булевых функций от n переменных, а $P^n_{1,2}$ — это класс булевых функций, существенно зависящих от всех своих n переменных. Таким образом, вторая тривиальная нижняя оценка — это оценка для функций из класса $P^n_{1,2}$.

Нетрудно убедиться в том, что при $p \geqslant 1$

$$P^n_{p,\,2^{p-1}+1} \supseteq P^n_{p+1,\,2^p+1}.$$

Отсюда следует, что

$$P^n_{1,2} \supseteq P^n_{2,3} \supseteq P^n_{3,5} \supseteq \ldots \supseteq P^n_{p,\,2^{p-1}+1} \supseteq \ldots .$$

В работе [62] показано, что если $f \in P^n_{2,3}$, то

$$L_{Б_2}(f) \geqslant n,$$

причем эта оценка неулучшаема. Для ее доказательства достаточно заметить, что у элементов минимальной схемы в базисе $Б_2$ по 2 входа ($n > 1$) и что одна из переменных, поступающих на первый элемент, должна поступать еще на один элемент.

Здесь еще, пожалуй, нельзя говорить о продвижении по сравнению с тривиальной нижней оценкой. Правда, приведенная оценка не была первой, полученной в данном направлении.

По-видимому, первые нетривиальные нижние оценки для сложности схем произвольного вида (т. е. без ограничений) получили в 1963—1965 гг. В. А. Малышев и Б. М. Клосс [5] (судя по указанию, имеющемуся в этой работе, вошедшие в нее нижние оценки принадлежат уже Б. М. Клоссу). Здесь рассмотрен класс функций, каждая из которых при подстановке констант вместо любых двух ее переменных порождает различные подфункции при подстановках $(0,0)$ и $(0,1)$, $(0,1)$ и $(1,1)$, $(0,0)$ и $(1,1)$. Такие функции были названы парно разделимыми. Класс парно разделимых функций от n переменных у́же, чем $P^n_{2,3}$, но шире, чем $P^n_{2,4}$. Мы его условно обозначим через $P^n_{2,\,3-4}$.

Б. М. Клосс показал [5], что если $f \in P^n_{2,\,3-4}$, то

$$L_{Б_2}(f) \geqslant \frac{10\,(n-1)}{9}.$$

Эта оценка доказывается уже непросто. Ее доказательство опирается на то, что минимальная схема для парно разделимой функции не может содержать некоторые конкретные фрагменты из двух-трех элементов. Этого оказывается достаточно для того, чтобы, разбив элементы и входы схемы на несколько сортов, составить соотношения (главным образом неравенства) между

числами элементов и входов разных сортов, а затем извлечь из этих соотношений ранее приведенную нижнюю оценку.

Примерно в таком же стиле была доказана следующая нижняя оценка [62]: если $f \in P^n_{3,5}$, то

$$L_{Б_2}(f) \geqslant \frac{11n-14}{10}.$$

Вскоре ее усилил В. Хси [66] (см. также [60]), доказав, что если $f \in P^n_{3,5}$, то

$$L_{Б_2}(f) \geqslant \frac{7n-4}{6}.$$

Эта оценка весьма близка к окончательной (множитель при n не может быть увеличен больше, чем на 0.01) [66] и доказывается уже трудно, хотя в общем в том же стиле, что и две предыдущие оценки. Здесь составляется больше неравенств, они сложнее, и, чтобы извлечь из них нижнюю оценку, приходится обращаться к теореме двойственности линейного программирования — другого способа перейти от полученных неравенств к оценке не видно.

Это наводит на мысль, что в рамках понятия класса $P^n_{p,q}$ трудно рассчитывать на получение высоких нижних оценок. Еще больше убеждает в этом пример функции f^*, построенной Сэвиджем [82] (как он отмечает), по аналогии с функцией из неопубликованной работы А. Мейера и М. Патерсона (по-видимому, здесь есть влияние и примера Э. И. Нечипорука [19]). В монографии [82] показано, что

$$f^* \in P^n_{p^*,2^{p^*}}, \quad \text{где} \quad p^* = \lfloor n/2(\log_2 n + 1) \rfloor - 1$$

и вместе с тем

$$L_{Б_2}(f^*) \leqslant cn,$$

где c — некоторая (сравнительно небольшая) константа.[1]) Нетрудно заметить, что при любом $p \leqslant p^*$ и любом $q \leqslant 2^p$

$$P^n_{p,q} \supseteq P^n_{p^*,2^{p^*}}.$$

Поэтому все непустые классы $P^n_{p,q}$ при $p = 1, \ldots, p^*$ содержат функцию f^*, а значит, опираясь на понятие класса $P^n_{p,q}$, при $p \leqslant p^*$ можно получать лишь линейные (относительно n) нижние оценки сложности.

Итак, наличие у функции большого числа подфункций (даже при таком «равномерном распределении» их, как в функции f^*)

[1]) Запись $\lfloor x \rfloor$ обозначает наибольшее целое, не превосходящее x, а запись $\lceil x \rceil$ — наименьшее целое, не меньшее, чем x.

еще не гарантирует большую сложность функции. Д. Улиг построил [42] еще более впечатляющий пример функции, которая имеет асимптотически максимально возможное число подфункций и тем не менее реализуется с линейной сложностью. Правда, в этой функции подфункции распределены не так равномерно. Все-таки сложная функция должна иметь много подфункций, однако, как мы видели, для получения высокой нижней оценки ее сложности этой информации недостаточно, чем-то она должна быть дополнена.

В духе работ [5, 82] введем еще два семейства классов булевых функций, которые сохраняют свойство принадлежать классу вида $P^n_{p,\,q}$ при подстановке констант вместо переменных. Условимся через $f|_{x_i=\sigma}$ обозначать подфункцию функции f, получающуюся при подстановке в нее вместо переменной x_i константы σ. Новые классы $Q^{n,\,k}_{p,\,q}$ и $R^{n,k}_{p,q}$ определим индуктивно. Для этого положим

$$Q^{n,\,0}_{p,\,q} = R^{n,\,0}_{p,\,q} = P^n_{p,\,q},$$

а при $k \geqslant 1$ класс $Q^{n,\,k}_{p,\,q}$ (соответственно класс $R^{n,\,k}_{p,\,q}$) определим как множество таких булевых функций f от n переменных, что

1) $f \in P^n_{p,\,q}$;

2) для любой переменной x_i, $i = 1, \ldots, n$, при некоторой (соответственно при любой) константе $\sigma_i \in \{0, 1\}$

$$f|_{x_i=\sigma_i} \in Q^{n-1,\,k-1}_{p,\,q}$$

$\left(\text{соответственно } f|_{x_i=\sigma_i} \in R^{n-1,\,k-1}_{p,\,q}\right)$.

Очевидно, что

$$P^n_{p,\,q} \supseteq Q^{n,\,k}_{p,\,q} \supseteq R^{n,\,k}_{p,\,q}$$

и

$$Q^{n,\,0}_{p,\,q} \supseteq Q^{n,\,1}_{p,\,q} \supseteq \ldots \supseteq Q^{n,\,k}_{p,\,q},$$
$$R^{n,\,0}_{p,\,q} \supseteq R^{n,\,1}_{p,\,q} \supseteq \ldots \supseteq R^{n,\,k}_{p,\,q}.$$

Трудно сказать, позволит ли рассмотрение этих классов получать «нелинейные» нижние оценки сложности, т. е. оценки, растущие быстрее линейных. Во всяком случае более высокие линейные нижние оценки на этом пути были получены.

Б. М. Клосс доказал [5], что если $f \in Q^{n,\,k}_{2,\,3}$, то

$$L_{Б_2}(f) \geqslant n + k$$

(на самом деле Б. М. Клосс рассматривал функции из класса $Q_{2,\,3-4}^{n,\,k}$, но его доказательство сохраняется и для класса $Q_{2,\,3}^{n,\,k}$). Это тоже одна из первых нетривиальных нижних оценок сложности, но уже более высокая и имеющая при этом простое доказательство. Дело в том, что сохранение некоторого свойства позволяет вести доказательство нижней оценки по индукции. В данном случае в схеме для функции $f \in Q_{2,\,3}^{n,\,k} \subseteq P_{2,\,3}^{n}$ есть переменная, поступающая на входы хотя бы двух элементов. Вместо нее можно подставить такую константу, что получится схема для функции из класса $Q_{2,\,3}^{n-1,\,k-1}$, причем из этой схемы можно удалить по крайней мере два элемента, имеющие на входе константу, и изменить «соседние» с ними элементы так, чтобы функционирование схемы не нарушилось. Так как схема для функции из класса $Q_{2,\,3}^{n-k,\,0} = P_{2,\,3}^{n-k}$ содержит не менее $n-k$ элементов, отсюда следует оценка Клосса.

Как отмечалось выше, оценка такого вида — это уже метод. Покажем, как применяется метод Клосса к симметрическим функциям. Напомним, что функция называется *симметрической*, если она не изменяется при перестановке своих переменных. Ясно, что симметрическая булева функция принимает одинаковые значения на наборах значений переменных, содержащих одинаковое число единиц. Поэтому симметрическую булеву функцию можно задавать перечислением ее *рабочих чисел*, т. е. чисел единиц в тех наборах, на которых она равна 1. Симметрическая функция от n переменных с рабочими числами $a_1, \ldots, a_m$ обозначается через $S_n^{a_1, \cdots, a_m}$. Нетрудно проверить, что, например, такие симметрические функции, как $S_n^{2,\,3,\,6,\,7,\,\ldots}$ или монотонная симметрическая функция $S_n^{m,\,m+1,\,\ldots,\,n}$ при условии, что $2 \leqslant m \leqslant n-1$, принадлежат классу $Q_{2,\,3}^{n,\,n-3}$. Поэтому

$$L_{Б_2}(S_n^{2,\,3,\,6,\,7,\,\cdots}) \geqslant 2n-3,$$

$$L_{Б_2}(S_n^{m,\,m+1,\,\ldots,\,n}) \geqslant 2n-3 \ (2 \leqslant m \leqslant n-1).$$

Можно показать, что кроме констант 0 и 1, имеющих в базисе $Б_2$ сложность 1, и функций

$$S_n^0(x_1, \ldots, x_n) = \bar{x}_1 \,\&\, \ldots \,\&\, \bar{x}_n,$$

$$S_n^n(x_1, \ldots, x_n) = x_1 \,\&\, \ldots \,\&\, x_n,$$

$$S_n^{1,\,3,\,5,\,\cdots}(x_1, \ldots, x_n) = x_1 \oplus \ldots \oplus x_n,$$

а также их отрицаний, имеющих в базисе $Б_2$ сложность $n-1$,

существуют еще только две симметрические функции:

$$S_n^{0,\,n}(x_1, \ldots, x_n) = \bar{x}_1 \ldots \bar{x}_n \vee x_1 \ldots x_n \quad (n \geqslant 4)$$

и ее отрицание, не принадлежащие классу $Q_{2,\,3}^{n,\,n-3}$. Поэтому для любой симметрической функции S_n, кроме десяти перечисленных выше, справедлива оценка

$$L_{Б_2}(S_n) \geqslant 2n - 3.$$

Другим способом К. Шнорр доказал [84], что

$$L_{Б_2}(S_n^{0,\,n}) \geqslant 2n - 3,$$

причем эта нижняя оценка совпала с верхней и тем самым дала точное значение сложности симметрической функции $S_n^{0,\,n}$. В данном случае нижняя оценка — это «метод» только для одной функции (одной последовательности функций). Тем не менее и она применима к целому ряду функций. Например, пользуясь тем, что функция

$$\overline{S_n^{0,\,n}} = S_n^{1,\,2,\,\ldots,\,n-1}$$

является не только отрицанием функции $S_n^{0,\,n}$, но и двойственной к ней функцией, а также тем, что базис $Б_2$ содержит вместе с любой функцией двойственную ей функцию, мы приходим к выводу, что

$$L_{Б_2}(\overline{S_n^{0,\,n}}) = L_{Б_2}(S_n^{0,\,n}) \geqslant 2n - 3$$

(подобно тому, как выше, эта оценка дает точное значение сложности симметрической функции $\overline{S_n^{0,\,n}}$). Многочисленные примеры применения «метода одной функции» можно построить, рассматривая близкие к ней по сложности функции. Естественные примеры такого рода нам еще встретятся.

Последний пример нас несколько увел в сторону. Возвращаясь к классам $Q_{p,\,q}^{n,\,k}$, можно добавить только следующее. Если больше использовать специфику базиса $Б_2$ и симметричность функции, то для многих симметрических функций приведенную выше оценку можно усилить. Об этом речь пойдет ниже. А здесь пора подвести итог: классы $Q_{p,\,q}^{n,\,k}$ исследованы пока мало.

Еще меньше исследованы классы $R_{p,\,q}^{n,\,k}$. Исключение составляет самый простой класс $R_{1,\,2}^{n,\,n-1}$. Легко показать, что он состоит из двух линейных функций:

$$S_n^{1,\,3,\,5,\,\cdots}(x_1, \ldots, x_n) = x_1 \oplus \ldots \oplus x_n$$

и ее отрицания. Введем сокращенное обозначение

$$\Lambda_n(x_1, \ldots, x_n) = x_1 \oplus \ldots \oplus x_n.$$

Как уже отмечалось,

$$L_{Б_2}(\Lambda_n) = L_{Б_2}(\overline{\Lambda}_n) = n - 1.$$

Рассмотрим реализацию функций Λ_n и $\overline{\Lambda}_n$ в других базисах.

К. Шнорр показал [83], что

$$L_{Б_1}(\Lambda_n) = L_{Б_1}(\overline{\Lambda}_n) = 3(n - 1)$$

(напомним, что $Б_1 = Б_2 \setminus \{\oplus, \sim\}$). Как верхняя, так и нижняя оценки устанавливаются здесь очень просто. Доказательство нижней оценки опирается на то, что для любой функции $\varphi(x_1, x_2)$ из базиса $Б_1$ существуют определяющие значения переменных, т. е. такие константы $\sigma_1, \sigma_2 \in \{0, 1\}$, что константами являются $\varphi(\sigma_1, x_2)$ и $\varphi(x_1, \sigma_2)$. Из этого свойства и свойства функции $\Lambda_n(\overline{\Lambda}_n)$ сохранять при подстановке константы вместо переменной существенную зависимость от всех остальных переменных вытекает, что в схеме для функции $\Lambda_n(\overline{\Lambda}_n)$ обе переменные, поступающие на входы первого элемента схемы, поступают еще на какие-нибудь элементы. Подставляя вместо любой из этих переменных определяющее значение для функции какого-нибудь элемента, на который она поступает, мы получим схему для функции Λ_{n-1}, либо для функции $\overline{\Lambda}_{n-1}$, причем из нее можно удалить по крайней мере три элемента (третий тот, на который константа поступает с выхода элемента). Поскольку $\Lambda_n, \overline{\Lambda}_n \in R_{1,2}^{n,n-1}$, это можно проделать $n - 1$ раз.

Перейдем к базису $Б_0 = \{\&, \vee, {}^-\}$. Н. П. Редькин доказал [31], что

$$L_{Б_0}(\Lambda_n) = L_{Б_0}(\overline{\Lambda}_n) = 4(n - 1).$$

Здесь нижняя оценка доказывается уже нелегко. Трудность состоит в необходимости рассмотреть много различных случаев. Однако свойства, на которые опирается доказательство, те же, что и для базиса $Б_1$. Впрочем, не следует забывать, что Н. П. Редькин получил оценку раньше К. Шнорра.

В этой же работе [31] найдена сложность реализации в базисе $Б_0$ двух более простых функций, но нижние оценки для них доказываются ничуть не легче. Наконец, в этой работе приведено еще два результата,

$$L_{\{\&, \,^-\}}(\Lambda_n) = L_{\{\&, \,^-\}}(\overline{\Lambda}_n) = 7(n - 1),$$

$$L_{\{\vee, \,^-\}}(\Lambda_n) = L_{\{\vee, \,^-\}}(\overline{\Lambda}_n) = 7(n - 1),$$

которые, по словам автора, получаются аналогично.

Рассмотрим теперь базис $Б_{Ш}$, состоящий из одной функции — штриха Шеффера. Н. П. Редькин доказал [32], что

$$L_{Б_{Ш}}(\Lambda_n) = 4(n-1).$$

Здесь нижняя оценка доказывается, пожалуй, еще труднее, но примерно в том же стиле, что и для базиса $Б_0$. Можно отметить один прием, состоящий во временном расширении базиса путем введения в него операции отрицания $^-$ и в способе подсчета сложности, при котором инверторы (т. е. элементы, реализующие отрицание) учитываются с коэффициентом 0.5.

В этом же базисе $Б_{Ш}$ были получены одни из первых нетривиальных нижних оценок сложности. Они принадлежат Е. П. Сопруненко, которая показала [36], что для конъюнкции $K_n = x_1 \& \ldots \& x_n$ и дизъюнкции $D_n = x_1 \vee \ldots \vee x_n$ справедливы соотношения

$$L_{Б_{Ш}}(K_n) = 2n - 2,$$

$$L_{Б_{Ш}}(D_n) = 3n - 3.$$

Впоследствии эти результаты были обобщены Е. С. Гореликом, который к тому же нашел гораздо более простое доказательство. Пусть $\tilde{\sigma}^n = (\sigma_1, \ldots, \sigma_n)$ — набор из нулей и единиц, $\|\tilde{\sigma}^n\|$ — число единиц в наборе $\tilde{\sigma}^n$, а

$$K_{\tilde{\sigma}^n} = x_1^{\sigma_1} \& \ldots \& x_n^{\sigma_n} \quad \text{и} \quad D_{\tilde{\sigma}^n} = x_1^{\sigma_1} \vee \ldots \vee x_n^{\sigma_n}.$$

Е. Е. Горелик доказал [3], что

$$L_{Б_{Ш}}(K_{\tilde{\sigma}^n}) = 3n - 2 - \|\tilde{\sigma}^n\|,$$

$$L_{Б_{Ш}}(D_{\tilde{\sigma}^n}) = 2n - 3 + \|\tilde{\sigma}^n\| \quad (n \geqslant 2).$$

Здесь нижние оценки доказываются в три этапа. Сначала получается нижняя оценка для $K_{\tilde{1}^n} = K_n$, равная $2n - 2$. Она доказывается путем приведения произвольной минимальной схемы для $K_{\tilde{1}^n}$ к некоторому стандартному виду. На этом этапе используется преобразование, при котором на входе одного из элементов схемы переменная (входная или внутренняя) заменяется другой переменной, т. е. вход элемента «переключается» с выхода одного элемента (или входа схемы) на другой, причем преобразование выполняется так, чтобы не нарушилось функционирование схемы. По-видимому, здесь впервые для получения нижней оценки применяется подстановка настолько общего вида.

Второй этап во многом аналогичен первому — на нем получаются нижние оценки для всех конъюнкций $K_{\tilde{\alpha}^n}$.

Третий этап — это иллюстрация метода одной функции. Заметив, что

$$\overline{D_{\tilde{\sigma}^n}} = \overline{x_1^{\sigma_1} \vee \ldots \vee x_n^{\sigma_n}} = x_1^{\bar{\sigma}_1} \,\&\, \ldots \,\&\, x_n^{\bar{\sigma}_n} = K_{\tilde{\bar{\sigma}}^n}$$

(мы положили $\tilde{\bar{\sigma}}^n = (\bar{\sigma}_1, \ldots, \bar{\sigma}_n)$), добавим в схему для дизъюнкции $D_{\tilde{\sigma}^n}$ один элемент и на оба его входа подадим $D_{\tilde{\sigma}^n}$. При этом получится схема для конъюнкции $K_{\tilde{\bar{\sigma}}^n}$, откуда сразу следует, что

$$L_{\text{Б}_\text{Ш}}(D_{\tilde{\sigma}^n}) + 1 \geqslant L_{\text{Б}_\text{Ш}}(K_{\tilde{\bar{\sigma}}^n})$$

или

$$L_{\text{Б}_\text{Ш}}(D_{\tilde{\sigma}^n}) \geqslant L_{\text{Б}_\text{Ш}}(K_{\tilde{\bar{\sigma}}^n}) - 1 \geqslant 3n - 3 - \|\tilde{\bar{\sigma}}^n\| =$$

$$= 3n - 3 - (n - \|\tilde{\sigma}^n\|) = 2n - 3 + \|\tilde{\sigma}^n\|.$$

Так нижняя оценка, полученная для одной функции, позволяет получить нижнюю оценку для другой функции, близкой к ней по сложности.

Перенесение этих результатов на базисы, близкие к $\text{Б}_\text{Ш}$, можно найти в работах [24, 25].

Перейдем к рассмотрению наиболее сильных нижних оценок сложности, полученных для базиса Б_2. В них уже в большей степени используется специфика этого базиса, хотя по-прежнему многое определяется свойствами реализуемой функции. Можно выделить два основных приема, которые впервые были применены В. Паулем [77] и К. Шнорром [84].

Первый из этих приемов в самом простом своем варианте состоит в том, что вместо некоторой переменной x_i, поступающей на вход элемента $\oplus$ или элемента $\sim$, подставляется функция, поступающая на второй вход этого элемента (при условии, что она не зависит от x_i). Это позволяет удалить из схемы данный элемент и элемент, следующий за ним (так как теперь на него поступает константа). Если после такого преобразования схема реализует функцию с тем же свойством, что и раньше, то возможен индуктивный переход. Эта подстановка для линейных элементов обычно применяется в сочетании с подстановкой определяющего значения для нелинейных элементов, однако ее применение гораздо сложнее, поскольку функция, подставляемая вместо переменной, вообще говоря, неизвестна (отметим определенную аналогию с переключением входа элемента у Е. С. Горелика [3], хотя там, конечно, положение облегчается тем, что базис значительно уже).

Второй прием связан с учетом возможных разветвлений цепей, идущих от входов схемы к ее выходу. Грубо говоря, каж-

дое разветвление увеличивает хотя бы на единицу число элементов, в которых должны сходиться цепи (это верно для любого базиса, содержащего не более чем двуместные функции).

Пользуясь этими двумя приемами и применяя еще одно оригинальное преобразование схемы (которое пока еще не нашло других применений), К. Шнорр получил ту нижнюю оценку сложности симметрической функции $S_n^{0,n}$, которая была приведена ранее.

В работе В. Пауля [77] оба приема встречаются как в самых простых своих вариантах, так и в более общем виде. Остановимся только на том, как обобщается первый из них.

Прием Пауля. Пусть переменная x_i поступает на вход элемента $\oplus$ или $\sim$. Если выход этого элемента не разветвляется и поступает снова на вход элемента $\oplus$ или $\sim$, рассматриваем его выход и т. д., строя цепь из элементов $\oplus$ и $\sim$ до тех пор, пока не встретится разветвление или элемент, отличный от $\oplus$ и $\sim$. Пусть эта цепь состоит из k элементов и $g_1, \ldots, g_k$ — функции, поступающие на их «вторые» входы (требуется, чтобы $g_1, \ldots, g_k$ не зависели от x_i). Тогда существует такая константа $\sigma \in \{0, 1\}$, что, подставляя вместо x_i функцию $g_1 \oplus \ldots \oplus g_k \oplus \sigma$ и тем самым вводя в схему $k-1$ элементов, мы можем удалить из нее k элементов построенной цепи и 2 элемента после цепи, т. е. уменьшить число элементов схемы на 3 (конечно, важно, чтобы получившаяся схема реализовала подходящую функцию).

Казалось бы, недалеко до получения нижней оценки сложности вида $3n - O(1)$ в базисе $Б_2$. Однако не так просто подобрать функцию, свойства которой мало изменяются при подстановке вместо переменной заранее неизвестной функции. В. Пауль построил функцию f от $n + 2\log_2 n + 1$ переменных (мы ее еще рассмотрим), у которой данному условию удовлетворяют n переменных. Тем не менее при получении нижней оценки из-за неравноправия переменных пришлось рассматривать дополнительный случай, в котором применяется уже второй прием (тоже в обобщенном виде). В результате В. Пауль получил оценку [77]

$$L_{Б_2}(f) \geqslant 2{,}5n - 2.$$

Прием Пауля остроумно применил к симметрическим функциям Л. Стокмейер. В схеме для симметрической функции от n переменных ему удалось сразу вместо двух переменных подставить неизвестную функцию и ее отрицание и при этом удалить из схемы 5 элементов. Ясно, что получившаяся схема снова реализует симметрическую функцию, но уже от $n-2$ переменных. На этом пути (не прибегая ко второму приему) Л. Стокмейер доказал [90], что для любой симметрической функции

S_n, удовлетворяющей условию

$$S_n(x_1, \ldots, x_{n-2k}, \underbrace{0, \ldots, 0}_{k}, \underbrace{1, \ldots, 1}_{k}) \in Q_{2,3}^{n-2k,\ n-2k-3},$$

справедлива оценка

$$L_{Б_2}(S_n) \geqslant 2n + k - 3.$$

Отметим, что данное условие легко проверить, зная рабочие числа симметрической функции. Так, например, для симметрической функции $S_n^{2,\ 3,\ 6,\ 7,\ \cdots}$ можно положить $k = \lfloor (n-3)/2 \rfloor$ и получить оценку

$$L_{Б_2}(S_n^{2,\ 3,\ 6,\ 7,\ \cdots}) \geqslant 2.5n - 5.$$

Вообще Л. Стокмейер показал [90], что для почти всех симметрических функций S_n от n переменных

$$L_{Б_2}(S_n) \gtrsim 2.5n.$$

Прием Пауля использовал также Н. П. Редькин, доказавший в работе [33] следующее соотношение для $(n/2)$-разрядного параллельного сумматора (n четное):

$$L_{Б_2}(\Sigma_{n/2}) = 2.5n - 3.$$

Здесь $\Sigma_{n/2}$ — это $(n, (n/2)+1)$-функция, выражающая цифры суммы двух $(n/2)$-разрядных слагаемых через цифры этих слагаемых.

Рассмотрим теперь функцию f, предложенную В. Паулем [77]. Ее переменные разбиты на четыре группы: первая и вторая группы содержат по $m = \log_2 n$ переменных (т. е. n — натуральная степень двойки), третья группа состоит из одной переменной, а четвертая группа содержит n переменных. Набор значений переменных первой группы служит адресом (номером), по которому выбирается одна из переменных четвертой группы, а набор значений переменных второй группы — адресом (номером), по которому выбирается другая переменная из четвертой группы, наконец, q — переменная третьей группы; она определяет, какая операция выполняется над выбранными переменными: & или $\oplus$. Формально эта функция задается следующим образом:

$$f(y_1, \ldots, y_m, z_1, \ldots, z_m, q, x_0, \ldots, x_{n-1}) =$$

$$= \begin{cases} x_{|\tilde{y}|} \,\&\, x_{|\tilde{z}|}, & \text{если} \quad q = 1, \\ x_{|\tilde{y}|} \oplus x_{|\tilde{z}|}, & \text{если} \quad q = 0, \end{cases}$$

где $\tilde{y} = (y_1, \ldots, y_m)$, $\tilde{z} = (z_1, \ldots, z_m)$, а $|\tilde{y}|$ и $|\tilde{z}|$ — числа, двоичной записью которых являются соответственно $\tilde{y}$ и $\tilde{z}$.

Функция f при подстановке вместо любой переменной x_i, $i = 0, \ldots, n-1$, произвольной функции от остальных переменных может изменить свои значения лишь при $|\tilde{y}| = i$ или $|\tilde{z}| = i$, т. е. для всех пар переменных (x_j, x_k), где $j \neq i$, $k \neq i$, ее свойства сохраняются. Всего можно выполнить $n-2$ подстановок, после которых какое-то свойство еще будет сохраняться.

При получении нижней оценки В. Паулю пришлось рассмотреть несколько случаев в соответствии с тем, на какие элементы могут поступать переменные четвертой группы. Наибольшие затруднения вызвал случай, когда каждая из этих переменных поступает только на один элемент, и этот элемент нелинейный. Данный случай рассматривался с учетом возможных разветвлений цепей (второй прием), и как раз он не позволил поднять нижнюю оценку до $3n - O(1)$.

Было такое впечатление, что это удалось сделать К. Шнорру [86], но впоследствии в его доказательстве был обнаружен пробел. Он находится в самом начале доказательства леммы 9 и состоит в том, что там рассматривается вход i, обладающий рядом свойств, без предварительного решения вопроса о существовании входа с такими свойствами.

Дальше этой задачей занимался Н. Блюм. Сначала ему удалось «исправить» доказательство К. Шнорра, но лишь настолько, чтобы получить нижнюю опенку сложности вида $2.75n - o(n)$. Однако затем для другой, похожей, но более сложно определяемой булевой функции Н. Блюм получил [49] нижнюю оценку сложности в базисе $Б_2$ вида $3n - o(n)$.

Можно указать еще работу [50], в которой вместо получения нижней оценки сложности построенной схемы непосредственно доказывается ее минимальность. При этом для любых конкретных значений параметров соответствующей вектор-функции можно найти точное значение сложности минимальной схемы, но выразить его в аналитическом виде через параметры не удается.

Наконец, серия работ [56, 14, 15] была посвящена изучению *инверсионной сложности*, т. е. рассматривался базис, состоящий из операции отрицания и системы монотонных булевых функций, являющейся полной для класса монотонных булевых функций (например, базис $Б_0 \cup \{0, 1\} = \{\&, \vee, -, 0, 1\}$), а сложность схемы определялась как число инверторов в этой схеме. Для вектор-функции f определим сложность $\mathscr{L}(f)$ как минимум числа инверторов в схеме (над рассматриваемым базисом), реализующей f. Задача синтеза состоит в том, чтобы для заданной вектор-функции f построить схему, на которой достигается $\mathscr{L}(f)$.

В этом варианте задача синтеза оказалась значительно легче и была решена до конца. Существенный шаг на пути к ее ре-

шению сделал Э. Гилберт [56], затем А. А. Марков решил ее сначала для функции [14], а потом и для вектор-функции [15]. Сформулируем этот результат.

Пусть задана (n, m)-функция $f = (f_1, \ldots, f_m)$. Назовем *цепью* последовательность $\tilde{\alpha}_0, \tilde{\alpha}_1, \ldots, \tilde{\alpha}_n$ n-мерных наборов из нулей и единиц, в которой $\tilde{\alpha}_0 = (0, \ldots, 1)$, $\tilde{\alpha}_n = (1, \ldots, 1)$ и любой набор $\tilde{\alpha}_i$, $i = 1, \ldots, n$, получается из $\tilde{\alpha}_{i-1}$ заменой одного нуля на единицу. *Спуском* вектор-функции $f = (f_1, \ldots, f_m)$ на цепи $\tilde{\alpha}_0, \tilde{\alpha}_1, \ldots, \tilde{\alpha}_n$ назовем пару наборов $(\tilde{\alpha}_{i-1}, \tilde{\alpha}_i)$, $i = 1, \ldots, n$, на которой для какой-нибудь компоненты f_j $(1 \leqslant j \leqslant m)$ вектор-функции f выполняются равенства: $f_j(\tilde{\alpha}_{i-1}) = 1$, $f_j(\tilde{\alpha}_i) = 0$. Рассмотрим число спусков вектор-функции f на каждой цепи и максимум этих чисел, взятый по всем цепям, обозначим через $M^-(f)$. А. А. Марков показал [15], что

$$\mathscr{L}(f) = \lceil \log_2 (M^-(f) + 1) \rceil,$$

и указал алгоритм построения схемы с такой сложностью.

Высокие нижние оценки

Мы видели, что все полученные до сих пор нижние оценки сложности для схем произвольного вида, реализующих конкретные функции (вектор-функции), линейны относительно n. Такое положение продолжает сохраняться, несмотря на то что ведется активный поиск методов получения более высоких, нелинейных нижних оценок. Кое-что об этом уже говорилось. По-видимому, в этом направлении было довольно много неудачных попыток, и некоторые из них весьма поучительны. На одном таком примере мы остановимся.

В предыдущем разделе речь шла главным образом о реализации отдельных функций. Переходя к рассмотрению вектор-функций, следует отметить, что для (n, m)-функции f, у которой все компоненты существенно зависят от n переменных, первая тривиальная нижняя оценка сложности, например, в базисе $Б_2$, принимает вид

$$L_{Б_2}(f) \geqslant n + m - 2.$$

Поэтому для (n, m)-функции под «нелинейной» нижней оценкой следует понимать оценку, растущую быстрее, чем $n + m$ (сумма числа входов и выходов соответствующей схемы).

Предполагалось, что нелинейные нижние оценки сложности можно будет получить для коммутационных сетей [51, 2] или близких к ним объектов. Напомним, что коммутационная сеть — это ориентированный граф с n входными и n выходными полюсами, в котором для любой перестановки $i_1, \ldots, i_n$ чисел $1, 2, \ldots, n$ существует n таких непересекающихся ни по реб-

рам, ни в вершинах цепей, что j-я цепь соединяет вход с номером j и выход с номером i_j ($j = 1, \ldots, n$). Известно [2, 27], что коммутационная сеть содержит асимптотически не меньше $n \log_2 n$ ребер. Возникает вопрос, не приближает ли это нас к получению нелинейной нижней оценки сложности для какой-нибудь схемы?

Хотя коммутационная сеть еще не является схемой из функциональных элементов (не указаны операции, которые должны выполняться в вершинах, отличных от входов), ее в некотором смысле можно «вложить» в схему из функциональных элементов. Однако, для того чтобы эта схема действительно могла для любой перестановки «соединить» непересекающимися цепями входы с соответствующими выходами, схема должна иметь возможность «распознавать» перестановки. Это значит, что перестановки должны кодироваться и их коды в виде дополнительных переменных должны поступать в схему. Очевидно, что число новых переменных не меньше, чем $\log_2(n!) \gtrsim n \log_2 n$. Эта оценка совершенно аналогична приведенной ранее оценке для числа ребер и показывает, что простого пути получения нелинейной нижней оценки сложности здесь нет.

Оставалась все-таки надежда, что нелинейную нижнюю оценку сложности можно получить, рассматривая такую сеть с n входами и n выходами, для которой несущественна информация о том, какой именно вход с каким выходом должен соединяться, но зато имеется возможность любое множество входов соединить непересекающимися цепями с любым равномощным множеством выходов. Этим путем пытался идти Э. И. Нечипорук, пока не обнаружил, что число ребер такой сети может быть сделано порядка n, а значит, этот путь для получения нелинейных нижних оценок сложности тоже закрыт. (Такие сети теперь известны под названием суперконцентраторов [93].)

В сложившейся ситуации важно было бы понять, в какой мере трудности получения нелинейных нижних оценок сложности для конкретных функций (вектор-функций) являются принципиальными. При решении этого вопроса можно пойти даже на построение искусственного примера функции с высокой нижней оценкой сложности. Главное только, чтобы это способствовало разработке методов получения нижних оценок для тех функций (вектор-функций), которые действительно представляют интерес.

Попробуем обратиться к сложным функциям, о существовании которых уже говорилось. Сначала уточним, что о них известно. Доказательство существования булевых функций от n переменных со сложностью примерно $2^n/n$ [73] основано на том, что схем меньшей сложности просто не хватит для всех 2^{2^n} булевых функций от n переменных (метод Риордана — Шен-

нона [81]). Таким образом, это доказательство не несет никакой информации о сложных функциях, кроме того, что они *существуют*. Далее, О. Б. Лупанов доказал [9], что почти все булевы функции от n переменных имеют сложность, асимптотически равную $\rho 2^n/n$, где ρ — константа, зависящая от базиса и легко по нему вычисляемая. Здесь в основе нижней оценки лежит та же идея, а верхнюю оценку дает универсальный метод синтеза, применимый ко *всем* булевым функциям. В итоге оказывается, что многое известно о сложности сложных булевых функций, но сами они фактически неизвестны.

Поэтому если при каждом n выбрать некоторым способом по одной самой сложной функции от n переменных, то, конечно, для такой последовательности функций будет получена высокая нижняя оценка сложности, однако сами эти функции будут трудно доступны. Например, для вычисления значений такой функции не видно другого пути, кроме перебора всех более простых функций вместе со схемами, которые их реализуют. А это — возвращение к тривиальному алгоритму с его огромной трудоемкостью. Именно к последовательности самых сложных булевых функций относятся результаты работы [48], убеждающие в неизбежности такой трудоемкости. Однако как раз ее мы постоянно стремимся избежать. Но тогда нельзя определять функцию через величину ее сложности. Более того, желательно, чтобы функция была *конкретной*.

Следует, однако, признать, что все далеко не так просто, как мы об этом говорим. А. Эренфейхт [53], а затем А. Мейер [72] и Л. Стокмейер [89] указали путь построения сложных булевых функций на базе формальных логических теорий. При этом символы теории (их конечное число) кодируются наборами из нулей и единиц и в соответствии с этим кодируются предложения этой теории. Для каждого n рассматривается задача распознавания по двоичному набору длины n, является ли он кодом истинного предложения данной формальной теории (при ее естественной интерпретации). Оказывается, что булева функция f, решающая эту задачу (т. е. принимающая значение 1 на кодах истинных предложений и значение 0 в остальных случаях), может быть очень сложной.

Дело в том, что выразительные средства формальной теории обычно настолько велики, что в ней можно составить предложение (даже не очень большой длины) примерно такого содержания: «*Функция с наименьшим номером среди булевых функций от l переменных, имеющих в базисе* Бш *сложность не меньше* $2^{l-\varphi(l)}$, *принимает на наборе* $\tilde{\sigma}^l$ *значение* 1» (здесь l — фиксированное число ($l < n$), $\varphi(l)$ — вполне определенная функция от l, причем чаще всего $\varphi(l) = o(l)$, а набор $\tilde{\sigma}^l$ является переменным). Совершенно ясно, что схема для булевой функции f

от n переменных, распознающей истинность предложений формальной теории, при достаточно большом n будет распознавать и истинность предложения, приведенного выше, т. е. будет вычислять функцию, о которой в нем говорится. Поэтому такая схема имеет сложность не меньше $2^{l-\varphi(l)}$, и если разница между l и n не очень велика, то булева функция f оказывается сложной.

В данном случае величина сложности булевой функции f непосредственно в ее определение не входит. Более того, в предложении, которое еще надо обнаружить среди предложений, имеющих отношение к определению функции f, речь идет о сложности совсем другой функции. Однако оказывается, что эти функции (и их сложности) тесно связаны между собой. И снова приходится сомневаться в *конкретности* функции, для которой получена высокая нижняя оценка.

Тем не менее булева функция, распознающая истинность предложений формальной теории, производит впечатление интересной функции. Поэтому нельзя сказать, что здесь все ясно. Можно только предположить, что в любой формальной теории на самом деле представляют интерес лишь сравнительно простые фрагменты, в которых примеры, подобные приведенному выше, отсутствуют.

Другим способом сложные булевы функции построил Л. А. Шоломов [47]. В основе его способа лежит определение значений булевой функции по начальным значениям примитивно-рекурсивных функций, имеющих начальные номера в некоторой вычислимой нумерации (как отмечает автор, вместо класса примитивно-рекурсивных функций можно было бы взять и более просто вычисляемый класс функций). Л. А. Шоломов показал, что построенные им функции имеют сложность вплоть до $c2^n/n$, где c — некоторая константа. Таким образом, эти функции имеют по порядку максимальную сложность.

И здесь сложность булевой функции никак не фигурирует в ее определении. Зато для задания булевой функции используются такие сильные вычислительные средства, как примитивно-рекурсивные функции. В результате трудоемкость вычисления примитивно-рекурсивных функций частично переносится на построенные булевы функции и оказывается, что они могут вычисляться лишь с трудоемкостью перебора аналогичного тому, что в тривиальном алгоритме.

К настоящему времени построено уже немало примеров сложных булевых функций. В любом из них используется идея, близкая к тем, которые были рассмотрены выше. Отметим еще только работу С. С. Марченкова [16], в которой идет речь о функциях, имеющих определенное математическое содержание.

Рассмотренные примеры сложных булевых функций, по-видимому, не приближают нас к получению высоких нижних оценок для конкретных функций. Безусловно они интересны тем, что вскрывают связь между вещами, казавшимися далекими друг от друга. Однако для нас важнее другое. Эти примеры показывают, что те представления, которые сложились в области синтеза схем, не всегда уже помогают ориентироваться в сложных ситуациях, с которыми приходится сталкиваться. Далее, так и остался открытым вопрос о том, существуют ли принципиальные препятствия для получения высоких нижних оценок сложности, хотя бы нелинейных. Трудно сказать, каким путем будут решаться эти проблемы. Видимо, будет продолжаться поиск нижних оценок, но что-то, наверно, должно измениться и в общей системе взглядов и понятий.

Нижние оценки для формул

Каждой схеме из функциональных элементов с одним выходом соответствует формула в том же базисе, реализующая ту же функцию. Эта формула получается, если переменную, приписанную выходу схемы, выразить через входные переменные $x_1, \ldots, x_n$, исключая внутренние переменные $z_1, \ldots, z_{t-1}$ с помощью равенств, определяющих схему. При этом формула соответствует, вообще говоря, многим схемам. Однако если рассмотреть класс схем без ветвлений, в которых выход каждого элемента (кроме выходного) поступает ровно на один вход какого-нибудь одного элемента (на входах схемы ветвления допускаются), то между этими схемами и формулами можно будет установить такое взаимно однозначное соответствие, что соответствующие друг другу схема и формула будут изоморфны (т. е. они будут содержать одинаковое число операций и соответствующие друг другу операции будут просто совпадать).

Естественно, что задача получения нижних оценок сложности рассматривалась и в ослабленном варианте — для схем без ветвлений, т. е. для формул. Следует отметить, что это не только полезно для совершенствования техники получения нижних оценок, но имеет и самостоятельное значение, так как каждая нижняя оценка для сложности формул дает соответствующую нижнюю оценку для времени работы (глубины) схем, реализующих ту же функцию.

Под сложностью формулы F (в отличие от схемы без ветвлений) обычно понимают число символов переменных, входящих в формулу F, и обозначают ее через $L(F)$. Для булевой функции f сложность ее реализации формулами в базисе Б вводится как число $L^0_{\text{Б}}(f) = \min L(F)$, где минимум берется по всем формулам F в базисе Б, реализующим функцию f. Отме-

тим, что сложность функции f в классе схем без ветвлений мало отличается от $L^0_{\text{Б}}(f)$.

Среди методов получения нижних оценок для сложности $L^0_{\text{Б}}(f)$ мы рассмотрим только такие, которые позволяют получать нелинейные нижние оценки.

Рассмотрим сначала методы, применимые для любого базиса. Самый сильный метод принадлежит Э. И. Нечипоруку [19]. В своей статье [19] он не стал отделять метод от функции, получив прямо для предложенной им функции максимально возможную нижнюю оценку, равную по порядку $n^2/\log_2 n$. Сформулировать метод Нечипорука в самом общем виде несколько затруднительно (возможно, поэтому он и не рассматривал метод отдельно от функции). Однако для наиболее интересного случая (когда группы переменных не пересекаются — см. ниже) это, конечно, стоит сделать. Разные формулировки метода Нечипорука действительно через некоторое время были опубликованы [61, 77]. Здесь приводится формулировка, которая непосредственно извлекается из доказательства Э. И. Нечипорука [19].

Пусть $f(x_1, \ldots, x_n)$ — произвольная булева функция. Разобьем ее переменные на t непересекающихся групп. Пусть i-я группа содержит n_i переменных $\left(\sum_{i=1}^{t} n_i = n\right)$, и пусть при всевозможных подстановках констант в функцию $f(x_1, \ldots, x_n)$ вместо всех переменных, не входящих в i-ю группу, получается q_i различных функций, зависящих от переменных i-й группы $\left(q_i \leqslant 2^{n-n_i}\right.$ и $\left. q_i \leqslant 2^{2^{n_i}}\right)$. Тогда для любого (конечного) базиса Б

$$L^0_{\text{Б}}(f) \geqslant \frac{1}{r+2} \sum_{i=1}^{t} \log_2 q_i,$$

где r — максимум числа переменных у функций базиса Б.

Нетрудно заметить, что метод Нечипорука дает высокие нижние оценки для тех функций, у которых много подфункций определенного вида (эти функции немного напоминают функции из классов $P^n_{p_i, q_i}$, где $p_i = n - n_i$). Максимальная нижняя оценка, которую можно получить методом Нечипорука, как уже отмечалось, равна по порядку $n^2/\log_2 n$ (она получается, если, например, $t = n/\log_2 n$, $n_1 = \ldots = n_t = \log_2 n$ и $q_i = 2^{n-\log_2 n}$).

После Э. И. Нечипорука [19] его метод применялся многими. Пожалуй, наиболее интересное его применение было к определителю над полем из двух элементов 0 и 1. Для определителя была получена нижняя оценка сложности порядка $n^{3/2}$ [6], где $n = m^2$ — общее число элементов определителя. Дру-

гие применения метода Нечипорука можно найти в работах [61, 70, 76, 77, 1, 13].

Известен еще только один метод, применимый для любого базиса, это метод Шпекера — Ходеса [65, 88]. Однако этот метод позволяет получать лишь невысокие нижние оценки. В совершенствование этого метода и перенесение его на формулы k-значной логики внес вклад Б. Вильфан [94, 95]. А совсем недавно П. Пудлак [80] не только нашел более простое доказательство основной леммы, на которую опирается метод Шпекера — Ходеса, но и показал, что этим методом можно получить нижнюю оценку, по порядку равную $n \log_2 \log_2 n$ (до П. Пудлака множитель, вносящий нелинейность, имел гораздо более медленный рост — медленнее m-кратного логарифма при любом конечном m). Метод Шпекера — Ходеса мы сформулируем, почти не отступая от формулировки, данной П. Пудлаком.

Для любого (конечного) базиса Б существует такая положительная константа $c_{\text{Б}}$, что если булева функция $f(x_1, \ldots, x_n)$ при $r \geqslant 3$ удовлетворяет условию

$$L^0_{\text{Б}}(f) \leqslant c_{\text{Б}} n \left(\log_2 \log_2 n - \log_2 r\right),$$

то существуют такие r переменных $x_{i_1}, \ldots, x_{i_r}$, что при подстановке константы 0 вместо всех остальных переменных функции f из нее получается функция вида

$$\alpha_0 \oplus \alpha_1 \Lambda_r(x_{i_1}, \ldots, x_{i_r}) \oplus \alpha_2(\bar{x}_{i_1} \& \ldots \& \bar{x}_{i_r}).$$

Совершенно аналогично метод Шпекера — Ходеса (в редакции Пудлака) формулируется для случая подстановки константы 1 — только в последнем выражении конъюнкция $\bar{x}_{i_1} \& \ldots \& \bar{x}_{i_r}$ заменяется конъюнкцией $x_{i_1} \& \ldots \& x_{i_r}$.

При такой формулировке метод Шпекера — Ходеса применяется следующим образом. Для заданной функции $f(x_1, \ldots, x_n)$ подбирается такое r, чтобы из нее подстановками константы (0 или 1) нельзя было получить функцию того вида, о котором говорится в формулировке. Тогда функция f не будет удовлетворять условию формулировки, что и дает нижнюю оценку.

Свойства функции f, на которых основан метод Шпекера — Ходеса, довольно заметно отличаются от тех, которые мы до сих пор рассматривали. Они заслуживали бы серьезного изучения, если бы позволяли получать более высокие нижние оценки. Однако, как показал П. Пудлак [80], $n \log_2 \log_2 n$ — это предел для метода Шпекера — Ходеса.

Тем не менее метод Шпекера — Ходеса интересен благодаря тому, что он позволяет получать нелинейные нижние оценки для симметрических функций, дополняя тем самым метод Нечипорука, который здесь не работает. Первыми к симметрическим

функциям применили свой метод Э. Шпекер и Л. Ходес [65]. Затем метод Шпекера — Ходеса был применен в работах [75, 46], что позволило доказать примерно следующее: все симметрические функции от n переменных, кроме 16 функций вида $\beta_0 \oplus \beta_1 \Lambda_n(x_1, \ldots, x_n) \oplus \beta_2 \bar{x}_1 \ldots \bar{x}_n \oplus \beta_3 x_1 \ldots x_n$, имеют нелинейную сложность. Согласно П. Пудлаку [80], эта сложность имеет порядок не меньше $n \log_2 \log_2 n$. Еще ряд применений метода Шпекера — Ходеса имеется в работе [64].

Очень близок по духу к методу Шпекера — Ходеса метод Фишера — Мейера — Патерсона [54, 55]. Он позволяет получать нижние оценки, по порядку равные $n \log_2 n$, но только в базисе $Б_2$ (или в базисе, являющемся частью базиса $Б_2$). Этот метод в окончательном виде опубликован недавно, и его изложение было решено включить в обзор в качестве дополнения.

Все остальные методы относятся к базису $Б_0 = \{\&, \vee, \neg\}$. В этом базисе многие функции реализуются значительно сложнее, чем, например, в базисе $Б_2$ (результат Д. Мюллера [73] о том, что все базисы для схем примерно равноправны, на формулы не переносится из-за отсутствия ветвлений). Поэтому неудивительно, что в базисе $Б_0$ нижние оценки получаются несколько легче. Тем не менее можно назвать еще только четыре метода.

Первый из них восходит к 1961 г., когда Б. А. Субботовская получила первую нелинейную нижнюю оценку для формул [37]. Для линейной функции $\Lambda_n(x_1, \ldots, x_n)$ она доказала [37], что

$$L^0_{Б_0}(\Lambda_n) \geqslant cn^{3/2},$$

где c — некоторая константа. Здесь метод тоже не был отделен от функции. Сформулировать метод Субботовской можно, например, следующим образом.

Пусть $f(x_1, \ldots, x_n)$ — произвольная булева функция с числом переменных $n \geqslant 2$. Тогда существует такая переменная x_j, что

1) при любом $\alpha \in \{0, 1\}$

$$L^0_{Б_0}(f) \geqslant L^0_{Б_0}(f|_{x_j=\alpha}) \Big/ \left(1 - \frac{1}{n}\right),$$

2) существует такая константа $\beta \in \{0, 1\}$, что

$$L^0_{Б_0}(f) \geqslant L^0_{Б_0}(f|_{x_j=\beta}) \Big/ \left(1 - \frac{3}{2n}\right).$$

При применении метода Субботовской на каждом шаге следует действовать в соответствии с п. 2), если при этом не исчезает существенная зависимость функции от каких-нибудь из остающихся переменных, в противном случае приходится дей-

ствовать в соответствии с п. 1) (иначе получится слабая нижняя оценка).

Итак, в методе Субботовской используется существенная зависимость функции (а также ее подфункций) от своих переменных. Предельная нижняя оценка, которую дает метод Субботовской, равна по порядку $n^{3/2}$. Ее оценка для линейной функции впоследствии была улучшена другим методом, однако имеются примеры функций, такие, как характеристические функции (n, k)-кодов, для которых метод Субботовской дает почти такую же нижнюю оценку. Применение метода Субботовской к похожим базисам можно найти в работах [38, 39].

Метод, близкий по духу к методу Субботовской, предложил В. А. Малышев [12]. Однако его метод дает невысокую нижнюю оценку. В. А. Малышев показал [12], что если $f \in R_{1,2}^{n,k}$, то

$$L_{\text{Б}_0}^0(f) \geqslant c \frac{k \log_2 k}{\log_2 \log_2 k},$$

где c — некоторая константа. Таким образом, и здесь в основном используется существенная зависимость функции от своих переменных. Метод Малышева был применен затем в работе [29].

Один из методов был предложен [43, 44] автором этого обзора. Он формулируется следующим образом. Пусть $f(x_1, \ldots, x_n)$ — произвольная (быть может, не всюду определенная) булева функция, M_1 — некоторое подмножество вершин n-мерного единичного куба, на которых f равна 1, M_0 — некоторое подмножество вершин, на которых f равна 0, и R — множество всех ребер, соединяющих вершины из M_1 с вершинами из M_0. Тогда

$$L_{\text{Б}_0}^0(f) \geqslant \frac{|R|^2}{|M_1||M_0|},$$

где, как обычно, $|M|$ обозначает мощность множества M.

Здесь тоже главное в использовании существенной зависимости функции от своих переменных. Но при этом в подсчет вовлекаются все ребра, на которых эта существенная зависимость проявляется. Этим методом была получена оценка [43]

$$L_{\text{Б}_0}^0(\Lambda_n) \geqslant n^2,$$

и это предел для данного метода. Другие его применения имеются в работах [44, 45, 28, 30].

Несколько особое положение занимает метод Кричевского [7, 8]. Фактически это метод для одной функции $S_n^{2,3,\ldots,n}$. Однако стоит вспомнить, что и в этом случае метод применим к многим функциям.

Главное в методе Кричевского — это доказательство того, что у функции $S_n^{2,3,\ldots,n}$ существует минимальная формула без отрицаний (по-видимому, это как-раз нетипичная ситуация). При этом Р. Е. Кричевский доказал даже нечто большее, выяснив довольно точно строение такой минимальной схемы. В результате в работе [7] он получил для функции $S_n^{2,3,\ldots,n}$ нижнюю оценку, совпадающую с верхней оценкой по порядку. Далее, опираясь на основной результат своей работы [7] и используя результаты Ж. Анселя [57], а в конечном счете и результаты О. Б. Лупанова [10] для контактных схем из замыкающих контактов, в работе [8] Р. Е. Кричевский, усилив свою нижнюю оценку, нашел точное значение $L^0_{\text{Б}_0}(S_n^{2,3,\ldots,n}) = n\lfloor \log_2 n \rfloor + 2\left(n - 2^{\lfloor \log_2 n \rfloor}\right)$. Вскоре аналогичный результат для контактных схем из замыкающих контактов получил Ж. Ансель [58].

В заключение отметим, что вопрос об инверсионной сложности для схем без ветвлений полностью решил Э. И. Нечипорук [18]. Она оказалась равной $M^-(f)$.

Нижние оценки в случае неполных базисов

При реализации функций схемами в неполных базисах обычно возникают жесткие ограничения на структуру схемы. Почти все нижние оценки, которые были здесь получены, основаны на том, что удавалось достаточно точно выяснить строение минимальной схемы. Трудно рассчитывать на то, что эти методы могут быть перенесены на случай полного базиса. К тому же для большинства известных примеров высоких нижних оценок в неполных базисах соответствующие реализации в полных базисах оказывались проще.

Чаще всего рассматривается монотонный базис $\{\&, \vee\}$ и реализация в нем монотонных функций или вектор-функций. До сих пор в этом базисе не удалось получить нелинейную нижнюю оценку для отдельной функции. Первую нелинейную нижнюю оценку для вектор-функции получил Э. И. Нечипорук [20], рассматривая систему из n дизъюнкций от n переменных, каждая пара которых имеет не более чем по одной общей переменной. Для такой (n, n)-функции Э. И. Нечипорук получил нижнюю оценку $n^{3/2}$ [20]. Впоследствии для похожих вектор-функций были получены нижние оценки $n^{5/3}$ [69, 78]. Серия работ [79, 74, 71] была посвящена булеву произведению матриц. Здесь найдено точное значение сложности соответствующей $(2n, n)$-функции и доказана единственность (с точностью до простых преобразований) минимальной схемы [74]. Наиболее высокую нижнюю оценку для предложенной им (n, n)-функции получил

И. Вегенер [96, 97]. Его оценка имеет порядок $n^2/\log_2 n$ [97]. К этому направлению относятся также работы [4, 68, 67, 92].

Значительно более высокие нижние оценки удается получать для формул, правда, пока в других монотонных базисах [21, 34]. Эти нижние оценки имеют вид n^c, где константу c можно сделать любой. Пример нелинейной нижней оценки для формулы в базисе $\{\&, \vee\}$ для случая, когда в полном базисе $\{\&, \vee, {}^-\}$ функция реализуется с линейной сложностью, приведен в работе [26].

Еще больше возможностей появляется при увеличении числа значений, которые принимают переменные. Уже в трехзначной логике для схем в неполном базисе удается получить экспоненциальную нижнюю оценку сложности [40, 41]. В случае бесконечного числа значений такая нижняя оценка получается и в монотонном базисе [85].

В действительности случай, когда переменные принимают бесконечное число значений, имеет слишком много особенностей, чтобы его здесь рассматривать. На эту тему имеется уже большое число работ, но, к счастью, многие из них получили освещение в обзорной статье А. О. Слисенко [35].

Обратим внимание только на задачу вычисления всех элементарных симметрических многочленов от n переменных. В случае бесконечного поля для ее сложности (в естественном базисе) получена нижняя оценка, равная по порядку $n \log_2 n$ [91, 87], а в случае конечного поля ее сложность имеет порядок n [17].

Надо сказать хотя бы несколько слов о работах Л. Харпера и Дж. Сэвиджа [59, 63], относящихся к сложности синхронных схем, т. е. схем, в которых все цепи, идущие от входов к выходам, имеют одинаковую длину. Они предложили метод получения нижних оценок сложности для синхронных схем и применили его к ряду функций, наиболее интересная из которых — это определитель. Полученные нижние оценки по порядку равны $n \log_2 n$. Метод Харпера — Сэвиджа изложен в монографии [23].

Метод Фишера—Мейера—Патерсона [55]

Лемма 1. *Пусть* $F_1, \ldots, F_t$ *— конечные множества и* M_1 *— множество элементов, принадлежащих хотя бы двум из них, т. е.*

$$M_1 = \bigcup_{1 \leqslant i < j \leqslant t} (F_i \cap F_j).$$

Тогда существует такое разбиение совокупности $\{F_1, \ldots, F_t\}$

на две непересекающиеся части $\{F_{i_1}, \ldots, F_{i_l}\}$ и $\{F_{j_1}, \ldots, F_{j_m}\} \times$ $\times (i_n \neq j_k,\ l+m=t)$, *что*

$$\left|\left(\bigcup_{h=1}^{l} F_{i_h}\right) \cap \left(\bigcup_{k=1}^{m} F_{j_k}\right)\right| \geqslant \frac{1}{2}|M_1|.$$

Доказательство. Число всевозможных (упорядоченных) разбиений совокупности $\{F_1, \ldots, F_t\}$ на две части равно 2^t. Рассмотрим матрицу размера $|M_1| \times 2^t$, в которой каждому элементу множества M_1 соответствует (взаимно однозначно) строка, а каждому разбиению — столбец, причем на пересечении строки и столбца стоит 1, если соответствующий элемент принадлежит множествам из обеих частей $\{F_{i_1}, \ldots, F_{i_l}\}$ и $\{F_{j_1}, \ldots, \ldots, F_{j_m}\}$ соответствующего разбиения, т. е. принадлежит множеству

$$\left(\bigcup_{h=1}^{l} F_{i_h}\right) \cap \left(\bigcup_{k=1}^{m} F_{j_k}\right).$$

Пусть $x \in M_1$. Тогда $x \in F_p$ и $x \in F_q$ при некоторых p и q. Множества F_p и F_q ровно в половине разбиений попадают в разные части (разбиению, при котором F_p и F_q находятся в одной части, можно сопоставить разбиение, при котором F_q перенесено в другую часть). Поэтому каждая строка рассматриваемой матрицы содержит не менее $^1/_2 \cdot 2^t$ единиц, а вся матрица — не менее $^1/_2 \cdot 2^t |M_1|$ единиц. Но тогда хотя бы один из 2^t столбцов матрицы содержит не менее $^1/_2 \cdot |M_1|$ единиц. Этот столбец и соответствует разбиению, о котором говорится в лемме.

Будем рассматривать формулы в базисе $Б_2$ (состоящем из констант 0 и 1 и булевых функций одной и двух переменных). Обозначим число переменных, входящих в формулу F, через $n(F)$. Отметим, что число несущественных переменных среди них может быть любым от 0 до $n(F)$.

Формулу, получающуюся из формулы F при подстановке констант $A = \{x_{i_1} = \sigma_1, \ldots, x_{i_m} = \sigma_m\}$, где $\sigma_j \in \{0, 1\}$, $j = 1, \ldots$ $\ldots, m$, будем обозначать через $F|_A$ (никакие упрощения при этом не производятся). Очевидно, что переменными формулы $F|_A$ являются все переменные формулы F, кроме $x_{i_1}, \ldots, x_{i_m}$, и

$$n(F|_A) = n(F) - m.$$

Подстановку $A = \{x_{i_1} = \sigma_1, \ldots, x_{i_m} = \sigma_m\}$ $(\sigma_j \in \{0, 1\},\ j = = 1, \ldots, m)$ будем называть *центральной*, если среди констант $\sigma_1, \ldots, \sigma_m$

1) поровну нулей и единиц (m четное),
2) число нулей на 1 больше числа единиц (m нечетное),
3) число единиц на 1 больше числа нулей (m нечетное).

В случаях 1) и 2) подстановку будем называть также 0-*центральной*, а в случаях 1) и 3) — 1-*центральной*.

Заметим, что пустая подстановка является одновременно центральной, 0-центральной и 1-центральной.

В основе метода Фишера — Мейера — Патерсона лежит следующая

Лемма 2. *Пусть F — формула в базисе $Б_2$, причем каждая переменная входит в формулу F не более r раз ($r \geqslant 1$). Тогда существует такая центральная подстановка A, что формула $F|_A$ реализует линейную функцию и*

$$n(F|_A) \geqslant \frac{2}{K^r} n(F), \tag{1}$$

где K — положительная константа (например, можно взять $K = 8000$), причем в качестве A можно выбрать как 0-центральную, так и 1-центральную подстановку.

Замечание. Безусловно константа K весьма велика. Отчасти это связано с попыткой сделать более доступным изложение. Однако и у авторов метода константа K велика (они даже не вычисляют ее, но, по-видимому, у них $K = 1440$).

Доказательство будем вести индукцией по сложности $L(F)$ формулы F. При индуктивном переходе вместо неравенства (1) удобнее доказывать неравенство

$$n(F|_A) \geqslant b(r)\, n(F), \tag{2}$$

где

$$b(r) = \frac{cr^2}{K^r}, \tag{3}$$

а c — положительная константа. Если выполняется условие

1° $$c \geqslant 2,$$

то при $r \geqslant 1$ неравенство (2) влечет за собой неравенство (1).

Константа c (как, впрочем, и константа K) будет определена в конце доказательства из тех условий (подобных условию 1°), которые постепенно будут на нее накладываться.

Итак, приступим к доказательству леммы в слегка усиленной форме, отличающейся заменой неравенства (1) неравенством (2) (подразумевается, что условие 1° выполнено).

Начальный шаг: $L(F) = 0$. При этом $n(F) = 0$ и сама формула F реализует линейную функцию (константу). Взяв в качестве A единственно возможную — пустую подстановку, мы

тривиальным образом удовлетворим требованиям леммы, в том числе неравенству (2).

Предположим теперь, что $L(F) > 0$ и что для любой формулы, сложность которой меньше $L(F)$, лемма (с неравенством (2)) уже доказана. Будем доказывать ее для формулы F. Заметим, что теперь $n(F) \geqslant 1$.

Случай 1. Формула F содержит подформулу, реализующую константу и имеющую ненулевую сложность (этой подформулой может быть и сама формула F).

Заменив эту подформулу в формуле F константой, которую она реализует, мы получим формулу F', для которой справедливо тождество

$$F \equiv F'. \tag{4}$$

Далее, поскольку

$$L(F') < L(F),$$

по предположению индукции существует такая подстановка A (как 0-центральная, так и 1-центральная), что формула $F'|_A$ реализует линейную функцию и

$$n(F'|_A) \geqslant b(r)\, n(F'). \tag{2'}$$

В силу (4)

$$F|_A \equiv F'|_A,$$

а значит, формула $F|_A$ тоже реализует линейную функцию. Остается доказать неравенство (2).

Положим

$$\nu = n(F) - n(F'). \tag{5}$$

Заметим, что $\nu \geqslant 0$ и

$$\nu = n(F|_A) - n(F'|_A) \tag{6}$$

(подстановка A фиксирует часть переменных формулы F', а это в свою очередь — часть переменных формулы F). В силу (6), (2') и (5)

$$n(F|_A) = n(F'|_A) + \nu \geqslant b(r)\, n(F') + \nu = b(r)(n(F) - \nu) + \nu = \\ = b(r)\, n(F) + \nu(1 - b(r)).$$

Введем условие

2^0 $$b(r) \leqslant 1.$$

Тогда неравенство (2) будет выполнено.

Случай 2. Формула F имеет вид

$$F = F' \oplus x_i \tag{7}$$

(формула F' может содержать переменную x_i, а может и не содержать ее).

Поскольку

$$L(F') < L(F),$$

по предположению индукции существует такая подстановка A (как 0-центральная, так и 1-центральная), что формула $F'|_A$ реализует линейную функцию и (так же, как в случае 1) выполняется неравенство (2′). В силу (7)

$$F|_A = (F'|_A) \oplus (x_i|_A),$$

а значит, формула $F|_A$ тоже реализует линейную функцию. Неравенство (2) доказывается в точности так же, как в случае 1.

Случай 3. Число переменных формулы F удовлетворяет неравенству

$$b(r)\,n(F) \leqslant 1.$$

В качестве A можно взять любую центральную подстановку, фиксирующую все переменные, кроме одной. Тогда $F|_A$ реализует линейную функцию (ибо все функции одной переменной являются линейными) и

$$n(F|_A) = 1 \geqslant b(r)\,n(F).$$

Случай 4 включает в себя все, что не вошло в предыдущие случаи и является основным.

Начнем с простого соображения. Если лемму доказать не для формулы F, а для эквивалентной ей формулы с тем же числом вхождений каждой переменной, то тем самым лемма будет доказана и для формулы F. Это соображение позволит нам делать некоторые простые преобразования формулы F.

Можно считать, что формула F не содержит подформул, тождественно равных константе. Действительно, если бы в формуле F была такая подформула ненулевой сложности, то мы имели бы случай 1, а если бы все такие подформулы имели нулевую сложность, то мы удалили бы их из формулы F, заменив одновременно те двуместные операции, у которых ровно на одно место поступает константа, соответствующими одноместными операциями. При этом число вхождений каждой переменной в формулу F осталось бы прежним. Итак, считаем, что в формуле F нет подформул, тождественно равных константе.

Пользуясь коммутативностью и ассоциативностью сложения по модулю 2, представим формулу F в виде

$$F = F_1 \oplus \ldots \oplus F_i \oplus \ldots \oplus F_t \oplus \sigma, \qquad (8)$$

где ни в одной подформуле F_i, $i = 1, \ldots, t$, внешняя операция (если она есть) уже не является линейной, причем $t \geqslant 1$, а $\sigma \in \{0, 1\}$. Заметим, что такое преобразование сохраняет неизменным число вхождений каждой переменной в формулу F.

Далее, каждая подформула F_i, $i = 1, \ldots, t$, отлична от переменной, так как иначе (с точностью до коммутативности и ассоциативности сложения по модулю 2) мы имели бы случай 2. Поэтому каждую подформулу F_i можно представить в следующем виде (сохранив число вхождений каждой переменной):

$$F_i = \left((G_i)^{\alpha_i} \& (H_i)^{\beta_i}\right) \oplus \gamma_i, \tag{9}$$

где G_i и H_i — подформулы формулы F_i, причем без ограничения общности можно считать, что $\alpha_i = \beta_i = 1$, а $\gamma_i = 0$ (константу γ_i можно включить в константу σ из (8), а изменив обозначения, можно взять в качестве G_i подформулу $(G_i)^{\alpha_i}$ из (9) и в качестве H_i подформулу $(H_i)^{\beta_i}$ из (9)). Итак, каждая подформула F_i, $i = 1, \ldots, t$, представима в виде

$$F_i = G_i \& H_i, \tag{9'}$$

где обе подформулы G_i и H_i отличны от констант.

Множество переменных формулы F разобьем на три подмножества:

M_1 — множество переменных, входящих хотя бы в две подформулы из $F_1, \ldots, F_t$;

M_2 — множество переменных, входящих лишь в одну подформулу F_i $(i = 1, \ldots, t)$ и входящих одновременно в подформулы G_i и H_i;

M_3 — множество всех остальных переменных формулы F. Очевидно, что множество M_3 состоит из переменных, каждая из которых входит либо лишь в подформулу G_i (при одном каком-то i), либо лишь в подформулу H_j (при одном каком-то j).

Для краткости положим $n = n(F)$ и введем новую константу $\alpha > 0$, которая также будет определена в конце доказательства. В случае 4 рассмотрим три подслучая, соответствующие следующим неравенствам:

$$|M_1| \geqslant 2\alpha n,$$
$$|M_2| \geqslant 3\alpha n,$$
$$|M_3| \geqslant (1 - 5\alpha) n.$$

Очевидно, что этими подслучаями случай 4 исчерпывается полностью.

Случай 4.1. Выполняется неравенство

$$|M_1| \geqslant 2\alpha n. \tag{10}$$

Применим лемму 1, понимая под $F_1, \ldots, F_t$ множества переменных, входящих соответственно в подформулы $F_1, \ldots, F_t$

из (8). На основе полученного разбиения преобразуем (8) к следующему виду:

$$F = F' \oplus F'' \oplus \sigma,$$

где

$$F' = F_{i_1} \oplus \ldots \oplus F_{i_l},$$
$$F'' = F_{j_1} \oplus \ldots \oplus F_{j_m},$$

причем в силу леммы 1 и неравенства (10) подформулы F' и F'' имеют не меньше αn общих переменных.

Пусть A_1 — центральная подстановка (ее можно взять как 0-центральной, так и 1-центральной), фиксирующая все переменные формулы F, не являющиеся общими для F' и F''. Тогда

$$F|_{A_1} = (F'|_{A_1}) \oplus (F''|_{A_1}) \oplus \sigma$$

и

$$n(F'|_{A_1}) = n(F''|_{A_1}) = n(F|_{A_1}) \geqslant \alpha n. \tag{11}$$

Очевидно, что каждая переменная входит в каждую из подформул $F'|_{A_1}$ и $F''|_{A_1}$ не более $r-1$ раз. Обозначим через V_s множество переменных, входящих ровно s раз в $F'|_{A_1}$ (и, следовательно, не более $r-s$ раз в $F''|_{A_1}$), $s = 1, \ldots, r-1$. В силу (11)

$$\sum_{s=1}^{r-1} |V_s| \geqslant \alpha n. \tag{12}$$

Пусть теперь s — произвольное фиксированное число из множества $\{1, \ldots, r-1\}$, и пусть A_2 — центральная подстановка, фиксирующая все переменные формулы $F|_{A_1}$, не входящие в V_s. Далее, пусть $B = A_1A_2$ (т. е. B — подстановка, сформированная из подстановок A_1 и A_2). Ясно, что при соответствующем выборе A_2 подстановку B можно сделать как 0-центральной, так и 1-центральной. Очевидно, что

$$F|_B = (F'|_B) \oplus (F''|_B) \oplus \sigma,$$

и

$$n(F'|_B) = n(F''|_B) = n(F|_B) = |V_s|, \tag{13}$$

причем каждая переменная входит ровно s раз в $F'|_B$ и не более $r-s$ раз в $F''|_B$.

Пусть A_3 — центральная подстановка, которая по предположению индукции существует для формулы $F'|_B$ (проверка неравенства $L(F'|_B) < L(F)$ не представляет труда), и пусть $C = BA_3$. Тогда

$$n(F'|_C) \geqslant b(s)\, n(F'|_B),$$

или (что то же самое)

$$n(F|_C) \geqslant b(s)\, n(F|_B), \tag{14}$$

причем в формуле

$$F|_C = (F'|_C) \oplus (F''|_C) \oplus \sigma$$

подформула $F'|_C$ уже реализует линейную функцию.

Пусть A_4 — центральная подстановка, которая по предположению индукции существует для формулы $F''|_C$, и пусть $A = CA_4$. Тогда

$$n(F''|_A) \geqslant b(r-s)\, n(F''|_C)$$

или (что то же самое)

$$n(F|_A) \geqslant b(r-s)\, n(F|_C), \tag{15}$$

причем в формуле

$$F|_A = (F'|_A) \oplus (F''|_A) \oplus \sigma$$

подформула $F''|_A$ уже реализует линейную функцию. Но и подформула $F'|_A$ реализует линейную функцию, так как она получена подстановкой из подформулы $F'|_C$, уже обладающей этим свойством. Значит, и формула $F|_A$ реализует линейную функцию, причем нетрудно проверить, что подстановку A можно взять как 0-центральной, так и 1-центральной. Осталось доказать неравенство (2).

Вспоминая, что $b(r) > 0$ при всех r (см. (3)), из (15), (14) и (13) получаем

$$n(F|_A) \geqslant b(r-s)\, b(s)\, |V_s|.$$

Отсюда видно, что неравенство (2) будет доказано, если для какого-нибудь s из множества $\{1, \ldots, r-1\}$ мы установим неравенство

$$|V_s| \geqslant \frac{b(r)\, n}{b(s)\, b(r-s)}.$$

Последнее неравенство в свою очередь будет выполняться хотя бы при одном s из множества $\{1, \ldots, r-1\}$, если будет выполняться неравенство

$$\sum_{s=1}^{r-1} |V_s| \geqslant \sum_{s=1}^{r-1} \frac{b(r)\, n}{b(s)\, b(r-s)}.$$

Наконец, это неравенство будет выполняться, если будет выполняться следующее условие (чтобы его получить, надо воспользоваться (12)):

3°
$$\alpha \geqslant b(r) \sum_{s=1}^{r-1} \frac{1}{b(s)\, b(r-s)},$$

которое мы и налагаем на константу α и на функцию $b(r)$ (т. е. фактически на константы K и c).

Подводя итог, мы можем сделать вывод, что при условии 3° неравенство (2) выполняется, а значит, подстановка A удовлетворяет лемме.

Случай 4.2. Выполняется неравенство

$$|M_2| \geqslant 3\alpha n. \tag{16}$$

Пусть A' — центральная подстановка, фиксирующая все переменные формулы F, не входящие в множество M_2. Тогда (см. (8))

$$F|_{A'} = (F_1|_{A'}) \oplus \ldots \oplus (F_i|_{A'}) \oplus \ldots \oplus (F_t|_{A'}) \oplus \sigma,$$

где (см. (9′))

$$F_i|_{A'} = (G_i|_{A'}) \,\&\, (H_i|_{A'}), \qquad i = 1, \ldots, t,$$
$$n(G_i|_{A'}) = n(H_i|_{A'}) = n(F_i|_{A'}),$$

причем множества переменных, входящих в формулы $F_1|_{A'}, \ldots, F_i|_{A'}, \ldots, F_t|_{A'}$, не пересекаются.

Очевидно, что каждая переменная входит в каждую из подформул $G_i|_{A'}$ и $H_i|_{A'}$, $i = 1, \ldots, t$, не более $r-1$ раз. Обозначим через V_{si} множество переменных, входящих ровно s раз в $G_i|_{A'}$ (и, следовательно, не более $r-s$ раз в $H_i|_{A'}$), $s = 1, \ldots, r-1$, $i = 1, \ldots, t$. Положим

$$V_s = \bigcup_{i=1}^{t} V_{si}.$$

В силу (16)

$$\sum_{s=1}^{r-1} |V_s| \geqslant 3\alpha n. \tag{17}$$

Дальше действуем подобно тому, как в случае 4.1. Однако, поскольку условие 3° уже введено, рассуждения теперь можно вести в другом, более удобном порядке.

Из (17) и условия 3° следует, что

$$\sum_{s=1}^{r-1} |V_s| \geqslant \sum_{s=1}^{r-1} \frac{3b(r)\,n}{b(s)\,b(r-s)},$$

а значит, для какого-нибудь s из множества $\{1, \ldots, r-1\}$ выполняется неравенство

$$|V_s| \geqslant \frac{3b(r)\,n}{b(s)\,b(r-s)}. \tag{18}$$

Рассмотрим это число s. Пусть A'' — центральная подстановка, фиксирующая все переменные формулы $F|_{A'}$, не входящие в V_s, и пусть $B = A'A''$. Тогда

$$F|_B = (F_1|_B) \oplus \ldots \oplus (F_i|_B) \oplus \ldots \oplus (F_t|_B) \oplus \sigma,$$

где

$$F_i|_B = (G_i|_B) \,\&\, (H_i|_B), \qquad i = 1, \ldots, t,$$

$$n(G_i|_B) = n(H_i|_B) = n(F_i|_B)$$

и

$$\sum_{i=1}^{t} n(F_i|_B) = |V_s|, \tag{19}$$

причем каждая переменная входит при некотором i ровно s раз в $G_i|_B$ и не более $r-s$ раз в $H_i|_B$ и не входит ни в $G_j|_B$, ни в $H_j|_B$ при $j \neq i$.

Рассмотрим теперь произвольную формулу $F_i|_B$. Пусть A_i''' — центральная подстановка, которая по предположению индукции существует для формулы $G_i|_B$, и пусть $C_i = BA_i'''$. Тогда

$$n(G_i|_{C_i}) \geqslant b(s)\, n(G_i|_B),$$

или (что то же самое)

$$n(F_i|_{C_i}) \geqslant b(s)\, n(F_i|_B), \tag{20}$$

причем в формуле

$$F_i|_{C_i} = (G_i|_{C_i}) \,\&\, (H_i|_{C_i})$$

подформула $G_i|_{C_i}$ реализует линейную функцию.

Пусть A_i^{IV} — центральная подстановка, которая по предположению индукции существует для формулы $H_i|_{C_i}$, и пусть $D_i = C_iA_i^{\mathrm{IV}}$. Тогда

$$n(H_i|_{D_i}) \geqslant b(r-s)\, n(H_i|_{C_i}),$$

или (что то же самое)

$$n(F_i|_{D_i}) \geqslant b(r-s)\, n(F_i|_{C_i}), \tag{21}$$

причем в формуле

$$F_i|_{D_i} = (G_i|_{D_i}) \,\&\, (H_i|_{D_i})$$

подформула $H_i|_{D_i}$ реализует линейную функцию, как, впрочем, и подформула $G_i|_{D_i}$.

Множество переменных формулы $F_i|_{D_i}$ разобьем на четыре подмножества: N_{1i} — множество переменных, несущественных для $G_i|_{D_i}$ и $H_i|_{D_i}$, N_{2i} — множество переменных, несущественн-

ных для $G_i|_{D_i}$ и существенных для $H_i|_{D_i}$, N_{3i} — множество переменных, существенных для $G_i|_{D_i}$ и несущественных для $H_i|_{D_i}$, и N_{4i} — множество переменных, существенных для $G_i|_{D_i}$ и $H_i|_{D_i}$. Тогда

$$|N_{1i}|+|N_{2i}|+|N_{3i}|+|N_{4i}|=n(F_i|_{D_i}). \tag{22}$$

Обозначим через N_i множество, на котором достигается

$$\max(|N_{2i}|,\ |N_{3i}|,\ |N_{4i}|).$$

Тогда из (22) получим

$$n(F_i|_{D_i})\leqslant|N_{1i}|+3|N_i|\leqslant 3(|N_{1i}|+|N_i|)$$

и, следовательно,

$$|N_{1i}|+|N_i|\geqslant\frac{1}{3}n(F_i|_{D_i}). \tag{23}$$

Пусть теперь $A_i^{\vee}$ — центральная подстановка, фиксирующая все переменные формулы $F_i|_{D_i}$, не входящие в множество $N_{1i}\cup N_i$, и пусть $E_i=D_iA_i^{\vee}$. Тогда независимо от того, совпадает N_i с N_{2i} или с N_{3i}, или с N_{4i}, формула

$$F_i|_{E_i}=(G_i|_{E_i})\,\&\,(H_i|_{E_i})$$

реализует линейную функцию, причем

$$n(F_i|_{E_i})=|N_{1i}|+|N_i|. \tag{24}$$

При $i\neq j$ подстановки E_i и E_j имеют общую часть — подстановку B, а в остальном они действуют на множествах переменных, входящих соответственно в $F_i|_B$ и $F_j|_B$, которые, согласно сказанному ранее, при $i\neq j$ не пересекаются. Поэтому подстановки E_i, $i=1,\ldots,t$, не противоречат друг другу, и из них можно сформировать новую подстановку, которую мы обозначим через A. Очевидно, что при любом $i=1,\ldots,t$

$$\begin{aligned} F_i|_A&=F_i|_{E_i},\\ n(F_i|_A)&=n(F_i|_{E_i}),\end{aligned} \tag{25}$$

и формула

$$F|_A=(F_1|_A)\oplus\ldots\oplus(F_i|_A)\oplus\ldots\oplus(F_t|_A)\oplus\sigma$$

реализует линейную функцию.

Из (25), (24), (23), (21) и (20) следует, что

$$n(F_i|_A)\geqslant\frac{1}{3}b(r-s)\,b(s)\,n(F_i|_B).$$

Суммируя эти неравенства и учитывая (19) и (18), получим

$$\sum_{i=1}^{t} n(F_i|_A) \geqslant b(r)\, n.$$

Поскольку множества переменных, входящих в разные подформулы $F_i|_A$, $i = 1, \ldots, t$, не пересекаются, сумма в левой части последнего неравенства совпадает с $n(F|_A)$, т. е. мы получили неравенство (2).

Наконец, ввиду того что каждую из подстановок A', A'', A_i''', A_i^{IV} и A_i^{V}, $i = 1, \ldots, t$, можно было выбрать как 0-центральной, так и 1-центральной, это же справедливо для подстановки A.

Случай 4.3. Выполняется неравенство

$$|M_3| \geqslant (1 - 5\alpha)\, n. \tag{26}$$

Кроме того, выполняется неравенство

$$b(r)\, n \gtrsim 1, \tag{27}$$

так как иначе мы имели бы случай 3.

Обозначим через g_i число переменных, входящих лишь в подформулу G_i, а через h_i число переменных, входящих лишь в подформулу H_i. Заметим, что

$$\sum_{i=1}^{t} g_i + \sum_{i=1}^{t} h_i = |M_3|. \tag{28}$$

Без ограничения общности можно считать, что

$$g_i \geqslant h_i.$$

При этом из (28) и (26) получаем

$$\sum_{i=1}^{t} g_i \geqslant \tfrac{1}{2}(1 - 5\alpha)\, n. \tag{29}$$

Из (9′) следует, что при любой подстановке, фиксирующей переменные подформулы H_i и обращающей ее в 0, становятся несущественными те переменные формулы F (см. (8)), которые входят лишь в подформулу G_i. Это подсказывает дальнейший план.

Однако сначала полезно выяснить, при каких условиях непустую центральную подстановку B (в формулу F), которая фиксирует переменные одного множества U и делает несущественными переменные другого множества V, можно продолжить до центральной подстановки A, удовлетворяющей лемме (с неравенством (2)).

Связь между подстановкой B и множествами U и V описывается следующими тремя соотношениями:

$$n = |U| + n(F|_B), \tag{30}$$

при любой подстановке C, фиксирующей переменные множества V,

$$F|_B \equiv F|_{BC} \tag{31}$$

(т. е. $F|_B$ и $F|_{BC}$ реализуют одну и ту же функцию) и

$$n(F|_B) = |V| + n(F|_{BC}). \tag{32}$$

Пусть D — центральная подстановка, которая по предположению индукции существует для формулы $F|_{BC}$. Тогда формула $F|_{BCD}$ реализует линейную функцию и

$$n(F|_{BCD}) \geqslant b(r)\, n(F|_{BC}). \tag{33}$$

В силу (31)

$$F|_{BD} \equiv F|_{BCD},$$

а значит, формула $F|_{BD}$ тоже реализует линейную функцию, причем

$$n(F|_{BD}) = |V| + n(F|_{BCD}),$$

откуда в силу (33), (32) и (30) следует, что

$$n(F|_{BD}) \geqslant |V| + b(r)\, n(F|_{BC}) = |V| + b(r)\,(n(F|_B) - |V|) =$$
$$= |V| + b(r)\,(n - |U| - |V|) = b(r)\, n + |V| - b(r)\,(|U| + |V|).$$

Отсюда видно, что подстановка $A = BD$ будет удовлетворять лемме (с неравенством (2)), если будет выполняться неравенство

$$|V| \geqslant b(r)\,(|U| + |V|), \tag{34}$$

а подстановку B можно будет выбрать как 0-центральной, так и 1-центральной.

Теперь при каждом $i = 1, \ldots, t$ среди подстановок, фиксирующих все переменные подформулы H_i и обращающих H_i в 0 (подформула H_i отлична от константы, так что такие подстановки существуют), выберем ту (одну из тех), для которой абсолютная величина разности между числом нулей и числом единиц в подстановке минимальна. Обозначим этот минимум через $d(H_i)$. Ясно, что

$$d(H_i) \leqslant n(H_i). \tag{35}$$

Далее, если

$$n(H_i) + d(H_i) \leqslant n, \tag{36}$$

то выбранную подстановку можно продолжить до центральной подстановки B_i, подставляя константу (одну и ту же) вместо $d(H_i)$ переменных формулы F, не входящих в подформулу H_i.

При каждом $i = 1, \ldots, t$ эти $d(H_i)$ переменных выберем так, чтобы среди них оказалось как можно меньше тех переменных, которые входят лишь в подформулу G_i. Обозначим через U_i множество переменных, фиксируемых подстановкой B_i, а через V_i — множество тех переменных, которые входят лишь в подформулу G_i и не принадлежат множеству U_i. Очевидно, что

$$|U_i| = n(H_i) + d(H_i), \tag{37}$$

$$|V_i| = \min(g_i,\ n - (n(H_i) + d(H_i))), \tag{38}$$

причем подстановка B_i делает несущественными переменные множества V_i.

Случай 4.3 разобьем еще на два подслучая.

Случай 4.3.1. При любом $i = 1, \ldots, t$ выполняется неравенство

$$n - (n(H_i) + d(H_i)) \geqslant g_i. \tag{39}$$

Тогда выполняется неравенство (36), а значит, подстановка B_i существует. При этом для U_i выполняется соотношение (37), а соотношение (38) для V_i в силу (39) можно переписать в виде

$$|V_i| = g_i. \tag{38'}$$

Согласно сказанному выше, центральную подстановку B_i можно продолжить до центральной подстановки A, удовлетворяющей лемме (с неравенством (2)), если выполняется неравенство (34), которое в данном случае с учетом (37) и (38′) можно записать так:

$$g_i \geqslant b(r)(n(H_i) + d(H_i) + g_i),$$

или (что то же самое)

$$(1 - b(r))\, g_i \geqslant b(r)(n(H_i) + d(H_i)).$$

Последнее неравенство будет выполняться хотя бы при одном i из множества $\{1, \ldots, t\}$, если будет выполняться неравенство

$$\sum_{i=1}^{t} (1 - b(r))\, g_i \geqslant \sum_{i=1}^{t} b(r)(n(H_i) + d(H_i)).$$

Это неравенство в свою очередь будет выполняться, если будет выполняться следующее неравенство (чтобы его получить, надо воспользоваться (29) и (35)):

$$\frac{1}{2}(1 - b(r))(1 - 5\alpha)\, n \geqslant 2b(r) \sum_{i=1}^{t} n(H_i).$$

Наконец, из цепочки неравенств

$$\sum_{i=1}^{t} n(H_i) \leqslant \sum_{i=1}^{t} L(H_i) \leqslant L(F) \leqslant nr$$

вытекает, что предыдущее неравенство будет выполняться, если будет выполняться условие

$$(1 - b(r))(1 - 5\alpha) \geqslant 4b(r)r$$

или несколько более сильное условие ($\alpha > 0$ и $b(r) > 0$ — см. (3))

4° $$1 - 5\alpha \geqslant b(r)(4r + 1),$$

которое мы и налагаем на константу α и функцию $b(r)$.

Подводя итог, мы можем сделать вывод, что при условии 4° для некоторого i из множества $\{1, \ldots, t\}$ подстановку B_i можно продолжить до подстановки A, удовлетворяющей лемме (с неравенством (2)).

Случай 4.3.2. При некотором i из множества $\{1, \ldots, t\}$ выполняется неравенство

$$n - (n(H_i) + d(H_i)) < g_i. \tag{40}$$

Если наряду с этим (при том же i) выполняется неравенство

$$n - (n(H_i) + d(H_i)) \geqslant b(r)n, \tag{41}$$

то выполняется неравенство (36), а значит, подстановка B_i существует. Но тогда в силу (37), (38) и (40)

$$n(F|_{B_i}) = n - (n(H_i) + d(H_i)) = |V_i|,$$

т. е., во-первых, все переменные формулы $F|_{B_i}$ являются несущественными и, таким образом, $F|_{B_i}$ реализует константу, а, во-вторых, неравенство (41) принимает вид

$$n(F|_{B_i}) \geqslant b(r)n.$$

Отсюда видно, что в качестве подстановки A, удовлетворяющей лемме (с неравенством (2)), можно взять подстановку B_i.

Поэтому остается рассмотреть ситуацию, когда неравенство (41) не выполняется, а значит, выполняется неравенство

$$(1 - b(r))n < n(H_i) + d(H_i). \tag{42}$$

Каждому набору $\tilde{\sigma}$ значений переменных формулы H_i сопоставим число $d(\tilde{\sigma})$, равное абсолютной величине разности между числом нулей и числом единиц в наборе $\tilde{\sigma}$. Согласно определению величины $d(H_i)$, существует такой набор $\tilde{\sigma}_0$, что $d(\tilde{\sigma}_0) =$

$= d(H_i)$ и $H_i(\tilde{\sigma}_0) = 0$, а для всех наборов $\tilde{\sigma}$, таких, что $d(\tilde{\sigma}) < d(H_i)$, непременно $H_i(\tilde{\sigma}) = 1$.

Поэтому если в формуле H_i выбрать любым способом менее $d(H_i)$ переменных, а вместо всех остальных ее переменных подставить поровну нулей и единиц, то H_i превратится в формулу, тождественно равную 1.

Наибольшее число $d(H_i) - 1$ переменных можно было бы выбрать, если бы было четным число $n(H_i) - d(H_i) + 1$ остальных переменных формулы H_i. Тем не менее либо число $n(H_i) - d(H_i) + 1$, либо число $n(H_i) - d(H_i) + 2$ четное (на самом деле четным является число $n(H_i) - d(H_i) + 2$, в чем можно убедиться, воспользовавшись существованием набора $\tilde{\sigma}_0$, для которого $d(\tilde{\sigma}_0) = d(H_i)$). Поэтому заведомо можно выбрать не меньше чем $d(H_i) - 2$ переменных.

Эти $d(H_i) - 2$ переменных выберем так, чтобы среди них оказалось как можно больше тех переменных, которые входят лишь в подформулу H_i. Затем сделаем центральную подстановку, фиксирующую все остальные переменные подформулы H_i (при этом, как уже отмечалось, H_i обратится в 1), и продолжим ее до центральной подстановки A, фиксирующей помимо этого все переменные подформулы G_i и подформул $F_1, \ldots, F_{i-1}, F_{i+1}, \ldots, F_t$ (см. (9′) и (8)). В результате получится формула $F|_A$, реализующая константу, причем

$$n(F|_A) = \min(d(H_i) - 2, h_i).$$

Осталось доказать неравенство (2), которое теперь запишется в виде

$$\min(d(H_i) - 2, h_i) \geqslant b(r)n. \qquad (2'')$$

Замечая, что h_i — это число переменных формулы H_i, принадлежащих множеству M_3, и учитывая, что число переменных формулы H_i, не принадлежащих множеству M_3, не может быть больше $n - |M_3|$, получим

$$h_i \geqslant n(H_i) - (n - |M_3|).$$

В силу (35) и (26) отсюда следует, что

$$h_i \geqslant d(H_i) - 5\alpha n$$

и, таким образом,

$$\min(d(H_i) - 2, h_i) \geqslant d(H_i) - 5\alpha n - 2.$$

Поэтому неравенство (2″) будет доказано, если мы установим неравенство

$$d(H_i) - 5\alpha n - 2 \geqslant b(r)n.$$

Из цепочки неравенств (полученной с помощью (42), определения чисел g_i и (29))

$$d(H_l) > (1-b(r))n - n(H_l) \geqslant$$

$$\geqslant (1-b(r))n - \left(n - \sum_{i=1}^{t} g_i\right) \geqslant \frac{1}{2}(1-5\alpha)n - b(r)n$$

вытекает, что предыдущее неравенство будет выполняться, если будет выполняться неравенство

$$\frac{1}{2}(1-5\alpha)n - b(r)n \geqslant b(r)n + 5\alpha n + 2.$$

Наконец, это неравенство будет выполняться, если будет выполняться следующее условие (чтобы его получить, надо воспользоваться (27)):

5° $$1 - 15\alpha \geqslant 8b(r),$$

которое мы и налагаем на константу α и функцию $b(r)$.

Подводя итог, мы можем сделать вывод, что при условии 5° неравенство (2″) выполняется, а значит, подстановка A удовлетворяет лемме.

Теперь остается подобрать положительные константы K, c и α так, чтобы выполнялись условия 1°—5°. С учетом (3) эти условия перепишутся в следующем виде:

1° $$c \geqslant 2,$$

2° $$1 \geqslant \frac{cr^2}{K^r},$$

3° $$\frac{\alpha c}{r^2} \geqslant \sum_{s=1}^{r-1} \frac{1}{s^2(r-s)^2},$$

4° $$1 \geqslant 5\alpha + \frac{cr^2(4r+1)}{K^r},$$

5° $$1 \geqslant 15\alpha + \frac{8cr^2}{K^r}.$$

Заметим, что условие 2° следует, например, из условия 5°, а условие 1° следует, например, из условий 3° и 5°: в силу 3° (которое должно выполняться при любом r, скажем при $r = 2$) $\alpha c \geqslant 4$, а в силу 5° $1 \geqslant 15\alpha$, т. е. $c \geqslant 60$. Поэтому достаточно, чтобы выполнялись условия 3°—5°.

Начнем с того, что заменим условие 3° более сильным условием, которое зато легче проверяется. С этой целью оценим сверху правую часть условия 3°:

$$\sum_{s=1}^{r-1} \frac{1}{s^2(r-s)^2} = \sum_{s=1}^{[r/2]} \frac{1}{s^2(r-s)^2} + \sum_{s=[r/2]+1}^{r-1} \frac{1}{s^2(r-s)^2} =$$

$$= \sum_{s=1}^{[r/2]} \frac{1}{s^2 (r-s)^2} + \sum_{k=1}^{r-[r/2]-1} \frac{1}{(r-k)^2 k^2} \leqslant \sum_{s=1}^{[r/2]} \frac{1}{s^2 (r/2)^2} +$$

$$+ \sum_{k=1}^{r-[r/2]-1} \frac{1}{k^2 (r/2)^2} < 2\left(\frac{2}{r}\right)^2 \sum_{k=1}^{\infty} \frac{1}{k^2} < \frac{16}{r^2}\ ^{1}).$$

Отсюда следует, что условие 3° будет выполнено, если будет выполняться условие

3°°
$$\alpha c \geqslant 16.$$

Итак, константы K, c и α будем выбирать, исходя из условий 3°°, 4° и 5°.

Возьмем (так же, как и авторы метода)

$$\alpha = 1/30.$$

Тогда в силу 3°° $c \geqslant 480$. Возьмем

$$c = 500.$$

Тогда из условия 5° при $r = 1$ получим: $K \geqslant 8000$. Возьмем

$$K = 8000.$$

Нетрудно проверить, что при этих значениях K, c и α условия 4° и 5° выполняются при любом $r \geqslant 1$. Как мы уже видели, при этом выполняются и условия 1°, 2° и 3°. Лемма доказана.

Для произвольной булевой функции f (так же, как и для формулы) $n(f)$ будет обозначать число ее переменных, а $f|_A$ — функцию, получающуюся из f при подстановке A. Положим

$$D(f) = \max n(f|_A),$$

где максимум берется по всем 1-центральным подстановкам A, при которых функция $f|_A$ линейна (с равным успехом макси-

---

[1]) Известно, что $\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6}$, однако для наших целей достаточно и той оценки, которую можно получить более простыми средствами:

$$\sum_{k=1}^{\infty} \frac{1}{k^2} = 1 + \sum_{k=2}^{\infty} \frac{1}{k^2} < 1 + \sum_{k=2}^{\infty} \frac{1}{k(k-1)} = 1 + \lim_{n\to\infty} \sum_{k=2}^{n} \frac{1}{k(k-1)} =$$

$$= 1 + \lim_{n\to\infty} \sum_{k=2}^{n} \left(\frac{1}{k-1} - \frac{1}{k}\right) = 1 + \lim_{n\to\infty} \left(\sum_{k=1}^{n-1} \frac{1}{k} - \sum_{k=2}^{n} \frac{1}{k}\right) =$$

$$= 1 + \lim_{n\to\infty} \left(1 - \frac{1}{n}\right) = 2.$$

мум можно было бы брать по 0-центральным подстановкам). Число $D(f)$ будем называть линейным диаметром функции f.

Теорема. *Для любой булевой функции f от n переменных справедливо неравенство*

$$L^0_{\text{Б}_2}(f) \geqslant \frac{n}{2} \log_K \frac{n}{D(f)}.$$

Доказательство. Рассмотрим произвольную минимальную формулу F в базисе Б_2 для функции f. Тогда

$$L^0_{\text{Б}_2}(f) = L(F). \tag{43}$$

Упорядочим переменные функции f так, чтобы раньше шли переменные, входящие в формулу F большее число раз. Обозначим через r число вхождений в формулу F переменной, занимающей место с номером $[n/2]+1$. Поскольку каждая из предыдущих переменных входит в формулу F не менее r раз,

$$L(F) \geqslant \left(\left[\frac{n}{2}\right]+1\right) r. \tag{44}$$

Для оценки r в этом неравенстве мы воспользуемся леммой 2.

Сделаем 1-центральную подстановку B, фиксирующую первые $[n/2]$ переменных функции f. При этом мы получим формулу $F|_B$, в которую каждая из оставшихся переменных входит не более r раз, причем

$$n(F|_B) = n - \left[\frac{n}{2}\right]. \tag{45}$$

Применяя к формуле $F|_B$ лемму 2, мы видим, что существует такая центральная подстановка A, что формула $F|_{BA}$ реализует линейную функцию и

$$n(F|_{BA}) \geqslant \frac{2}{K^r} n(F|_B), \tag{46}$$

причем подстановку BA можно сделать 1-центральной.

Наконец, в силу определения линейного диаметра функции

$$D(f) \geqslant n(F|_{BA}). \tag{47}$$

Остается свести полученные соотношения в одно. Из (47), (46) и (45) следует, что

$$D(f) \geqslant \frac{2}{K^r}\left(n - \left[\frac{n}{2}\right]\right) \geqslant \frac{n}{K^r},$$

откуда

$$r \geqslant \log_K \frac{n}{D(f)}.$$

Подставляя эту оценку в (44), учитывая, что $[n/2]+1 > n/2$, и пользуясь (43), получаем неравенство теоремы.

Эта теорема дает нижнюю оценку сложности формул в базисе $Б_2$ порядка $n \log n$, если с ростом n линейный диаметр $D(f)$ растет не слишком быстро. Особенно удобно применять теорему к симметрическим функциям. Например, для элементарной симметрической функции $S_n^{[n/2]}$ имеем

$$D\left(S_n^{[n/2]}\right) = \begin{cases} 1, & \text{если } n \text{ нечетно,} \\ 2, & \text{если } n \text{ четно,} \end{cases}$$

и, таким образом,

$$L^0_{Б_2}\left(S_n^{[n/2]}\right) \gtrsim n \log_2 n.$$

Аналогично, для монотонной симметрической функции (функции голосования) $S_n^{[n/2]+1, \ldots, n}$ имеем

$$D\left(S_n^{[n/2]+1, \ldots, n}\right) = 1$$

и, таким образом,

$$L^0_{Б_2}\left(S_n^{[n/2]+1, \ldots, n}\right) \gtrsim n \log_2 n$$

Вообще Фишер, Мейер и Патерсон показали, что для почти всех симметрических функций S_n

$$L^0_{Б_2}(S_n) \gtrsim n \log_2 n.$$

ЛИТЕРАТУРА

[1] Августинович С. В. Об одном подходе к получению нижних оценок сложности для булевых функций. Сб. «Дискретный анализ», вып. 35. — Новосибирск, 1980, с. 3—8.

[2] Бенеш В. Э. Математические основы теории телефонных сообщений. — М.: Связь, 1968.

[3] Горелик Е. С. О сложности реализации элементарных конъюнкций и дизъюнкций в базисе $\{x/y\}$. Сб. «Проблемы кибернетики», вып. 26. — М.: Наука, 1973, с. 27—36.

[4] Григорьев Д. Ю. Об одной нижней оценке сложности вычисления семейства дизъюнкций в монотонном базисе. — Зап. научн. семинаров ЛОМИ АН СССР 68, 1977, с. 19—25.

[5] Клосс Б. М., Малышев В. А. Оценки сложности некоторых классов функций. — Вестник Моск. ун-та, сер. матем., мех. № 4, 1965, 44—51.

[6] Клосс Б. М. Оценки сложности решения систем линейных уравнений. — ДАН СССР 171, № 4, 1966, с. 781—783.

[7] Кричевский Р. Е. О сложности параллельно-последовательных контактных схем, реализующих одну последовательность булевых функций. Сб. «Проблемы кибернетики», вып. 12. — М.: Наука, 1964, с. 45—55.

[8] Кричевский Р. Е. Минимальная схема из замыкающих контактов для одной булевой функции от n аргументов. Сб. «Дискретный анализ», вып. 5.— Новосибирск, 1965, с. 89—92.

[9] Лупанов О. Б. Об одном методе синтеза схем. Изв. вузов, Радиофизика № 1, 1958, с. 120—140.

[10] Лупанов О. Б. О сравнении сложности реализации монотонных функций

контактными схемами, содержащими лишь замыкающие контакты, и произвольными контактными схемами. — ДАН СССР 144, № 6, 1962, с. 1245—1248.

[11] Лупанов О. Б. О методах получения оценок сложности и вычисления индивидуальных функций. — Сб. «Дискретный анализ», вып. 25. — Новосибирск, 1974, с. 3—18.

[12] Малышев В. А. Класс «почти всех» функций с нелинейной сложностью при реализации П-схемами. — Сб. «Проблемы кибернетики», вып. 19. — М.: Наука, 1967, с. 299—306.

[13] Маматов Ю. А. Об одном принципе получения нижних оценок сложности формул. — ДАН СССР 245, № 4, 1979, с. 782—784.

[14] Марков А. А. Об инверсионной сложности систем функций. — ДАН СССР 116, № 6, 1957, с. 917—919.

[15] Марков А. А. Об инверсионной сложности системы булевых функций. — ДАН СССР 150, № 3, 1963, с. 477—479.

[16] Марченков С. С. О сложности вычисления экспоненты. — Матем. заметки 31, № 3, 1982, с. 457—463.

[17] Михайлюк М. В. О сложности вычисления элементарных симметрических функций в конечных полях. — ДАН СССР 244, № 5, 1979, 1072—1076.

[18] Нечипорук Э. И. О сложности схем в некоторых базисах, содержащих нетривиальные элементы с нулевыми весами. — Сб. «Проблемы кибернетики», вып. 8. — М.: Физматгиз, 1962, с. 123—160.

[19] Нечипорук Э. И. Об одной булевской функции. — ДАН СССР 169, № 4, 1966 с. 765—766.

[20] Нечипорук Э. И. Об одной булевской матрице. — Сб. «Проблемы кибернетики», вып. 21. — М.: Наука, 1969, с. 237—240.

[21] Нечипорук Э. И. О реализации дизъюнкции и конъюнкции в некоторых монотонных базисах. — Сб. «Проблемы кибернетики», вып. 23. — М.: Наука, 1970, с. 291—293.

[22] Нигматуллин Р. Г. Проблема нижних оценок сложности и теория *NP*-полноты. — Изв. вузов, Матем. № 5, 1981, с. 17—25.

[23] Нигматуллин Р. Г. Сложность булевых функций. — Казань, Изд. Каз. ун-та, 1983.

[24] Новиков С. В., Комиссаров В. Э., Супрун В. П. Минимальная реализация функции $f = x_1x_2 \ldots x_r$ схемами из элементов шефферовского типа, — Вестник Белорус. ун-та, сер. 1, № 2, 1975, с. 13—17.

[25] Новиков Я. А. О сложности и глубине минимальной реализации конъюнкций схемами из элементов шефферовского типа. — Редкол. ж. Изв. АН БССР, сер. физ.-мат. наук, Минск, 1980. (Рукопись деп. в ВИНИТИ 11 февр. 1980 г. № 480—80 Деп.).

[26] Окольнишникова Е. А. О роли отрицаний при реализации монотонных булевых функций формулами в базисе $\{\vee, \&, \neg\}$. — Сб. «Дискретный анализ», вып. 33. — Новосибирск, 1979, с. 68—76.

[27] Офман Ю. П. Универсальный автомат. — Труды Моск. матем. об-ва, 14. — М.: Изд. МГУ, 1965, с. 186—199.

[28] Пулатов А. К. О сложности реализации характеристических функций плотно упакованных $(n, 3)$-кодов в классе П-схем. — Сб. «Дискретный анализ», вып. 22. — Новосибирск, 1973, с. 53—56.

[29] Пулатов А. К. О геометрических свойствах и схемной реализации подгрупп в E^n. — Сб. «Дискретный анализ», вып. 23. — Новосибирск, 1973, с. 32—37.

[30] Пулатов А. К. Нижняя оценка сложности схемной реализации для одного класса кодов. — Сб. «Дискретный аналлиз», вып. 25. — Новосибирск, 1974, с. 56—61.

[31] Редькин Н. П. Доказательство имнимальности некоторых схем из функциональных элементов. — Сб. «Проблемы кибернетики», вып. 23. — М.: Наука, 1970, с. 83—101.

[32] Редькин Н. П. О минимальной реализации линейной функции схемой из функциональных элементов. — Кибернетика № 6, 1971, с. 31—38.
[33] Редькин Н. П. О минимальной реализации двоичного сумматора — Сб. «Проблемы кибернетики», вып. 38. — М.: Наука, 1981, с. 181—216.
[34] Рохлина М. М. О схемах, повышающих надежность. — Сб. «Проблемы кибернетики», вып. 23. — М.: Наука, 1970, с. 295—301.
[35] Слисенко А. О. Сложностные задачи теории вычислений. — УМН 36, № 6, 1981, с. 21—103.
[36] Сопруненко Е. П. О минимальной реализации некоторых функций схемами из функциональных элементов. — Сб. «Проблемы кибернетики», вып. 15. — М.: Наука, 1965, с. 117—134.
[37] Субботовская Б. А. О реализации линейных функций формулами в базисе $\vee$, &, $^-$, — ДАН СССР 136, № 3, 1961, с. 553—555.
[38] Субботовская Б. А. О сравнении базисов при реализации функций алгебры логики формулами — ДАН СССР 149, № 4, 1963, с. 784—787.
[39] (Субботовская Б. А.) Мучник Б. А. Оценка сложности реализации линейной функции формулами в некоторых базисах. — Кибернетика, № 4, 1970, с. 29—38.
[40] Ткачев Г. А. О сложности реализации одной последовательности функций k-значной логики. — Вестник Моск. ун-та, сер. вычисл. матем. киб. № 1, 1977, с. 45—57.
[41] Ткачев Г. А. О влиянии базиса на поведение функции Шеннона.— Сб. «Дискретный анализ», вып. 34. — Новосибирск, 1980, с. 88—99.
[42] Улиг Д. Об одной функции алгебры логики, имеющей много подфункций и небольшую сложность реализации. — Сб. «Проблемы кибернетики», вып. 35. — М.: Наука, 1979, с. 133—139.
[43] Храпченко В. М. О сложности реализации линейной функции в классе П-схем. — Матем. заметки 9, № 1, 1971, с. 35—40.
[44] Храпченко В. М. Об одном методе получения нижних оценок сложности П-схем. — Матем. заметки 10, № 1, 1971, с. 83—92.
[45] Храпченко В. М. Квадратичная нижняя оценка сложности, основанная на непрерывности второй производной. — Сб. — «Проблемы кибернетики», вып. 26. — М.: Наука, 1973, с. 203—206.
[46] Храпченко В. М. О сложности реализации симметрических функций алгебры логики формулами в конечных базисах. — Сб. «Проблемы кибернетики», вып. 31. — М.: Наука, 1976, с. 231—234.
[47] Шоломов Л. А. Об одной последовательности сложно реализуемых функций. — Матем. заметки 17, № 6, 1975, с. 957—966.
[48] Яблонский С. В. Об алгоритмических трудностях синтеза минимальных контактных схем. — Сб. «Проблемы кибернетики», вып. 2. — М.: Физматгиз, 1959, 75—121.
[49] Blum N. A Boolean function requiring $3n$ network size. — Theoret. Comput. Sci. 28, N 3, 1984, 337—345.
[50] Burks A. W., McNaughton R., Pollmar C., Warren D. W., Wright J. B. Complete decoding nets: general theory and minimality, J. Soc. Industr. and Appl. Math. 2, № 4, 1954, 201—243 [Имеется перевод: Беркс А. В., Мак-Нотон Р., Полмар К. Х., Уоррен Д. В., Райт Дж. Б. Полностью декодирующие сети. Общая теория и минимальность. — Киб. сб., вып. 6. — М.: ИЛ, 1963, 91—138.]
[51] Clos C., A study of non-blocking switching networks, BSTJ 32, № 2, 1953, 406—424.
[52] Erdös P., Rényi A., On random matrices, Publ. Math. Inst. Hung. Acad. Sci. 8, N 3, 1963, 455—461.
[53] Ehrenfeucht A., Practical decidability, Report CU-CS-008-72, Department of Computer Science, University of Colorado, 1972.
[54] Fischer M. J., Meyer A. R., Paterson M. S., Lower bounds on the size of

Boolean formulas, preliminary report, Proc. 7th Annual ACM Symposium on Theory of Computing, Albuquerque, 1975, 37—44.
[55] Fischer M. J., Meyer A. R., Paterson M. S., $\Omega(n \log n)$ lower bounds on length of Boolean formulas, SIAM J. Comput. 11, № 3, 1962, 416—427.
[56] Gilbert E. N., Lattice theoretic properties of frontal switching functions, J. Math. Phys. 33, № 1, 1954, 57—67. [Имеется перевод: Гилберт Э. Н. Теоретико-структурные свойства замыкающих переключательных функций. — Киб. сб., вып. I — М.. ИЛ, 1960, 175—188.]
[57] Hansel G., Nombre minimal de contacts de fermeture nécessaires pour réaliser une fonction booléenne symétrique de n variables, C. R. Acad. Sci. Paris, 258, № 25, Groupe 1, 1964, 6037—6040. [Имеется перевод: Ансель Ж. Минимальное число замыкающих контактов, достаточное для реализации одной симметрической булевой функции n переменных. — Киб сб., вып. 5, н. с. — М.: Мир, 1968, 47—52.]
[58] Hansel G., Nombre de lettres nécessaire pour écrire une fonction symétrique de n variables, C. R. Acad. Sci., Paris, 261, № 21, Groupe 1, 1965, 4297—4300
[59] Harper L. H., An $n \log n$ lower bound on synchronous combunational complexity, Proc. Amer Math. Soc. 64, № 2, 1977, 300—306.
[60] Harper L. H., Hsieh W. N., Savage J. E., A class of Boolean functions with linear combinational complexity, Theoret. Comput. Sci. 1, № 2, 1975, 161—183.
[61] Harper L. H., Savage J. E., On the complexity of the marriage problem, Adv. Math. 9, № 3, 1972, 299—312.
[62] Harper L. H., Savage J. E., Complexity made simple, Colloquio Internazionale sulle Teorie Combinatorie, Roma, 1973, Tomo II, Atti dei Convegni Lincei N 17, Roma, Accad. Naz. Lincei., 1976, 253—262.
[63] Harper L. H., Savage J. E., Lower bounds on synchronous combinational complexity, SIAM J. Comput. 8, № 2, 1979, 115—119.
[64] Hodes L., The logical complexity of geometric properties in the plane, J. ACM 17, № 2, 1970, 339—347.
[65] Hodes L., Specker E., Lengths of formulas and elimination of quantifiers, Contributions to mathematical logic, North Holland, Amsterdam, 1968, 175—188. [Имеется перевод: Ходес Л., Шпекер Е. Длины формул и исключение кванторов. — Киб. сб., вып. 10, н. с. — М.: Мир, 1973, 99—113.]
[66] Hsieh W. N., Intersection theorems for systems of finite vector spaces and other combinatorial results, Ph. D. Thesis, Department of Mathematics, MIT, Cambridge, Mass., 1974.
[67] Lamagna E. A., The complexity of monotone networks for certain bilinear forms, routing problems, sorting and merging, IEEE Trans. Comput. 28, № 10, 1979, 773—782.
[68] Lamagna E. A., Savage J. E., Combinational complexity of some monotone functions, Proc. 15th Annual IEEE Symposium on Switching and Automata Theory, New Orleans, 1974, 140—144.
[69] Mehlhorn K., Some remarks on Boolean sums, Acta Informatica 12, № 4, 1979, 371—375. [Имеется перевод: Мельхорн К. Некоторые замечания, касающиеся булевых сумм. — Киб. сб., вып. 18, н. с. — М.: Мир, 1981, 39—45.]
[70] Mehlhorn K., An improved lower bound on the formula complexity of context-free recognition, Elektron. Informationsverarb. und Kybern. 12, № 11—12, 1976, 523—524.
[71] Mehlhorn K., Galil Z., Monotone switching circuits and Boolean matrix product, Computing 16, № 1—2, 1976, 99—111.
[72] Meyer A. R., 6.853 Lecture Notes, Department of Electrical Engineering, MIT, Cambridge, Mass., 1974.
[73] Muller D. E., Complexity in electronic switching circuits, IRE Trans. Electron. Comput. 5, № 1, 1956, 15—19.
[74] Paterson M. S., Complexity of monotone networks for Boolean matrix pro-

duct, Theoret. Comput. Sci. 1, № 1, 1975, 13—20. [Имеется перевод: Патерсон М. С. Сложность монотонных схем для булева умножения матриц. — Киб. сб., вып. 15, н. с. — М.: Мир, 1978, 28—37.]
[75] Paterson M. S., An introduction to Boolean function complexity, Asterisque, № 38—39, 1976, 183—201.
[76] Paterson M. S., New bounds on formula size, Lecture Notes in Computer Science 48, Springer-Verlag, New York, 1977, 17—26.
[77] Paul W. J., A 2,5 n-lower bound on the combinational complexity of Boolean functions, SIAM J. Comput. 6, № 3, 1977, 427—443. [Имеется перевод: Пауль В. Й. Нижняя оценка 2,5n для комбинационной сложности булевых функций. — Киб. сб., вып. 16, н. с. — М.: Мир, 1979, 23—44.]
[78] Pippenger N., On another Boolean matrix, IBM Research Report 6914, 1977.
[79] Pratt V. R., The power of negative thinking in multiplying Boolean matrices, SIAM J. Comput. 4, № 3, 1975, 326—330.
[80] Pudlák P., Bounds for Hodes — Specker theorem, Lecture Notes in Computer Science 171, Springer-Verlag, 1984, 421—445.
[81] Riordan J., Shannon C. E., The number of two-terminal series-parallel networks, J. Math. Phys. 21, № 2, 1942, 83—93. [Имеется перевод: Риордан Дж., Шеннон К. Э. Число двухполюсных параллельно-последовательных сетей. — Шеннон К. Работы по теории информации и кибернетике. — М.: ИЛ, 1963, 46—58.]
[82] Savage J. E., The complexity of computing, Wiley-Interscience, N.—Y., 1976.
[83] Schnorr C. P., Zwei lineare untere Schranken für die Komplexität Boolescher Funktionen, Computing 13, № 2, 1974, 155—171.
[84] Schnorr C. P., The combinational complexity of equivalence, Theoret. Comput. Sci. I, № 4, 1976, 289—295. [Имеется перевод: Шнорр К. П. Комбинационная сложность эквивалентности. — Киб. сб. вып. 16, н. с. — М.: Мир, 1979, 74—81.]
[85] Schnorr C. P., A lower bound on the number of additions in monotone computations, Theoret. Comput. Sci. 2, № 3, 1976, 305—315. [Имеется перевод: Шнорр К. П. Нижняя оценка числа сложений в монотонных вычислениях. — Киб. сб., вып. 18, н. с. — М.: Мир, 1981, 5—20.]
[86] Schnorr C. P., A 3n-lower bound on the network complexity of Boolean functions, Theoret. Comput. Sci. 10, № 1, 1980, 83—92. [Имеется перевод: Шнорр К. П. Нижняя оценка 3n для комбинационной сложности булевых функций. — Киб. сб., вып. 18, н. с. — М.: Мир, 1981, 21—33.]
[87] Schönhage A., An elementary proof for Strassen's degree bound, Theoret. Comput. Sci. 3, № 2, 1976, 267—272. [Имеется перевод: Шёнхаге А. Элементарное доказательство оценки Штрассена. — Киб. сб., вып. 15, н. с. — М.: Мир, 1978, с. 22—27.]
[88] Specker E., Elimination von Quantoren und Länge von Formeln (Abstract), J. Symb. Logic 32, № 4, 1967, 567—568.
[89] Stockmeyer L. J., The complexity of decision problems in automata theory and logic, Project MAC, Technical Report 133, MIT, Cambridge, Mass., 1974.
[90] Stockmeyer L. J., On the combinational complexity of certain symmetric Boolean functions, Math. Systems Theory 10, № 4, 1976/1977. 323—336. [Имеется перевод: Стокмейер Л. Дж. О комбинационной сложности некоторых симметрических булевых функций. — Киб. сб., вып. 16, н. с. — М.: Мир, 1979, с. 45—61.]
[91] Strassen V., Die Berechnungskomplexität von elementar symmetrischen Funktionen und von Interpolationskoeffizienten, Numer. Math. 20, № 3, 1973, 238—251. [Имеется перевод: Штрассен Ф. Сложность вычисления элементарных симметрических функций и коэффициентов интерполяционного полинома. — Киб. сб., вып. 15, н. с. — М.: Мир, 1978, 5—21.]

[92] Tarjan R. E., Complexity of monotone networks for computing conjunctions, Ann. Discrete Math. 2, 1978, 121—133.
[93] Valiant L. G., On non-linear lower bounds in computational complexity, Proc. 7th Annual ACM Symposium on Theory of Computing, Albuquerque, 1975, 45—53.
[94] Vilfan B., A generalization of a theorem of Specker and some applications, Automata, Languages, and Programming, North Holland, Amsterdam, 1973, 609—622.
[95] Vilfan B., Lower bounds for the size of expressions for certain functions in d-ary logic, Theoret. Comput. Sci. 2, № 2, 1976, 249—269.
[96] Wegener I., Switching functions whose monotone complexity is nearly quadratic, Theoret. Comput. Sci. 9, № 1, 1979, 83—97. [Имеется перевод: Вегенер И. Булевы функции, чья монотонная сложность почти квадратична. — Киб. сб., вып. 18, н. с. — М.: Мир, 1981, с. 55—74.]
[97] Wegener I., Boolean functions whose monotone complexity is of size $n^2/\log n$, Theoret. Comput. Sci. 21, № 2, 1982, 213—224. [Имеется перевод: настоящий сборник, с. 71.]

Нахождение пересечения n полупространств за время $O(n \log n)$[1]

Ф. Препарата, Д. Маллер[2])

Предлагается алгоритм, который по заданному семейству n полупространств трехмерного пространства строит их общую часть за время $O(n \log n)$. Пересечение, в случае его непустоты, представляется в виде выпуклого многогранника. Указанный алгоритм состоит в следующем: (1) полупространства подразделяются на два подсемейства в зависимости от того, содержат ли они начало координат, или не содержат его;

(2) полупространства, образующие каждое из этих двух подсемейств, преобразуются в двойственные им точки, и строятся выпуклые оболочки полученных таким образом множеств точек — это делается за время $O(n \log n)$; (3) поскольку пересечение полупространств непусто в том и только в том случае, когда указанные выпуклые оболочки отделимы, отыскивается отделяющая плоскость — также за время $O(n \log n)$; (4) применяется линейное преобразование пространства, отображающее отделяющую плоскость на бесконечность, при котором образом искомого пересечения полупространств является выпуклая оболочка объединения образов двух построенных выпуклых оболочек; нахождение этих образов, их выпуклой оболочки и применение обратного преобразования выполняются за время $O(n)$. Таким образом, общее время работы алгоритма есть $O(n \log n)$. Важным следствием этого результата является возможность решать задачи линейного и выпуклого программирования с тремя переменными асимптотически быстрее, чем при помощи симплекс-алгоритма (в худшем случае).

1. ВВЕДЕНИЕ

Нахождение общей части семейства n полупространств трехмерного пространства — важная задача вычислительной геометрии. Пожалуй, наиболее известное из применений этой задачи встречается в математической теории оптимизации, в частности, в линейном и выпуклом программировании с тремя переменными [1]. В самом деле, как известно, задача линейного программирования в переменных (x, y, z) состоит в отыскании экстремума линейной целевой функции от переменных x, y, z, подчиненных системе n линейных неравенств. Эти неравенства определяют множество допустимых решений рассматриваемой

[1]) Preparata F. P., Muller D. E., Finding the intersection of n half-spaces in time $O(n \log n)$, Theoret. Comput. Sci. 8 (1979), 45—55.

[2]) Coordinated Science Laboratory, University of Illinois at Urbana-Champaign, Urbana.

задачи. Хорошо известно, что указанное множество представляет собой выпуклую область пространства, являющуюся пересечением соответствующих неравенствам полупространств; известно также, что экстремальное решение достигается в некоторой вершине, или *крайней точке*, области допустимых решений (*допустимой области*). Таким образом задача линейного программирования может быть решена путем нахождения крайних точек допустимой области и вычисления в каждой из них значения целевой функции. Те же соображения полностью применимы к выпуклому программированию с линейными ограничениями, когда характер выпуклости целевой функции таков, что ее экстремальное значение достигается в крайней точке допустимой области.

Наиболее успешным методом решения этих оптимизационных задач является симплекс-алгоритм [1]. Для случая трех переменных симплекс-алгоритм осуществляет продвижение вдоль пути, пролегающего по многогранной поверхности, ограничивающей допустимую область, вплоть до достижения точки экстремума. Поскольку симплекс-алгоритм затрачивает время $O(n)$ на перемещение из вершины в вершину, и в худшем случае должен побывать в $O(n)$ вершинах прежде, чем закончит работу, мы заключаем, что общее время его работы есть $O(n^2)$.

Для случая двух переменных Шамос и Хой [5] показали, что пересечение n полуплоскостей может быть найдено за время $O(n \log n)$, получив тем самым и метод решения задачи линейного программирования с двумя переменными, работающий быстрее симплекс-алгоритма в худшем случае. Их метод основан на приеме «разделяй и властвуй», а именно на нахождении за время $O(n)$ пересечения двух многоугольников, являющихся соответственно общими частями двух подсемейств, содержащих приблизительно по $n/2$ полуплоскостей.

Шамос [4] высказал также предположение, что если будет найден быстрый алгоритм нахождения пересечения двух многогранников, то его можно будет использовать (с привлечением приема «разделяй и властвуй») для построения быстрого алгоритма нахождения пересечения семейства полупространств. В свою очередь, последний может быть применен к задаче линейного программирования. В предшествующей работе [2] мы описали алгоритм нахождения пересечения двух многогранников с общим числом вершин n за время $O(n \log n)$. Этот метод, будучи использован как способ слияния при подходе «разделяй и властвуй», дал бы для нахождения пересечения n полупространств время работы $O(n \log^2 n)$. В данной работе показано, что последняя задача может быть, тем не менее, решена за время $O(n \log n)$. Наш способ существенно опирается на алгоритм нахождения пересечения многогранников, но не как на

способ слияния; вместо этого он использует его для преобразования исходной задачи в двойственную, т. е. в задачу нахождения выпуклой оболочки множества n точек в трехмерном пространстве, которая, как известно, решается (приемом «разделяй и властвуй») за время $O(n \log n)$.

Моделью вычислений, принятой в предшествующих алгоритмах, служила машина со свободным доступом к памяти, использующая арифметику действительных чисел. Мы и в дальнейшем будем придерживаться этой модели.

Дадим теперь точную формулировку нашей задачи. Нахождение пересечения n полупространств трехмерного пространства состоит в нахождении решений системы n линейных неравенств вида

$$a_{i1}x + a_{i2}y + a_{i3}z + a_{i4} \geqslant 0 \quad (i = 1, \ldots, n), \qquad (1)$$

где x, y и z — декартовы координаты решения в пространстве трех измерений, и где для каждого i коэффициенты a_{i1}, a_{i2}, a_{i3}, a_{i4} — действительные числа, одновременно не обращающиеся в 0.

В силу соображений, которые будут объяснены позже, удобно изображать точки с помощью однородных координат x_1, x_2, x_3, x_4, так что $x = x_1/x_4$, $y = x_2/x_4$ и $z = x_3/x_4$. Такому описанию соответствует система неравенств

$$a_{i1}x_1 + a_{i2}x_2 + a_{i3}x_3 + a_{i4}x_4 \geqslant 0 \quad (i = 1, \ldots, n), \qquad (2)$$

из которой можно получить все решения исходной системы. На самом деле решения системы (2) образуют выпуклое множество, которое может быть разбито на три непересекающихся подмножества, каждое из которых также выпукло. Это, во-первых, множество *положительных* точек, для которых $x_4 > 0$, и которые соответствуют решениям системы (1), во-вторых, множество *экваториальных* точек, для которых $x_4 = 0$, и, в-третьих, множество *отрицательных* точек, для которых $x_4 < 0$.

Множество решений системы (2) будет описываться при помощи систем данных следующего вида:

(а) Минимальной подсистемы системы (2) (т. е. такой, из которой исключены все неравенства-следствия), соответствующей граням F обобщенного многогранника A, определение которого будет дано позже.

(б) Минимального множества V' крайних точек (x_1, x_2, x_3, x_4). (Множество решений системы (2) состоит из линейных комбинаций элементов V' с неотрицательными коэффициентами. Отвечающие элементам V' точки трехмерного пространства, изображаемые в неоднородных координатах, образуют множество V вершин обобщенного многогранника A.)

(в) Структурой данных (*реберным списком с двойными связями*, см. [2]), описывающей связи, имеющиеся между F и V,

и содержащей циклические списки ребер, инцидентных вершинам из V, и циклические списки ребер, окружающих грани из F.

Одно из преимуществ рассмотрения решений системы (2) состоит в том, что в этом случае все крайние точки лежат в области конечных значений переменных, в то время как крайние точки (1) могут лежать на бесконечности. Другое преимущество, как мы увидим, состоит в том, что двойственность между V и F в случае (2) является полной, и мы сумеем воспользоваться этим в полной мере.

2. ГЕОМЕТРИЧЕСКОЕ ВСТУПЛЕНИЕ

Как известно, имеется удобная интерпретация однородных координат в пространстве E^3, при которой наборы (x_1, x_2, x_3, x_4) рассматриваются как координаты в четырехмерном пространстве E^4, нормализованные путем умножения на положительные числа, так что $x_1^2 + x_2^2 + x_3^2 + x_4^2 = 1$, т. е. как координаты точек на поверхности единичной гиперсферы S^4 с центром в начале координат. Положительные точки лежат в той ее части, которую мы будем называть положительной открытой полусферой и которая состоит из всех тех точек гиперсферы S^4, для которых $x_4 > 0$; аналогичным образом, отрицательные точки лежат в отрицательной полусфере, а экваториальные точки лежат в пересечении гиперсферы с экваториальной гиперплоскостью $x_4 = 0$.

Трехмерное пространство, содержащее решения системы (1), может рассматриваться как гиперплоскость $x_4 = 1$, касательная к S^4, а центральная проекция с центром в начале координат устанавливает взаимно однозначное соответствие между точками положительной полусферы и точками пространства E^3. Каждая из гиперплоскостей, отвечающих неравенствам системы (2), проходит через начало координат пространства E^4 и определяет соответствующее полупространство; пересечение этих полупространств представляет собой выпуклый конус C^4. Пересечение конуса C^4 с гиперсферой S^4 представляет собой связную область, которая может пересекать экваториальную гиперплоскость; в последнем случае точки пересечения $C^4 \cap S^4$, попавшие в положительную полусферу, будут проецироваться в точки положительного множества (см. ниже), в то время как точки пересечения $C^4 \cap S^4$, попавшие в отрицательную полусферу, будут проецироваться в множество точек, лежащих на гиперплоскости $x_4 = -1$, которые в свою очередь будут отображаться на отрицательное множество в гиперплоскости $x_4 = 1$ путем центральной симметрии относительно начала координат пространства E^4 (рис. 1 (b) иллюстрирует сказанное в пространстве на единицу меньшего числа измерений).

Построенное таким образом множество, лежащее в гиперплоскости $x_4 = 1$, может, тем самым, состоять из двух отдельных неограниченных выпуклых подмножеств — *положительного* и *отрицательного*, как показано на рис. 1 (а) для случая двух измерений. Мы назовем это *гиперболическим* случаем. Если все точки решения проецируются в положительную (либо отрицательную) открытую полусферу, то соответствующее подмноже-

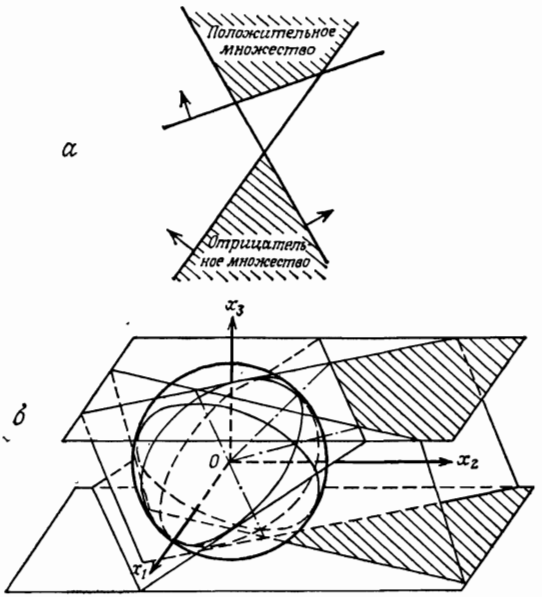


Рис. 1. Изображение положительного и отрицательного множеств для случая двух измерений. Стрелки указывают полуплоскости, отвечающие неравенствам системы (1).

ство пространства E^3 ограничено, и мы назовем это *эллиптическим* случаем. *Параболический* случай возникает, когда имеется лишь одно, но неограниченное подмножество E^3, т. е. когда пересечение $C^4 \cap S^4$ содержит наряду с положительными (либо отрицательными) точками экваториальные точки. Множества, возникающие во всех трех случаях, мы будем называть *обобщенными* выпуклыми многогранниками. Из них только эллиптическому случаю соответствуют обычные многогранники.

В дальнейшем мы часто будем использовать инволютивное преобразование, сопоставляющее точкам плоскости и наоборот, называемое *преобразованием двойственности*. При преобразовании двойственности координаты точки $(\xi_1, \xi_2, \xi_3, \xi_4)$ трактуются как коэффициенты, определяющие полупространство

$\xi_1 x_1 + \xi_2 x_2 + \xi_3 x_3 + \xi_4 x_4 \geqslant 0$. Преобразование двойственности поддается двум интуитивным геометрическим интерпретациям — в пространстве E^4 и в пространстве E^3. При интерпретации в E^4 каждая точка на гиперсфере S^4 рассматривается как конец (единичного) вектора, приложенного к началу координат, и двойственным к ней считается полупространство, содержащее этот вектор, и ограниченное гиперплоскостью, проходящей через начало координат ортогонально к указанному вектору. И наоборот, каждому полупространству, ограниченному гиперплоскостью, проходящей через начало координат, ставится в соответствие двойственная ему точка, являющаяся концом соответствующего вектора. При интерпретации в E^3 каждой положительной точке, находящейся на расстоянии l от начала координат, ставится в соответствие полупространство, содержащее начало координат, граничная гиперплоскость которого расположена на расстоянии $1/l$ от начала координат в направлении, противоположном по отношению к рассматриваемой точке. Если точка отрицательна, берется дополнение к полупространству с той же границей. И наоборот, каждому полупространству, чья граничная плоскость не проходит через начало координат, ставится в соответствие двойственная ему точка. Эта точка трактуется как положительная или отрицательная в зависимости от того, содержит ли указанное полупространство начало координат.

После перехода к двойственной системе неравенств, нужно будет применить какой-либо из разнообразных алгоритмов построения выпуклой оболочки множества точек, и вслед за тем совершить обратное преобразование двойственности. Поскольку алгоритмы построения выпуклых оболочек в их опубликованном варианте [3] применяются к обычному, или эллиптическому случаю, будет удобно сначала применить к системе (2) обратимое линейное преобразование координат в пространстве E^4. Осуществление этого преобразования вызовет перемещение начала координат пространства E^3 в избранную точку, и тем самым позволит свести рассмотрение двойственной системы к желательному для нас эллиптическому случаю. Указанное преобразование будет описываться невырожденной матрицей T размера 4×4. Чтобы не создавать излишних трудностей, имеет смысл с самого начала ограничиться лишь теми линейными преобразованиями пространства E^4, которые оставляют на месте гиперсферу S^4. Охарактеризовать эти преобразования нетрудно. Изображая точку на гиперсфере S^4 вектор-строкой ξ, имеем [1]) $\xi\xi' = 1$; образ ξT вектора ξ должен также лежать на S^4, откуда имеем

$$(\xi T)(\xi T)' = \xi T T' \xi' = 1.$$

[1]) Здесь ξ' обозначает вектор-столбец, транспонированный к ξ.

В силу произвола выбора ξ, последнее соотношение выполняется тогда и только тогда, когда $TT' = I_4$, где I_4 — единичная матрица размера 4×4; стало быть, $T^{-1} = T'$. Это — характеристическое свойство *вращений*. Итак, мы будем рассматривать только вращения E^4. Стоит отметить, что вращения коммутируют с преобразованием двойственности, т. е. при применении преобразования двойственности к образу точки, подвергнутой вращению, и при применении вращения к образу точки, подвергнутой преобразованию двойственности, получается одно и то же.

3. НАХОЖДЕНИЕ ПЕРЕСЕЧЕНИЯ

Перепишем систему (2) в матричном виде

$$A\mathbf{x}' \geqslant 0, \tag{3}$$

где $A = \|a_{ij}\|$ — матрица размера $n \times 4$. Обозначим через A_+, A_0 и A_- матрицы, образованные теми строками матрицы A, для которых соответственно $a_{i4} > 0$, $a_{i4} = 0$, и $a_{i4} < 0$. Впоследствии мы покажем, что всегда (кроме вырожденных случаев, которые могут быть сведены к задачам меньшей размерности) можно найти такое вращение R_0 матрицы A, что матрица A_0 исчезнет. Поэтому будем считать, что к системе (3) такое вращение уже применено, и что для каждого i либо $a_{i4} > 0$, либо $a_{i4} < 0$.

Каждой строке $(a_{i1}, a_{i2}, a_{i3}, a_{i4})$ матрицы A_+ отвечает плоскость в E^3 (или, равносильным образом, гиперплоскость в E^4, проходящая через начало координат). Перейдем от каждой из этих плоскостей к двойственной ей точке, т. е. построим в пространстве E^3 точки

$$(a_{i1}/a_{i4},\ a_{i2}/a_{i4},\ a_{i3}/a_{i4}),$$

и применим для нахождения выпуклой оболочки [1]). $\mathbf{A}_+^{(D)}$ множества этих точек алгоритм из работы [3] (отметим, что непосредственная применимость алгоритма оказывается возможной вследствие конечности значений координат всех рассматриваемых точек). Затем такая же процедура используется для построения выпуклого многогранника $\mathbf{A}_-^{(D)}$ по матрице A_-. Теперь мы можем применить к двум трехмерным многогранникам $\mathbf{A}_+^{(D)}$ и $\mathbf{A}_-^{(D)}$ алгоритм, описанный в работе [2], который, определяет, имеют ли эти многогранники непустое пересечение, и если да, находит это пересечение, а если нет, находит отделяющую плоскость. В действительности, если пересечение

[1]) Буква «D» в качестве верхнего индекса призвана напоминать о том, что мы имеем дело с выпуклой оболочкой множества *двойственных* точек.

$\mathbf{A}_+^{(D)} \bigcap \mathbf{A}_-^{(D)}$ имеет непустую внутренность (т. е. ненулевой объем), нам не нужно заниматься его построением, ибо в этом случае система неравенств (3) несовместна и не имеет решений. Покажем это. Пусть, без ограничения общности, $\{(a_{j1}/a_{j4},\ a_{j2}/a_{j4},\ a_{j3}/a_{j4})\colon\ 1 \leqslant j \leqslant s\}$ — множество всех вершин многогранника $\mathbf{A}_+^{(D)}$. Любую точку, принадлежащую $\mathbf{A}_+^{(D)}$, можно выразить в виде выпуклой комбинации его вершин; это можно записать в виде

$$(u_1,\ u_2,\ u_3,\ u_4,\ 1) = \sum_{j=1}^{s} c_j \left(\frac{a_{j1}}{a_{j4}},\ \frac{a_{j2}}{a_{j4}},\ \frac{a_{j3}}{a_{j4}},\ 1,\right). \tag{4}$$

Если точка (u_1, u_2, u_3) принадлежит внутренности $\mathbf{A}_+^{(D)}$, можно взять $c_j > 0$ для всех j. Если $(x_1^*,\ x_2^*,\ x_3^*,\ x_4^*)$ — произвольное ненулевое решение системы (3), то, очевидно,

$$a_{j1}x_1^* + a_{j2}x_2^* + a_{j3}x_3^* + a_{j4}x_4^* \geqslant 0,\ 1 \leqslant j \leqslant s.$$

Вместе с тем, левые части всех этих s неравенств не могут одновременно обращаться в 0 — действительно, будь это не так, все плоскости, задаваемые уравнениями $a_{j1}x + a_{j2}y + a_{j3}z + a_{j4} = 0$ пересекались бы в одной точке, в противоречии с тем, что эти плоскости двойственны вершинам многогранника $\mathbf{A}_+^{(D)}$, которые не лежат все в одной плоскости. Допустим теперь, что для некоторого m выполнено строгое неравенство $a_{m1}x_1^* + a_{m2}x_2^* + a_{m3}x_3^* + a_{m4}x_4^* > 0$. Умножая скалярно обе части неравенства (4) на вектор $(x_1^*,\ x_2^*,\ x_3^*,\ x_4^*)'$, заметим, что правая часть полученного неравенства представляет собой сумму неотрицательных слагаемых, среди которых $c_m(a_{m1}x_1^* + a_{m2}x_2^* + a_{m3}x_3^* + a_{m4}x_4^*)/a_{m4} > 0$, и, стало быть

$$u_1x_1^* + u_2x_2^* + u_3x_3^* + x_4^* > 0.$$

Если же точка (u_1, u_2, u_3) принадлежит также внутренности $\mathbf{A}_-^{(D)}$ то она должна точно так же удовлетворять и неравенству $-u_1x_1^* - u_2x_2^* - u_3x_3^* - x_4^* > 0$. Значит, система (3) не имеет решения.

Если пересечение $\mathbf{A}_+^{(D)} \bigcap \mathbf{A}_-^{(D)}$ непусто, но не имеет внутренности, рассматриваемая задача сводится к задаче меньшей размерности, которая будет обсуждена в разделе 4.

Наконец, допустим, что многогранники $\mathbf{A}_+^{(D)}$ и $\mathbf{A}_-^{(D)}$ не пересекаются. В этом случае методом работы [2] отыскивается отделяющая плоскость. Пусть $p_1x + p_2y + p_3z + p_4 = 0$ — уравнение этой плоскости, причем знаки коэффициентов выбраны так,

что для любой точки (a, b, c) многогранника $\mathbf{A}_{+}^{(D)}$ выполняется неравенство $p_1a + p_2b + p_3c + p_4 > 0$.

Найденная только что отделяющая плоскость может быть расширена до гиперплоскости $p_1x_1 + p_2x_2 + p_3x_3 + p_4x_4 = 0$, проходящей через начало координат пространства E^4 и имеющей нормалью вектор (p_1, p_2, p_3, p_4). Теперь нам хотелось бы повернуть систему координат в E^4 таким образом, чтобы этот вектор оказался направленным в начало координат пространства E^3. Иными словами, нам хотелось бы найти такое вращение R, для которого $(p_1, p_2, p_3, p_4)R = (0, 0, 0, K)$, где K — некоторое положительное число (в действительности $K = \sqrt{p_1^2 + p_2^2 + p_3^2 + p_4^2}$). Подходящее вращение легко строится в виде $R_1R_2R_3$, где каждое из вращений R_i вынуждает перейти в 0 соответствующую координату p_i. Так, например, можно взять

$$R_1 = \begin{bmatrix} p_4/\sqrt{p_1^2+p_4^2} & 0 & 0 & p_1/\sqrt{p_1^2+p_4^2} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -p_1/\sqrt{p_1^2+p_4^2} & 0 & 0 & p_4/\sqrt{p_1^2+p_4^2} \end{bmatrix}.$$

Элементы матрицы R размера 4×4 могут быть вычислены за время, ограниченное константой с использованием арифметики, включающей извлечение квадратного корня [1]).

Мы утверждаем, что все компоненты четвертого столбца матрицы AR сплошь положительны. Действительно, система (3) может быть переписана в виде

$$A\mathbf{x}' = A(RR^{-1})\mathbf{x}' = ARR'\mathbf{x}' = AR(\mathbf{x}R)' \leqslant 0. \qquad (3')$$

Поскольку точка $\mathbf{p} = (p_1, p_2, p_3, p_4)$ удовлетворяет для каждого i строгому неравенству

$$a_{i1}p_1 + a_{i2}p_2 + a_{i3}p_3 + a_{i4}p_4 > 0,$$

образ точки $\mathbf{p}$ под действием рассматриваемого вращения, т. е. вектор $(0, 0, 0, K)$, удовлетворяет соответствующей системе строгих неравенств, отвечающих строкам матрицы AR. В каждом из этих неравенств единственным ненулевым слагаемым является произведение константы K на четвертую компоненту соответствующей строки матрицы AR, что и доказывает высказанное утверждение.

---

[1]) Можно избежать появления квадратных корней, если вместо вращения R использовать линейное преобразование T более общего вида. Вместе с тем тогда необходимо подвергать векторы, представляющие грани, преобразованию, транспонированному к T^{-1}, когда векторы, представляющие вершины, подвергаются преобразованию T, и наоборот.

Вместе с тем, нет необходимости строить матрицу AR, поскольку она может содержать лишние строки, от которых мы хотели бы избавиться. Вместо этого, для каждой вершины $(a_{i1}/a_{i4},\ a_{i2}/a_{i4},\ a_{i3}/a_{i4})$ многогранника $\mathbf{A}_+^{(D)}$ строится строка $(a_{i1}, a_{i2}, a_{i3}, a_{i4})$ новой матрицы $A_+^{(m)}$, строки которой образуют подмножество строк матрицы A_+. Аналогичным образом, для каждой вершины многогранника $A_-^{(D)}$ отыскивается соответствующая строка матрицы A_-, и из этих строк формируется новая матрица $A_-^{(m)}$.

Пусть $A_+^* = A_+^{(m)}R$ и $A_-^* = A_-^{(m)}R$. Система неравенств

$$[A_+^*,\ A_-^*]\xi' \geqslant 0 \tag{5}$$

равносильна системе $AR\xi' \geqslant 0$, из которой исключены все лишние неравенства.

Поскольку системе (5) удовлетворяет вектор $(0, 0, 0, 1)$, отвечающий началу координат пространства E^3, пересечение исходных n полупространств может быть найдено следующим образом: (1) применением преобразования двойственности к плоскостям, отвечающим строкам матрицы $[A_+^*, A_-^*]$, (2) нахождением выпуклой оболочки полученного таким образом множества точек, и (3) применением обратного преобразования двойственности к полученной выпуклой оболочке. Эта последовательность операций может быть существенно упрощена. Двойственные образы плоскостей, отвечающих строкам матриц $A_+^{(m)}$ и $A_-^{(m)}$ уже известны — это вершины многогранников $\mathbf{A}_+^{(D)}$ и $\mathbf{A}_-^{(D)}$ соответственно. Таким образом, чтобы получить аналогичные многогранники $\mathbf{A}_+^{*(D)}$ и $\mathbf{A}_-^{*(D)}$ для матриц A_+^* и A_-^*, подвергнем многогранники $\mathbf{A}_+^{(D)}$ и $\mathbf{A}_-^{(D)}$, лежащие в гиперплоскости $x_4 = 1$, вращению R, а затем спроецируем их сквозь начало координат пространства E^4 на ту же гиперплоскость $x_4 = 1$. На практике это означает умножение координат каждой из подвергнутых вращению вершин на число, обращающее четвертую координату в 1.

Многогранники $\mathbf{A}_+^{*(D)}$ и $\mathbf{A}_-^{*(D)}$ не пересекаются, ибо их отделяет гиперплоскость, проходящая через начало координат E^4 нормально к R-образу вектора $(0, 0, 0, 1)$. Поэтому можно построить выпуклую оболочку объединения $\mathbf{A}_+^{*(D)} \bigcup \mathbf{A}_-^{*(D)}$ однократным применением «сливающей» части алгоритма из работы [3] за время, пропорциональное общему числу вершин двух этих многогранников. Обозначим получающийся таким образом многогранник через $\mathbf{A}^{*(D)}$.

Многогранник $\mathbf{A}^*$, двойственный к многограннику $\mathbf{A}^{*(D)}$, описывает множество решений системы неравенств

$$a^*_{i1}x + a^*_{i2}y + a^*_{i3}z + a^*_{i4} \geqslant 0 \quad (i = 1, \ldots, n,) \qquad (1')$$

где коэффициенты a^*_{ij} являются элементами матрицы $A^* = AR$; напомним, что все коэффициенты вида a^*_{i4} положительны. Поэтому применим обратное вращение R^{-1} к вершинам и граням многогранника $\mathbf{A}^*$. Многогранник $\mathbf{A}^*$ лежит целиком в гиперплоскости $x_4 = 1$, так что четырехмерное вращение R^{-1} применяется к вершинам $\mathbf{A}^*$, имеющим вид $(v^*_1, v^*_2, v^*_3, 1)$. Получающиеся таким образом вершины (v_1, v_2, v_3, v_4) должны быть нормализованы, если $v_4 \neq 0$ — это делается путем умножения каждой из координат на положительную постоянную $1/|v_4|$.

Результатом применения описанного вращения и последующей нормализации является множество вершин и связанных с ними граней, которое можно описать при помощи реберного списка с двойными связями, и которое мы будем называть обобщенным многогранником $\mathbf{A}$. Те вершины этого обобщенного многогранника, у которых четвертая координата равна 1, являются крайними точками пересечения полупространств.

В самом начале этого раздела утверждалось, что можно найти такое начальное вращение координат R_0, которое сделало бы каждый элемент a_{i4} четвертого столбца матрицы A отличным от 0. Сейчас мы опишем, как найти такое вращение R_0. Как прежде, пусть A_0 — матрица, состоящая из всех тех строк матрицы A, для которых $a_{i4} = 0$. Неравенства системы (1), отвечающие указанным строкам, имеют вид

$$a_{i1}x + a_{i2}y + a_{i3}z \geqslant 0. \qquad (1'')$$

Попробуем подыскать точку $\mathbf{u} = (u_1, u_2, u_3)$ пространства E^3, которая строго удовлетворяла бы всем этим неравенствам. Заметим, что система неравенств (1″) определяет выпуклый конус $\mathbf{C}^3$ в E^3 с вершиной в начале координат. За исключением вырожденных случаев, конус $\mathbf{C}^3$ непременно пересекает одну из плоскостей $z = 1$ или $z = -1$, ибо не может целиком располагаться в области $-1 \leqslant z \leqslant 1$. Поэтому, если точка $\mathbf{u}$ существует, она может быть найдена следующим образом. Полагая в неравенствах (1″) $z = 1$, и используя алгоритм Шамоса и Хоя [5] для нахождения за время $O(n \log n)$ пересечения n полуплоскостей, найдем соответствующий многоугольник решений. Если такого многоугольника не существует, положим $z = -1$ и повторим этот процесс. Либо в одном, либо в другом случае должен найтись многоугольник P, в котором можно выбрать в качестве искомой точки $\mathbf{u}$ одну из его внутренних точек.

Превратим теперь точку $\mathbf{u} = (u_1, u_2, u_3)$ в точку $\mathbf{v} = (u_1, u_2, u_3, M)$ пространства E^4, беря $M > 0$. Очевидно, вектор $\mathbf{v}$ строго удовлетворяет всем неравенствам системы $A_0\mathbf{v} \geqslant 0$. Как мы уже убедились при построении вращения R, можно построить такое вращение R_0, для которого $\mathbf{v}R_0 = (0, 0, 0, K_0)$, и которое обладает вдобавок тем свойством, что все элементы четвертого столбца матрицы A_0R_0 положительны.

Вращение R_0 может быть сделано сколь угодно мало отличающимся от тождественного преобразования за счет выбора достаточно большого значения M. Поэтому возьмем значение M настолько большим, чтобы ни один из элементов четвертых столбцов матриц A_+ или A_- не изменял своего знака под действием вращения R_0. Тогда все элементы четвертого столбца матрицы AR_0 будут отличны от нуля. Как отмечалось, построение точки $\mathbf{u}$ может быть выполнено за время $O(n \log n)$. Построение вращения R_0 может быть выполнено, как уже говорилось ранее в сходной ситуации, за время, ограниченное константой.

Итак, можно заменить исходную систему неравенств (3) системой

$$AR_0(\mathbf{x}R_0)' \geqslant 0. \qquad (3'')$$

Подведем итог всем предшествующим рассуждениям. Пересечение n полупространств, описываемых системой (3), может быть получено следующим образом.

Шаг 1. Найти точку u во внутренности множества решений системы (1″) и построить соответствующее вращение R_0. Заменить матрицу A матрицей AR_0. Это может быть сделано за время $O(n \log n)$ с использованием алгоритма нахождения пересечения полуплоскостей [5].

Шаг 2. Найти выпуклую оболочку $\mathbf{A}_+^{(D)}$ множества всех точек $(a_{i1}/a_{i4}, a_{i2}/a_{i4}, a_{i3}/a_{i4})$, для которых $a_{i4} > 0$. Найти выпуклую оболочку $\mathbf{A}_-^{(D)}$ множества всех тех точек $(a_{i1}/a_{i4}, a_{i2}/a_{i4}, a_{i3}/a_{i4})$, для которых $a_{i4} < 0$. Это может быть сделано за время $O(n \log n)$ с использованием алгоритма нахождения выпуклой оболочки, описанного в [3].

Шаг 3. Используя алгоритм нахождения пересечения многогранников [2], проверить, имеет ли пересечение $\mathbf{A}_+^{(D)} \bigcap \mathbf{A}_-^{(D)}$ непустую внутренность. Если да, то пересечение полупространств пусто; если нет, найти отделяющую плоскость $p_1x + p_2y + p_3z + p_4 = 0$. Это может быть сделано за время $O(n \log n)$ в соответствии с результатами, изложенными в [2].

Шаг 4. Найти вращение R пространства E^4, отображающее отделяющую плоскость на бесконечность — это может быть сделано за константу времени. Найти R-образы многогранников $\mathbf{A}_+^{(D)}$ и $\mathbf{A}_-^{(D)}$ и спроецировать их сквозь начало координат на ги-

перплоскость $x_4 = 1$ — это требует времени $O(n)$. Пусть $\mathbf{A}_+^{*(D)}$ и $\mathbf{A}_-^{*(D)}$ — полученные многогранники.

Шаг 5. Найти выпуклую оболочку $\mathbf{A}^{*(D)}$ объединения многогранников $\mathbf{A}_+^{*(D)}$ и $\mathbf{A}_-^{*(D)}$. Это делается за время $O(n)$ с использованием алгоритма [3].

Шаг 6. Вычислить $R^{-1}R_0^{-1}$ — преобразование вершин и граней многогранника $\mathbf{A}^*$, двойственного многограннику $\mathbf{A}^{*(D)}$, затем, нормализуя образы вершин, получить обобщенный многогранник $\mathbf{A}$, совпадающий с искомым пересечением исходных полупространств. Этот шаг также требует времени $O(n)$.

В заключение убеждаемся, что описанный алгоритм выполняется за время $O(n \log n)$, причем основной вклад вносят шаги 1, 2 и 3.

Чтобы исключить решения системы (2), отвечающие отрицательным точкам, можно перед применением алгоритма расширить систему (2) добавлением неравенства $x_4 \geqslant 0$. Оставшиеся положительные точки проецируются в решения системы (1), и, если имеются экваториальные точки, являющиеся крайними, их следует сохранить, чтобы облегчить описание множества решений.

4. ВЫРОЖДЕННЫЕ СЛУЧАИ

Один вырожденный случай может возникнуть при выполнении шага 3 основного алгоритма. Он имеет место, когда пересечение $\mathbf{A}_+^{(D)} \bigcap \mathbf{A}_-^{(D)}$ непусто, но не имеет внутренности, т. е. имеет нулевой объем. В этом случае алгоритм, описанный в работе [2], может быть использован для нахождения точки $\mathbf{q} = (q_1, q_2, q_3)$ в пересечении $\mathbf{A}_+ \bigcap \mathbf{A}_-^{(D)}$. Если $\mathbf{x} = (x_1, x_2, x_3, x_4)$ — какое-нибудь решение системы (3), то одновременно должны удовлетворяться неравенства

$$q_1x_1 + q_2x_2 + q_3x_3 + x_4 \geqslant 0,$$
$$-q_1x_1 - q_2x_2 - q_3x_3 - x_4 \geqslant 0.$$

Стало быть, множество решений системы (3) содержится в гиперплоскости $q_1x_1 + q_2x_2 + q_3x_3 + x_4 = 0$. Поэтому строится такое вращение R^*, что $(q_1, q_2, q_3, 1)R^* = (0, 0, 0, K^*)$, где K^* — положительная постоянная.

После применения R^* из системы неравенств $AR^*\mathbf{x}' \geqslant 0$ вытекает соотношение $x_4 = 0$, так что четвертый столбец матрицы AR^* оказывается избыточным. Поэтому система неравенств $AR^*x' \geqslant 0$ может быть заменена системой вида (1′), алгоритмическое решение которой уже описывалось. Таким образом, для получения решения системы (2) нужно только применить к результату вращение $(R^*)^{-1}$.

Другой вырожденный случай может возникнуть при выполнении шага 1 основного алгоритма. Он имеет место, когда не существует внутренней точки **u** множества решений системы (1′), т. е. системы $A_0\mathbf{x}' \geqslant 0$. В этом случае в процессе поиска требуемой внутренней точки обнаруживается либо (а) что в то время, как внутренних точек вовсе не существует, имеется точка $\mathbf{v} = (v_1, v_2, v_3)$, у которой или $v_3 = 1$ или $v_3 = -1$, и которая удовлетворяет (1′), либо (б) что конус $\mathbf{C}^3$ не пересекает ни одну из плоскостей $z = +1$ или $z = -1$.

В случае (а) точка **v** будет лежать на плоскости, проходящей через начало координат, которая содержит все решения (1′). Пусть эта плоскость имеет уравнение $m_1x + m_2y + m_3z = 0$. Выберем R^* таким, чтобы для некоторого положительного K^* выполнялось соотношение $(m_1, m_2, m_3, 0)R^* = (0, 0, 0, K^*)$, и поступим как в предыдущем случае. Уравнение искомой плоскости получается в результате применения алгоритма Шамоса и Хоя [5].

В случае (б) из неравенств (1′) вытекает, что $z = 0$, и поэтому третий столбец матрицы A является избыточным. Путем исключения третьего столбца, система неравенств $A\mathbf{x}' \geqslant 0$, как она дана в (3), может быть заменена системой вида (1′). Решение системы (1′) может быть получено прежним путем.

ЛИТЕРАТУРА

[1] Gass S. L. Linear programming. — McGraw-Hill, New York, 1969.
[2] Muller D. E., Preparata F. P. Finding the intersection of two convex polyhedra. — Theoret. Comput. Sci. 7 (2) (1978), 217—236. [Имеется перевод: Маллер Д., Препарата Ф. Нахождение пересечения двух выпуклых многогранников. — Киб. сб., вып. 20, н. с. — М.: Мир, 1983, 5—29.]
[3] Preparata F. P., Hong S. G. Convex hulls of finite sets in two and three dimensions. — Comm. ACM 20 (2) (1977), 87—93.
[4] Shamos M. I. Computational geometry. — Springer-Verlag, Berlin, 1980.
[5] Shamos M. I., Hoey D. Geometric intersection problems. — Proc. 17th Symp. on Found. Comput. Sci., Houston (October 1976), 208—215.

Булевы функции, чья монотонная сложность имеет величину порядка $n^2/\log n$[1]

И. Вегенер[2])

Строится последовательность систем монотонных булевых функций $h_n:\{0,1\}^n\to\{0,1\}^n$, таких, что монотонная сложность h_n имеет порядок роста $n^2/\log n$. В этом результате содержится наиболее высокая из известных нижняя оценка такого рода. Прежде имелись нижняя оценка вида $\Omega(n^{3/2})$ для булева матричного произведения, оценка вида $\Omega(n^{5/3})$ для булевых сумм, и принадлежащая автору оценка вида $\Omega(n^2/\log^2 n)$ для этой же системы h_n. Указанная новая нижняя оценка получена новыми методами, которые, вероятно, окажутся полезными и для других задач.

1. ВВЕДЕНИЕ

В данной работе улучшена нижняя оценка работы [13]. Мы напоминаем лишь наиболее важные определения и результаты этой работы, а за дальнейшими подробностями отсылаем читателя к оригиналу.

Мы снова рассматриваем систему функций f^m_{MN}, которую можно назвать «прямым произведением» m матриц, имеющих по M строк и N столбцов каждая. Эти матрицы обозначаются через

$$\left(x^1_{h_1 l}\right)_{\substack{1\leqslant h_1\leqslant M,\\ 1\leqslant l\leqslant N}},\ \ldots,\ \left(x^m_{h_m l}\right)_{\substack{1\leqslant h_m\leqslant M,\\ 1\leqslant l\leqslant N}}.$$

Считается, что во всех случаях $m\geqslant 2$ и $M\geqslant 2$. Содержательно, функции $y_{h_1\ldots h_m}(1\leqslant h_1,\ \ldots,\ h_m\leqslant M)$, составляющие рассматриваемую систему, определяются следующим образом. Рассматривая h_1-ю строку первой матрицы, h_2-ю строку второй матрицы, ..., и h_m-ю строку последней матрицы, проверяем, имеют ли все эти строки общую единичную компоненту, и полагаем

---

[1]) Wegener Ingo, Boolean functions whose monotone complexity is of size $n^2/\log n$. — Theoretical Computer Science, v. 21 (1982), 213—224.

[2]) Fakultät für Mathematika, Universität Bielefeld, Fed. Rep. Germany.

значение $y_{h_1 \dots h_m}$ равным 1 в том и только в том случае, когда это так. Таким образом

$$y_{h_1 \dots h_m} = \bigvee_{1 \leqslant l \leqslant N} x_{h_1 l} x^2_{h_2 l} \dots x^m_{h_m l},$$

где $1 \leqslant h_1, \dots, h_m \leqslant M$. (Знак конъюнкции «$\wedge$» опущен; в данной работе ab означает конъюнкцию $a \wedge b$, где a и b — булевы функции, переменные, или константы.) Совокупность M^m функций $y_{h_1 \dots h_m}$, определенных указанным образом, образует систему функций

$$f^m_{MN} : \{0, 1\}^{mMN} \to \{0, 1\}^{M^m}.$$

Отметим, что при $m = 2$ система f^2_{MN} вычисляет обычное булево матричное произведение матрицы $\left(x^1_{h_1 l}\right)_{\substack{1 \leqslant h_1 \leqslant M \\ 1 \leqslant l \leqslant N}}$ и матрицы, транспонированной к матрице $\left(x^2_{h_2 l}\right)_{\substack{1 \leqslant h_2 \leqslant M \\ 1 \leqslant l \leqslant N}}$. В связи с этим система f^m_{MN} и получила название «прямого произведения» m матриц.

В данной работе монотонная сложность системы f^m_{MN} определена с точностью до мультипликативной постоянной. Монотонная сложность системы булевых функций g — это наименьшее число функциональных элементов, достаточное для реализации системы g схемой из элементов $\wedge$, $\vee$.

В разделе 2 обсуждаются методы, использовавшиеся до сего времени для доказательства нижних оценок монотонной сложности систем булевых функций. Обсуждается также новый метод, предлагаемый в данной работе.

В разделе 3 методы работы [13] применяются для получения нашей новой нижней оценки. Определяется мера сложности, которая учитывает только элементы $\wedge$, причем в схемах допускается свободное вхождение некоторых функций. Для получения нижних оценок для монотонной сложности достаточно нахождения нижних оценок для указанной новой меры сложности. Поскольку допускается свободное вхождение некоторых функций и элементов $\vee$, становится возможным преобразовывать минимальную монотонную схему, реализующую систему f^m_{MN}, без увеличения ее сложности в смысле рассматриваемой новой меры. В результате этих преобразований получается схема, реализующая f^m_{MN}, о строении которой многое известно. На этом этапе применение старого метода подстановки констант с целью исключения из схемы ряда элементов позволяет получить лишь нижнюю оценку $(2/m)NM^m$ работы [13].

В разделе 4 мы доказываем наш основной результат, а именно повышаем нижнюю оценку до величины $(1/2)NM^m$, которая отличается от верхней оценки $3NM^m$ (см. [13]) лишь в 6 раз.

После этого без труда определяется система булевых функций, обладающая свойствами, объявленными в заголовке статьи.

Закончим данный раздел некоторыми определениями и обозначениями. Переменные обозначаются здесь через $x_1, \ldots, x_n$.

(1) Для системы монотонных функций g обозначим через $C(g)$ монотонную сложность этой системы, а через $C\hat{}(g)$ — наименьшее число элементов $\wedge$, достаточное для реализации системы g схемой из элементов $\wedge$, $\vee$.

(2) Функция t, представляющая собой конъюнкцию некоторых переменных, называется *конъюнкцией*, и может быть приведена к виду

$$t(x_1, \ldots, x_n) = x_{i_1} \ldots x_{i_m},$$

где $i_1, \ldots, i_m$ все различны и принадлежат $\{1, \ldots, n\}$. *Длиной* конъюнкции называется число входящих в нее различных переменных. Длина указанной выше конъюнкции t равна m.

(3) Конъюнкция t называется *импликантом* монотонной функции g, если $t \leqslant g$, то есть если для любого набора $a \in \{0, 1\}^n$ соотношение $t(a) = 1$ влечет $g(a) = 1$.

(4) Импликант t монотонной функции g называется *простым импликантом* этой функции, если никакая конъюнкция t', удовлетворяющая условиям $t' \neq t$ и $t \leqslant t'$, не является импликантом функции g. Множество всех простых импликантов функции g обозначается через $PI(g)$.

Отметим также следующий хорошо известный факт: для любой монотонной функции g

$$g(x_1, \ldots, x_n) \bigvee_{t \in PI(g)} t(x_1, \ldots, x_n).$$

Такое представление функции g называется *монотонной дизъюнктивной нормальной формой* функции g (кратко, МДНФ (g)). Это представление было использовано для определения функций $y_h \ldots {}_{hm}$,

Наконец, напомним следующие обозначения для функций $f, g\colon \mathbb{N} \to \mathbb{R}^+$:

$$f = O(g) \Leftrightarrow \exists c \in \mathbb{R}^+ \exists N_0 \in \mathbb{N} \forall n \geqslant N_0 : f(n) \leqslant cg(n);$$
$$f = \Omega(g) \Leftrightarrow \exists c \in \mathbb{R}^+ \exists N_0 \in \mathbb{N} \forall n \geqslant N_0 : f(n) \geqslant cg(n).$$

2. МЕТОДЫ ДОКАЗАТЕЛЬСТВА НИЖНИХ ОЦЕНОК МОНОТОННОЙ СЛОЖНОСТИ СИСТЕМ БУЛЕВЫХ ФУНКЦИЙ

Вначале напомним историю доказательства нижних оценок монотонной сложности систем булевых функций h_n: $\{0,1\}^n \to \{0,1\}^n$. Это делается для того, чтобы обсудить возможное значение нашего нового метода.

Прежде всего[1]) напомним, что монотонная схема может быть описана ориентированным ациклическим графом, вершины которого имеют полустепени захода либо 0 (переменные), либо 2 (элементы). Один из первых подходов состоял в том, чтобы забыв о булевой алгебре доказывать нижние оценки, используя теоретико-графовые свойства. Пиппенджер и Валиант [8] сумели доказать для некоторых систем монотонных функций наподобие системы функций сортировки и булевой свертки, что любая монотонная схема, реализующая такую систему, должна представляться сдвигающим графом (a shifting graph), и поэтому должна содержать $\Omega(n \log n)$ элементов по числу $\Omega(n \log n)$ вершин графа.

Другая идея состояла в том, чтобы осуществлять подстановку констант вместо некоторых переменных для достижения двух следующих целей.

(1) После осуществления подстановки констант можно было исключить из схемы возможно большее число элементов, оказавшихся избыточными (элемент избыточен, если, например, хотя бы на один из его входов поступает функция, равная константе).

(2) Система функций, реализуемая схемой, полученной путем указанного преобразования, совпала бы с некоторой системой h_i: $\{0,1\}^i \to \{0,1\}^i$ той же природы, что и исходная система h_n. Тогда можно действовать путем индукции, подсчитывая число всех элементов, подвергнувшихся исключению.

Этот метод был использован ван Воорисом [11] для доказательства нижней оценки вида $\Omega(n \log n)$ для системы функций сортировки. Ламанья [1] соединил теоретико-графовый подход с методом подстановки констант и исключения элементов. На этом пути он получил общую нижнюю оценку вида $\Omega(n \log n)$ для многих систем полуразделимых билинейных функций. В числе других к этому классу систем принадлежат булева свертка и система функций слияния.

Теперь коснемся подхода, связанного с изучением класса булевых сумм. Булевы суммы — это системы функций, которые мо-

[1]) Следовало бы отметить, что первая нелинейная нижняя оценка сложности реализации систем булевых функций схемами из функциональных элементов в монотонном базисе была опубликована Э. И. Нечипоруком в 1969 г. (см. [5]). — *Прим. перев.*

гут быть реализованы с использованием одних только элементов $\vee$. Идея доказательства нижних оценок здесь, казалось бы, ясна — сначала доказать, что минимальные схемы вовсе не содержат элементов $\wedge$, а затем уже нетрудно доказывать нижние оценки в предположении, что наличествуют одни лишь элементы $\vee$. К несчастью, первое из этих предположений в действительности не верно. Тарьян [10] доказал, что вообще говоря использование элементов $\wedge$ может упрощать реализацию булевых сумм. Тем не менее, если функции, образующие систему, «значительно отличаются» друг от друга, этот подход может быть использован и приводит к получению нижних оценок вида $\Omega(n^{3/2})$ (Нечипорук [5], а также Тарьян [10], Ламанья и Сэведж [2]). Следующий шаг состоял в том, чтобы сформулировать условия, при которых использование элементов $\wedge$ оказывает лишь небольшое влияние на сложность реализации, то есть может уменьшать число требуемых элементов лишь в ограниченное число раз. Вегенер [14] добился успеха на этом пути. В своей работе он использовал также метод предыдущей работы [13], а именно, он считал, что допускается свободное вхождение некоторых функций. Мы обсуждаем этот метод ниже. Обобщая эти результаты. Мельхорн [3] получил нижнюю оценку вида $\Omega(n^{5/3})$, которая была независимо получена Пиппенджером [7]. Подход, основанный на ограничении рассмотрений случаем наличия одних только элементов $\vee$, и доказательстве того, что такое ограничение не увеличивает сложности определенных систем функций, очевидно, подходит лишь для булевых сумм, и не может быть обобщен подобно методам, о которых говорилось ранее.

До того, как были получены упомянутые нижние оценки вида $\Omega(n^{5/3})$, был успешно развит следующий подход. Изучались системы, состоящие из булевых функций, чьи простые импликанты содержат не менее двух переменных, и вдобавок обладающих тем свойством, что простые импликанты каждой из этих функций не имеют общих переменных. Легко доказать, что реализация каждой из этих функций в отдельности требует по меньшей мере столько элементов $\wedge$, сколько имеется простых импликантов. Различные функции этой системы должны быть заданы таким образом, чтобы не существовало лучшей схемы для их совместной реализации, чем та, которая получается комбинированием минимальных схем, реализующих отдельные функции. Такой системой функций является булево матричное произведение, как было доказано Праттом [9], Патерсоном [6], и Мельхорном и Галилом [4]. Ими были получены оценки вида $\Omega(n^{3/2})$.

Они использовали метод подстановки констант и исключения элементов. Для того, чтобы применять этот метод, было не-

обходимо знать нечто о строении минимальных монотонных схем. Мельхорн и Галил сумели даже обобщить те идеи, которые оказались полезными при изучении схем, реализующих булево матричное произведение. Они доказали два правила замены следующего рода. Если функция s, реализуемая в монотонной схеме, реализующей систему g, обладает определенными свойствами, то можно исключить из схемы элемент, на выходе которого реализуется функция s, заменяя его подсхемой, реализующей некоторую иную функцию s'. Если функция s' окажется константой или переменной, то исходная схема не может быть минимальной. Для булева матричного произведения оказалось достаточным использовать правило замены в этом его специальном виде. Кроме того, оказалось возможным исключать все ранее обнаруженные элементы.

Вегенер [13] изучал системы функций f^m_{MN}, представляющие собой естественное обобщение булева матричного произведения. Здесь потребовалось применение одного из правил замены, в котором $s' = s \vee t$, где t — конъюнкция длины меньшей m. В этом случае возникающая при применении правила замены новая схема должна была бы содержать больше элементов, чем исходная. Таким образом, коль скоро происходило бы добавление к схеме новых элементов, и число новых элементов оказалось достаточно велико, то не было бы ничего удивительного в том, что впоследствии удалось бы доказать наличие большого числа элементов. Альтернатива состояла в том, чтобы отказаться от подсчета числа элементов $\vee$ и допустить свободное вхождение всевозможных конъюнкций, содержащих менее m переменных. Тогда при реализации s' стало возможным обойтись без новых элементов положительного веса.

Допущение свободного вхождения некоторых объектов могло бы, вообще говоря, привести к упрощению реализации системы f^m_{MN}. В нашей ситуации этого не происходит. Напомним, что мы хотели бы доказать существование одного элемента $\wedge$ для каждого из простых импликантов. Поскольку все простые импликанты имеют длину m, мы должны их в конечном счете вычислять, ибо свободное вхождение допускают лишь конъюнкции длины меньшей m. Основываясь на указанных допущениях, оказалось возможным доказать существование M^m различных элементов $\wedge$, на которых вычисляются M^m простых импликантов вида $x^1_{h_1 1} \ldots x^m_{h_m 1} (1 \leqslant h_1, \ldots, h_m \leqslant M)$. Затем был применен метод подстановки констант и исключения элементов. К несчастью, все M^m элементов исключить не удалось. С использованием принципа ящиков была установлена возможность исключения по меньшей мере $(2/m) M^m N$ элементов $\wedge$, что привело к нижней оценке вида $(2/m) M^m N$.

Здесь видны пределы метода подстановки констант и исключения элементов. Значимость этого метода убывает с ростом длины простых импликантов. Кроме того, в общем случае трудно надеяться, что один элемент $\wedge$ участвует в вычислении лишь одного простого импликанта. Удивительно, что булево матричное произведение обладает этим свойством, но, как можно наблюдать на некоторых примерах, это свойство не выполняется при $m > 2$.

В силу этих соображений мы предлагаем следующий метод. Мы не связываем с каждым элементом $\wedge$ соответствующего простого импликанта; мы связываем с каждым элементом $\wedge$ оценочную функцию. Эта функция обладает следующим свойством. Для каждого элемента $\wedge$ на множестве всех простых импликантов задается распределение некоторого значения, не превосходящего 1. Это означает, что если берется сумма по всем простым импликантам, и если суммируются значения, приписанные этим простым импликантам в распределении, связанном с данным элементом $\wedge$, то эта сумма не превосходит 1. Поэтому сумма всех значений, приписанных простым импликантам, не превосходит общего числа элементов $\wedge$. Затем доказывается, что каждому простому импликанту приписано значение, не меньшее некоторого заданного $\alpha > 0$. Стало быть, сумма всех значений, приписанных простым импликантам, не меньше числа всех простых импликантов, умноженного на α, и стало быть, общее число элементов $\wedge$ не меньше числа всех простых импликантов, умноженного на α. В данной работе $\alpha = 1/2$.

Мы не утверждаем, что мы — первые, кому пришла мысль о применении этого метода. Напротив, мы уверены, что почти всякий, кто занимается нижними оценками для какой-либо меры сложности булевых функций, имеет этот метод в виду. В данной работе впервые оказалось возможным применить этот метод с успехом.

3. ПРИМЕНЕНИЕ ИЗВЕСТНЫХ МЕТОДОВ ДЛЯ ОЦЕНКИ МОНОТОННОЙ СЛОЖНОСТИ СИСТЕМЫ f^m_{MN}

В работе [13] (лемма 4.2) применялось одно из двух правил замены, принадлежащих Мельхорну и Галилу [4]. Мы напоминаем этот результат в неформальном виде. Для этой цели вводятся следующие понятия и обозначения.

Простой импликант $x^1_{h_1 l} \ldots x^m_{h_m l}$ для краткости обозначается через $(h_1, \ldots, h_m, l)$. Конъюнкция t' называется *укорочением* конъюнкции t, если все переменные, входящие в t', входят также в t. Укорочение t' конъюнкции t называется *собственным* ее укорочением, если в t имеется переменная, не входящая в t'.

Отсюда должно быть ясно, какой смысл вкладывается в понятие *удлинения*. *Общей частью* нескольких конъюнкций $t_1, \ldots, t_r$ называется конъюнкция, состоящая из всех тех переменных, которые входят в каждую из конъюнкций $t_i (1 \leqslant i \leqslant r)$. Пустой конъюнкции соответствует константа 1.

Лемма (см. [13]). *Пусть S — монотонная схема, реализующая f^m_{MN} и пусть s — монотонная функция, реализуемая на выходе элемента G схемы S. Пусть для некоторых $l \in \{1, \ldots, N\}$ и $i_1, \ldots, i_m, j_1, \ldots, j_m \in \{1, \ldots, M\}$ некоторые укорочения конъюнкций $(i_1, \ldots, i_m, l)$ и $(j_1, \ldots, j_m, l)$ одновременно являются простыми импликантами функции s, и пусть s' — общая часть двух этих простых импликантов. Пусть, наконец, схема S' получена из схемы S путем реализации функции $s \vee s'$ на выходе нового элемента G' и присоединения к выходу G' некоторых из входов элементов, присоединявшихся к выходу G. Тогда схема S' также реализует f^m_{MN}.*

Мы будем рассматривать $*$-схемы, представляющие собой монотонные схемы со следующим дополнительным свойством $*$: *на входы схемы помимо переменных можно подавать всевозможные конъюнкции, содержащие менее m переменных; про такие конъюнкции говорят, что они имеют свободное вхождение.*

Через C^* (соответственно, $C\,\wedge$) обозначается мера сложности C (соответственно, $C^\wedge$), распространенная на $*$-схемы. Очевидно, что $C^* \leqslant C$ и $C^{*\wedge} \leqslant C^\wedge$.

Рассмотрим $*$-схему, реализующую f^m_{MN} и содержащую $C^{*\wedge}(f^m_{MN})$ элементов $\wedge$. Для всех элементов этой $*$-схемы в соответствии с их топологическим порядком осуществляется следующая процедура. (Топологический порядок — это такой порядок, при котором не существует цепи из элементов, ведущей от элемента G к какому-либо из предшествующих ему элементов G'.) Простые импликанты вида $(h_1, \ldots, h_m, l)$ мы будем называть простыми импликантами типа l.

Для каждого элемента G осуществляются следующие преобразования. Если функция s, реализуемая на выходе G, имеет по меньшей мере два различных простых импликанта, которые являются укорочениями простых импликантов типа l, положим s_l равным общей части всех этих простых импликантов, а в противном случае положим $s_l = 0$. На выходе нового элемента G' реализуем функцию $s' = s \vee s_1 \vee \ldots \vee s_N$, и присоединим к выходу G' входы всех тех элементов, которые в исходной схеме присоединялись к выходу G.

Функции $s_1, \ldots, s_N$ имеют свободное вхождение, поскольку каждая из них есть либо 0, либо общая часть по меньшей мере

двух различных укорочений простых импликантов типа l, и в силу этого содержит менее m переменных. Поэтому сложность новой $*$-схемы по отношению к мере $C^{*\wedge}$ совпадает со сложностью исходной $*$-схемы. Применив нужное число раз сформулированную выше лемму, заключаем, что новая $*$-схема также реализует f^m_{MN}.

Если для некоторого элемента $s_l = 0$, то те простые импликанты, которые состоят только из переменных вида x^b_{al}, не затрагиваются описанными преобразованиями, причем наличествует не более одного укорочения простого импликанта типа l. Если же $s_l \neq 0$, все укорочения простых импликантов типа l заменяются на s_l, а прочие простые импликанты, состоящие только из переменных вида x^b_{al}, либо остаются нетронутыми, либо также заменяются на s_l. Поэтому можно сделать следующие выводы, касающиеся новой $*$-схемы, полученной после осуществления описанных преобразований над всеми элементами исходной схемы в соответствии с их топологическим порядком. Функции, поступающие на входы элементов $\wedge$, могут иметь не более одного простого импликанта, являющегося укорочением простого импликанта типа l. В силу элементарных свойств элементов $\wedge$ отсюда следует, что и функции, реализуемые на выходах элементов $\wedge$, могут иметь не более одного простого импликанта, являющегося укорочением простого импликанта типа l. Эти выводы играют в данной работе роль основной леммы работы [13].

4. МОНОТОННАЯ СЛОЖНОСТЬ СИСТЕМЫ f^m_{MN}

По теореме 3.1 (I) из [13]

$$C(f^m_{MN}) \leqslant N \sum_{2 \leqslant i \leqslant m} M^i + (N-1) M^m.$$

Упростим эту верхнюю оценку.

Теорема 1. *Если* $M \geqslant 2$, *то*

$$C(f^m_{MN}) \leqslant 3NM^m.$$

В оставшейся части данного раздела будет доказана следующая

Теорема 2. *Если* $m \geqslant 2$, *то*

$$C(f^m_{MN}) \geqslant C^{\wedge}(f^m_{MN}) \geqslant C^{*\wedge}(f^m_{MN}) > (1/2) NM^m.$$

Первые два из указанных неравенств очевидны. Для доказательства последнего из этих неравенств рассмотрим монотонную $*$-схему S, реализующую систему f^m_{MN}, и содержащую $C^{*\wedge}(f^m_{MN})$ элементов $\wedge$. При этом считается, что схема преобра-

зована в соответствии с процедурой раздела 3, и что на входы всех ее элементов поступают функции, отличные от констант.

Определим теперь оценочную функцию, о которой говорилось в разделе 2. Пусть G — некоторый элемент $\wedge$ схемы S, s — функция, реализуемая на выходе элемента G, и пусть s' и s'' — функции, поступающие соответственно на левый и правый входы элемента G; таким образом $s = s's''$. В силу результатов раздела 3, функции s, s' и s'' имеют для каждого $l \in \{1, \ldots, N\}$ не более одного простого импликанта, являющегося укорочением простого импликанта типа l. Пусть индексы $i_1, \ldots, i_q$, $1 \leqslant i_1, \ldots, i_q \leqslant N$, выбраны так, что выполняются следующие условия:

(1) $i_1, \ldots, i_q$ различны;

(2) простой импликант s_r функции s принадлежит к типу i_r $(1 \leqslant r \leqslant q)$;

(3) простой импликант s'_r функции s' является собственным укорочением $s_r (1 \leqslant r \leqslant q)$;

(4) не существует $i_{q+1} \in \{1, \ldots, N\}$ такого, чтобы условия (1)—(3) выполнялись для $i_1, \ldots, i_q, i_{q+1}$.

Тогда оценочная функция v'_G определяется следующим образом. Функция v'_G приписывает каждому из простых импликантов $s_1, \ldots, s_q$ значение $(2q)^{-1}$, а каждому из остальных простых импликантов функций, составляющих f^m_{MN} — значение 0. Отсюда

$$\sum_{1 \leqslant h_1, \ldots, h_m \leqslant M} \sum_{1 \leqslant l \leqslant N} v'_G(h_1, \ldots, h_m, l) \leqslant 1/2.$$

Точнее, эта сумма равна либо 1/2, либо 0. Аналогичным образом, заменяя функцию s' функцией s'', определим оценочную функцию v''_G. Наконец, положим $v_G = v'_G + v''_G$. Отсюда

$$\sum_{1 \leqslant h_1, \ldots, h_m \leqslant M} \sum_{1 \leqslant l \leqslant N} v_G(h_1, \ldots, h_m, l) \leqslant 1/2 + 1/2 = 1,$$

и

$$\sum_{G \in S\wedge} \sum_{1 \leqslant h_1, \ldots, h_m \leqslant M} \sum_{1 \leqslant l \leqslant N} v_G(h_1, \ldots, h_m, l) \leqslant |S\wedge| = C^*\wedge(f^m_{MN}),$$

где $S\wedge$ — множество всех элементов $\wedge$ схемы S.

Положим $v(h_1, \ldots, h_m, l) = \sum_{G \in S\wedge} v_G(h_1, \ldots, h_m, l)$.

Утверждение. *При любых* $h_1, \ldots, h_m \in \{1, \ldots, M\}$ и $l \in \{1, \ldots, N\}$ $v(h_1, \ldots, h_m, l) > 1/2$.

Доказав это утверждение, нетрудно вывести из него утверждение теоремы. Действительно, тогда

$$\sum_{1\leqslant h_1, \ldots, h_m\leqslant M} \sum_{1\leqslant l\leqslant N} \sum_{G\in S\wedge} v_G(h_1, \ldots, h_m, l) > (1/2)\, NM^m,$$

и, соединяя это неравенство с полученным выше, имеем $C^*{}^{\wedge}(f^m_{MN}) > (1/2)\, NM^m$.

Чтобы доказать сформулированное утверждение, введем в рассмотрение подсхему $S_{h_1 \ldots h_m l}$ схемы S, состоящую из следующих элементов и входов.

(1) Каждый элемент G, связанный с выходом $y_{h_1 \ldots h_m}$ цепью из элементов, любой элемент которой (включая и сам элемент G) имеет конъюнкцию $(h_1, \ldots, h_m, l)$ простым импликантом реализуемой на его выходе функции, входит в подсхему $S_{h_1 \ldots h_m l}$.

(2) Если вход элемента G, вошедшего согласно (1) в подсхему $S_{h_1 \ldots h_m l}$, присоединен к выходу элемента, не вошедшего в $S_{h_1 \ldots h_m l}$, то этот вход является входом подсхемы $S_{h_1 \ldots h_m l}$.

В соответствии с этим построением возникает связная подсхема $S_{h_1 \ldots h_m l}$, имеющая один-единственный выход, на котором реализуется функция $y_{h_1 \ldots h_m}$. По определению ни одна из функций, поступающих на входы подсхемы $S_{h_1 \ldots h_m l}$ не имеет конъюнкцию $(h_1, \ldots, h_m, l)$ своим простым импликантом, в то время как функции, реализуемые на выходах всех элементов подсхемы $S_{h_1 \ldots h_m l}$ имеют $(h_1, \ldots, h_m, l)$ своим простым импликантом.

Опишем два свойства входов подсхемы $S_{h_1 \ldots h_m l}$.

P1. *Если оба входа элемента G, входящего в $S_{h_1 \ldots h_m l}$, являются входами $S_{h_1 \ldots h_m l}$, то G — элемент* $\wedge$.

Действительно, в противном случае G оказался бы элементом $\vee$, реализующим функцию, имеющую $(h_1, \ldots, h_m, l)$ своим простым импликантом, в то время, как поступающие на его входы функции не имеют таких простых импликантов. Получается противоречие, ибо на элементах $\vee$ новые простые импликанты не возникают.

P2. *Если вход подсхемы $S_{h_1 \ldots h_m l}$ является входом элемента* $\wedge$, *то поступающая на этот вход функция имеет своим простым импликантом некоторое собственное укорочение конъюнкции* $(h_1, \ldots, h_m, l)$.

Действительно, поскольку $(h_1, \ldots, h_m, l)$ является простым импликантом функции, реализуемой на выходе рассматриваемого элемента $\wedge$, обе функции, поступающие на его входы, должны иметь своими простыми импликантами некоторые уко-

рочения конъюнкции $(h_1, \ldots, h_m, l)$ — это одно из элементарных свойств элементов $\wedge$. По определению, сама конъюнкция $(h_1, \ldots, h_m, l)$ не является простым импликантом функций, поступающих на входы подсхемы $S_{h_1 \ldots h_m l}$, стало быть тот простой импликант, о котором говорится в P2, должен быть собственным укорочением $(h_1, \ldots, h_m, l)$.

Пусть теперь $s_1, \ldots, s_D$ — функции, поступающие на те входы подсхемы $S_{h_1 \ldots h_m l}$, которые являются входами элементов $\wedge$. В силу P1 множество этих функций непусто, а в силу P2 каждая из функций $s_1, \ldots, s_D$ имеет своим простым импликантом некоторое собственное укорочение конъюнкции $(h_1, \ldots, h_m, l)$. Для каждой из функций s_i $(1 \leqslant i \leqslant D)$ выберем какой-нибудь элемент G_i — элемент $\wedge$ подсхемы $S_{h_1 \ldots h_m l}$, на вход которого поступает s_i. Положим $v^*_{G_i}$ равным v'_{G_i} (соответственно v''_{G_i}), если s_i поступает на левый (соответственно, на правый) вход G_i. По определению оценочной функции $v^*_{G_i}(h_1, \ldots, h_m, l) > 0$. Для доказательства требуемого утверждения достаточно доказать, что

$$\sum_{1 \leqslant i \leqslant D} v^*_{G_i}(h_1, \ldots, h_m, l) > 1/2,$$

ибо в этой сумме каждое значение учитывается не более одного раза. Указанное неравенство доказывается от противного. Итак, допустим, что

$$\sum_{1 \leqslant i \leqslant D} v^*_{G_i}(h_1, \ldots, h_m, l) \leqslant 1/2.$$

Используя обозначение $b_i = 2v^*_{G_i}(h_1, \ldots, h_m, l)$, запишем это допущение в виде

$$b_1 + \ldots + b_D \leqslant 1.$$

Не ограничивая общности можно считать, что $b_1 \geqslant \ldots \geqslant b_D$. Выберем теперь простые импликанты w_i функций s_i так, чтобы выполнялись следующие свойства:

(1) некоторое удлинение w^*_i конъюнкции w_i является простым импликантом функции, реализуемой на выходе элемента G_i;

(2) $v^*_{G_i}(w^*_i) > 0$;

(3) тип конъюнкции w^*_i отличен от типов конъюнкций $w^*_1, \ldots, w^*_{i-1}$.

Конъюнкция w_1 всегда может быть выбрана такой, что $w^*_1 = (h_1, \ldots, h_m, l)$, ибо для любого i из промежутка $1 \leqslant i \leqslant D$ $v^*_{G_i}(h_1, \ldots, h_m, l) > 0$. Если бы выбор какой-либо конъюнкции w_i

с описанными выше свойствами был невозможен, соотношение $v^*_{G_i}(t) > 0$ было бы выполнено не более, чем для $i-1$ простых импликантов. Это означало бы, что $b_i \geqslant (i-1)^{-1}$, и тогда в силу соотношений $b_1 \geqslant \ldots \geqslant b_D$ оказалось бы, что $b_1 + \ldots + b_D \geqslant b_1 + \ldots b_i \geqslant ib_i \geqslant i/(i-1) > 1$. Стало быть, описанный выше выбор конъюнкций $w_1, \ldots, w_D$ возможен.

Поскольку w_i является простым импликантом функции s_i, выполняются соотношения $w_i \leqslant s_i$ и $w_1 \ldots w_D \leqslant s_1 \ldots s_D$. Далее, докажем, что выполняется соотношение $s_1 \ldots s_D \leqslant y_{h_1 \ldots h_m}$.

Действительно, в противном случае нашелся бы набор значений переменных, на котором все функции s_i обращаются в 1, в то время как $y_{h_1 \ldots h_m}$ обращается в 0. Тогда оказалось бы, что подсхема $S_{h_1 \ldots h_m l}$ — монотонная схема, обладающая следующими свойствами: на все входы этой подсхемы, являющиеся входами элементов $\wedge$, поступает значение 1, причем все элементы $S_{h_1 \ldots h_m l}$ у которых оба входа являются входами $S_{h_1 \ldots h_m l}$ являются элементами $\wedge$, и в то же время на выходе подсхемы реализуется 0. Эти свойства противоречивы. Индукцией по топологическому порядку элементов подсхемы $S_{h_1 \ldots h_m l}$ нетрудно показать, что на выходе каждого элемента реализуется значение 1 — это и дает требуемое противоречие. Действительно, в силу свойства P1 первый по порядку («самый верхний») элемент является элементом $\wedge$, причем на оба его входа поступает 1 — стало быть, на его выходе реализуется 1. Рассматривая произвольный элемент $\wedge$ подсхемы $S_{h_1 \ldots h_m l}$, убеждаемся, что на каждый из его входов поступает 1 — либо по предположению индукции (если этот вход не является входом подсхемы), либо в силу того, что значения всех s_i равны 1 (если этот вход является входом подсхемы). Рассматривая произвольный элемент $\vee$ подсхемы $S_{h_1 \ldots h_m l}$, убеждаемся, что в силу свойства P1 один из входов этого элемента не является входом подсхемы; стало быть, по предположению индукции на этот вход поступает 1 и выход рассматриваемого элемента $\vee$ реализует 1. Требуемое соотношение доказано, а вместе с ним и соотношение $w_1 \ldots w_D \leqslant y_{h_1 \ldots h_m}$.

Пусть l_i обозначает тип конъюнкции w_i. Конъюнкция w_i состоит только из переменных типа l_i и по определению является собственным укорочением конъюнкции w_i^*. Поэтому конъюнкция w_i состоит самое большее из $m-1$ переменных типа l_i. Далее, по определению конъюнкций $w_1, \ldots, w_D$ типы $l_1, \ldots, l_D$ различны. Если все те переменные, которые входят в $w_1, \ldots, w_D$ положить равными 1, а все остальные — равными 0, то не более $m-1$ переменных каждого типа окажутся равными 1. Но тогда

функция $y_{h_1 \ldots h_m}$, представляющая собой дизъюнкцию простых импликантов, содержащих по m переменных одного и того же типа, окажется равной 0.

Таким образом, опровергнуто соотношение

$$w_1 \ldots w_D \leqslant y_{h_1 \ldots h_m}.$$

Соединяя эти результаты, заключаем, что неравенство

$$\sum_{1 \leqslant i \leqslant D} v^*_{G_i}(h_1, \ldots, h_m, l) > 1/2$$

доказано, а вместе с ним и теорема 2.

В работе [13] было показано, что

$$C^*\wedge(f^m_{MN}) \geqslant (2/m)\, NM^m.$$

в то время как в данной работе установлено, что

$$C^*\wedge(f^m_{MN}) > (1/2)\, NM^m.$$

Наша новая оценка лучше старой при $m \geqslant 4$, в то время как при $m = 2$ или $m = 3$ оценка работы [13] несколько лучше новой. В работе [13] уже было отмечено, что при $m = 2$ старая оценка даже оптимальна. В следующем разделе мы увидим, что нижние оценки получаются особенно высокими, если m стремится к бесконечности с ростом числа переменных. Поэтому с помощью нашей новой оценки мы можем получать более высокие нижние оценки, чем с помощью старой.

Нижняя и верхняя оценки теорем 1 и 2 различаются всего лишь в 6 раз. Нетрудно видеть, что $C^*\wedge(f^m_{MN}) \leqslant NM^m$.

Стало быть, наша нижняя оценка могла бы быть лучше всего лишь в 2 раза. Впоследствии мы обсудим, почему мы не смогли избавиться от этого разрыва, но прежде этого применим наши результаты.

5. БУЛЕВЫ ФУНКЦИИ, ЧЬЯ МОНОТОННАЯ СЛОЖНОСТЬ ИМЕЕТ ВЕЛИЧИНУ ПОРЯДКА $n^2/\log n$

Введем следующие обозначения. Если определена система функций g: $\{0,1\}^n \to \{0,1\}^m$, то тем же символом g мы будем обозначать систему g': $\{0,1\}^{n+n'} \to \{0,1\}^{m+m'}$, в которой для каждого i из промежутка $1 \leqslant i \leqslant m$ $g'_i(x_1, \ldots, x_{n+n'}) = g_i, \ldots, x_n)$, а для каждого i из промежутка $m < i \leqslant m + m'$ $g'_i(x_1, \ldots, x_{n+n'}) = 0$. Очевидно, для всех мер сложности сложность системы g совпадает со сложностью системы g'.

Для $n \geqslant 4$ положим $m(n) = \lfloor \log_2 n \rfloor$, $M(n) = 2$, и $N(n) = \lfloor n/(2 \log_2 n) \rfloor$. Поскольку $m(n) M(n) N(n) \leqslant n$ и $M(n)^{m(n)} \leqslant$

$\leqslant n$, можно определить систему h_n: $\{0,1\}^n \to \{0,1\}^n$ соотношением $h_n = f^{m(n)}_{M(n)N(n)}$.

Теорема 3. *Система монотонных функций h_n задана явным конкретным образом, и* $\Omega(n^2/\log n) = C(h_n) = O(n^2/\log n)$.

Эта теорема есть прямое следствие наших оценок из раздела 4.

Элементарные вычисления показывают, что монотонная сложность всех систем $h'_n : \{0,1\}^n \to \{0,1\}^*$, эквивалентных какой-либо из систем f^m_{MN}, ограничена сверху величиной $O(n^2/\log n)$. Поэтому из наших результатов мы не можем извлечь нижних оценок, по порядку величины лучших, чем оценка теоремы 3. Мы можем лишь улучшить эту оценку в небольшое число раз, выбирая $m'(n) = \lfloor \log_3 n \rfloor$, $M'(n) = 3$ и $N'(n) = \lfloor n/(3\log_3 n) \rfloor$ для $n \geqslant 9$.

6. ЗАКЛЮЧИТЕЛЬНЫЕ ЗАМЕЧАНИЯ

(1) В работе [12] мы обобщили результаты [13] для отыскания монотонной сложности разделимых полилинейных форм. Теперь снова легко распространить результаты данной работы на класс разделимых монотонных полилинейных форм, устанавливая нижнюю оценку, равную половине числа простых импликантов рассматриваемых функций. С помощью этого распространения наших результатов мы не можем получать более высоких нижних оценок, чем оценки данной работы.

(2) Предположение $C^{*\wedge}(f^m_{MN}) = NM^M$ по-прежнему остается открытым. Доказательство этого предположения представляется не таким уж важным, но, возможно, оно поможет лучше проникнуть в суть наших методов. Для того, чтобы доказать это предположение, следует задаться вопросом, по какой причине мы не смогли избавиться от множителя 1/2 в нашей нижней оценке.

Пытаясь доказать желаемую нижнюю оценку, автор заметил, что нельзя получить лучших нижних оценок, используя ту же самую оценочную функцию. Им были построены монотонные схемы, реализующие систему f^m_{MN}, для которых $v(h_1, \ldots, h_m, l) = (i+1)/(2i)$, что для любого положительного ε меньше $(1/2)+\varepsilon$, если i достаточно велико.

В одной из этих схем почти всем элементам, являющимся элементами $\wedge$, соответствует значение 1/2, так как $v'(G) = 0$, а $v''(G) = 1/2$ (или наоборот). В другой из этих схем $v(G) = 0$ приблизительно для половины всех элементов $\wedge$, и $v(G) = 1$ для другой половины.

Имеются также примеры $*$-схем для f^m_{MN}, содержащих NM^m элементов $\wedge$, в которых некоторое число элементов участвует в реализации сразу нескольких выходных функций.

(3) Мы уверены, что метод, использующий оценочные функции, окажется важным орудием доказательства нижних оценок не только для монотонной сложности булевых функций, но также и для немонотонной сложности булевых функций.

ЛИТЕРАТУРА

[1] Lamagna E. A. The complexity of monotone networks for certain bilinear form, routing problems, sorting and merging. — IEEE Trans. Comput. 28 (1979), 773—782.
[2] Lamagna E. A., Savage J. E. Combinational complexity of some monotone functions. — Proc. 15th SWAT Conference, New Orleans (1974), 140—144.
[3] Mehlhorn K. Some remarks on Boolean sums. — Acta Informat. 12 (1979), 371—375. [Имеется перевод: Мельхорн К. Некоторые замечания, касающиеся булевых сумм. — Киб. сб., вып. 18, н. с. — М.: Мир, 1981, 39—45.]
[4] Melhorn K., Galil Z. Monotone switching circuits and Boolean matrix product. — Computing 16 (1976), 99—111.
[5] Нечипорук Э. И., Об одной булевской матрице. — В сб. «Проблемы кибернетики», вып. 21. — М.: Наука, 1969.
[6] Paterson M. S. Complexity of monotone networks for Boolean matrix product. — Theoret. Comput. Sci. I (1975), 13—20. [Имеется перевод: Патерсон М. С. Сложность монотонных схем для булева умножения матриц. — Киб. сб., вып. 15, н. с. — М.: Мир, 1978, 28—37.]
[7] Pippenger N On another Boolean matrix. — IBM Research Report 6914 (1977).
[8] Pippenger N., Valiant L G. Shifting graphs and their applications. — J. ACM 23 (1976), 423—432.
[9] Pratt V. R. The power of negative thinking in multiplying Boolean matrices. — SIAM J. Comput. 4 (1975). 326—330.
[10] Tarjan R. E. Complexity of monotone networks for computing conjuctions. — Ann. Discrete Math. 2 (1978), 121—133.
[11] Voorhis D. C., van. An improved lower bound for sorting networks. — IEEE Trans. Comput. 21 (1972), 612—613.
[12] Wegener I. Boolesche Funktionen, deren monotone Komplexität fast quadratisch ist, Dissertation, Universität Bielefeld (1978).
[13] Wegener I. Switching functions whose monotone complexity is nearly quadratic. — Theoret. Comput. Sci. 9 (1979), 83—97. [Имеется перевод: Вегенер И., Булевы функции, чья монотонная сложность почти квадратична, Киб. сб., вып. 18, н. с. — М.: Мир, 1981, 55—74.]
[14] Wegener I. A new lower bound on monotone network complexity of Boolean sums. — Acta Informat. 13 (1980), 109—114.

Нумераторы спектра для некоторых кодов над целочисленными алфавитами, исправляющих аддитивные ошибки[1)]

Ф. Дельсарт, Ф. Пире [2)]

В статье изучаются блоковые коды с целочисленными координатными символами, предназначенные для исправления аддитивных ошибок некоторых типов. Коды определяются проверочным уравнением над конечной абелевой группой и, таким образом, являются обобщением кодов Варшамова и Констэнтина — Рао. Основные результаты статьи касаются корректирующей способности, включая метод декодирования обобщенных кодов Варшамова, и главным образом связаны с перечислительными задачами. Получена общая формула для нумератора спектра; она применена к кодам Варшамова и Констэнтина — Рао.

1. ВВЕДЕНИЕ

Целью этой статьи является исследование некоторых классов блоковых кодов над *целочисленными алфавитами*, допускающих исправление определенных *аддитивных ошибок*. Типичный код C состоит из n-последовательностей $\mathbf{v}$ с координатными символами v_i из данного алфавита, удовлетворяющих *проверочному уравнению* вида $v_1h_1 + v_2h_2 + \ldots + v_nh_n = h_0$, где h_i — фиксированные элементы конечной абелевой группы. Пусть s — период этой группы. В частном случае, когда алфавит есть отрезок $[0, s-1]$ целых чисел, код C является *смежным классом* s-ичного *группового кода*. С другой стороны, оказывается, что код C общего вида (определенный над произвольным целочисленным алфавитом) может быть непосредственно получен из только что упомянутого специального кода. В частности, когда алфавит конечный, мощность кода C непосредственно вычисляется из *нумератора спектра* специального кода.

Нас будут главным образом интересовать те коды C, которые исправляют все аддитивные ошибки $\mathbf{e}$ с координатами e_i из $[0, s-1]$, такие что их *амплитуда* $e_1 + \ldots + e_n$ не превышает заданного целого числа t. В настоящей статье коды, обла-

[1)] Delsarte Ph., Piret Ph. Spectral Enumerators for Certain Additive-Error-Correcting Codes over Integer Alphabetes. Information and Control, 48, 3, 193—210, 1981.

[2)] Philips Research Laboratory, Brussels, Belgium.

дающие указанной корректирующей способностью, называются t-кодами. Впервые построенные Варшамовым и Тененгольцем [21] 1-коды были обобщены Стенли и Йодером [18], которые рассмотрели весь класс 1-кодов максимальной длины над отрезком $[0, c-1]$ целых чисел. Тот же самый класс кодов был недавно вновь открыт и детально изучен Констэнтином и Рао [3] в двоичном случае ($c=2$). Мощность двоичных кодов Варшамова — Тененгольца была найдена Гинзбургом [7], а кодов Констэнтина — Рао — Мак-Элисом и Родемичем [15]. Распределение весов кодов Варшамова — Тененгольца было вычислено Мазуром [13] в двоичном случае и Стенли и Йодером [18] в общем c-ичном случае. Мощность и распределение весов c-ичных кодов Стэнли — Йодера (которые сводятся к кодам Констэнтина — Рао в двоичном случае) были недавно получены в работе [9].

Варшамов [20] открыл замечательный класс c-ичных t-кодов с $t \geqslant 1$, которые тесно связаны с классическими кодами БЧХ над простыми полями. Коды Варшамова были далее исследованы Мак-Элисом [14], который привел простое доказательство их корректирующей способности и указал полезную среднюю границу их мощности. Мазур [13] показал, что мощность любого из данных кодов довольно близка к этой средней величине. Интересный метод декодирования двоичных кодов Варшамова был предложен Налбандяном [16].

В настоящей статье детально исследуются некоторые обобщения упомянутых выше кодов как с точки зрения исправления *ошибок*, так и *перечисления*. Статья имеет следующую структуру.

В разд. 2 сначала дается краткое алгебраическое описание типа ошибок, которые должны исправляться рассматриваемыми кодами. Затем дано определение кодов через проверочную $(n+1)$-последовательность над абелевой группой. Простая теорема устанавливает связь между этими двумя понятиями. Вслед за этим описано обобщение кодов Констэнтина — Рао и Варшамова. В частности, приведена теорема о корректирующей способности кодов Варшамова. Доказательство теоремы является развитием того доказательства, которое использовал Налбандян [16]; оно включает алгоритм декодирования, аналогичный алгоритму, разработанному для кодов БЧХ. Интересным следствием этой теоремы является тот факт, что p-ичные коды БЧХ способны исправлять больше асимметричных ошибок, чем это гарантируется конструктивным расстоянием Хэмминга. В конце раздела упомянуто построение равновесных кодов из двоичных кодов, исправляющих асимметрические ошибки. Идея такого построения впервые была использована Боузом и Рао [2] для получения из кодов Констэнтина — Рао некоторых кодов с хоро-

шими корректирующими и обнаруживающими свойствами. Ту же идею неявно использовали Грэхэм и Слоэн [8], которые получили интересные двоичные равновесные коды из класса кодов, тесно связанного с обобщенными кодами Варшамова.

В разд. 3 рассматривается перечисление. После предварительного результата, который обобщает приведенную Мак-Элисом [14] границу усреднения, вводится понятие s-ичного нумератора спектра кода C. Наше определение непосредственно связано с определением «полного нумератора весов» из классической теории кодирования, см. [12]. В важном случае двоичного алфавита $\{0, 1\}$ нумератор спектра сводится к нумератору веса Хэмминга. Основным результатом, который потенциально имеет большую область приложений, является явная формула для нумератора спектра кода C, выраженная через проверочную $(n+1)$-последовательность, задающую код C. Идеи, лежащие в основе доказательства, возвращают нас к работе Мак-Вильямс [11]. В действительности наша формула может рассматриваться как обобщение тождества Мак-Вильямс для нумератора веса двойственных кодов. Остальная часть разд. 3 посвящена приложению общего результата к нумераторам весов кодов Константина — Рао, с одной стороны, и некоторых обобщенных кодов Варшамова — с другой. Для первого класса кодов результатом является выражение для нумератора весов через элементарные теоретико-числовые и теоретико-групповые функции; эквивалентное выражение было найдено и детально проанализировано в работе [9]. Случай обобщенных кодов Варшамова гораздо труднее; здесь приведены только результаты о классе двоичных 2-кодов (полной длины), для которого с помощью теории квадратичных вычетов получены явные формулы. Весь этот раздел сопровождается разбором нескольких примеров.

2. КОРРЕКТИРУЮЩАЯ СПОСОБНОСТЬ

Прежде чем определять наши коды, кратко объясним, к какому классу каналов передачи они приспособлены. Передаваемые символы являются элементами данного подмножества A (называемого *алфавитом*) в кольце целых чисел $\mathbb{Z}$, $|A| \geqslant 2$, тогда как символы ошибок принадлежат подмножеству $S = \{0, 1, \ldots, s-1\}$ в $\mathbb{Z}$, где s — некоторое фиксированное число, $s \geqslant 2$. Для данного элемента e из S обозначим через A_e подмножество алфавита A, состоящее из тех символов, на которые может воздействовать ошибка e. В случае непустого A_e действие ошибки e описывается взаимооднозначным отображением f_e множества A_e в $\mathbb{Z}$, удовлетворяющим аддитивному условию $f_e(a) \equiv a + e \pmod{s}$ для всех $a \in A_e$. Укажем теперь три важных частных случая, которые служат оправданием на-

шего исследования: (1) строго аддитивный канал над алфавитом из целых чисел с $A_e = A = \mathbb{Z}$ и $f_e(a) = a + e$; (2) классический s-ичный канал с $A_e = A = S$ и $f_e(a)$, равным вычету $a + e$ по модулю s; (3) асимметрический c-ичный канал с $A = \{0, 1, \ldots, c-1\}$, $A_e = \{0, 1, \ldots, c-1-e\}$ и $f_e(a) = a + e$.

Пусть $(G, +, 0)$ — *конечная абелева группа* или, что эквивалентно, конечный модуль над кольцом $\mathbb{Z}$. Для данного целого числа $n \geqslant 2$ рассмотрим произвольную n-последовательность $\mathbf{h} = (h_1, h_2, \ldots, h_n)$ с элементами $h_i \in G$. Не ограничивая общности, будем предполагать, что элементы h_i порождают G. Определим код C длины n над алфавитом A как множество n-последовательностей, которые при действии на $\mathbf{h}$ в качестве коэффициентов линейной комбинации дают фиксированный элемент $h_0 \in G$. Формально это можно записать в виде

$$C = \{\mathbf{v} \in A^n : \mathbf{v}\mathbf{h}^T = h_0\}, \tag{1}$$

где $\mathbf{v}\mathbf{h}^T$ при $\mathbf{v} = (v_1, v_2, \ldots, v_n)$ означает «скалярное произведение» $v_1h_1 + v_2h_2 + \ldots + v_nh_n$. Ниже $(h_0, \mathbf{h})$ будем называть *проверочной* $(n+1)$-последовательностью кода C. Важным параметром окажется *период* группы G, т. е. наименьшее положительное целое число s, такое что $sh_i = 0$ для $i = 1, 2, \ldots, n$.

Обратим внимание читателя на два замечания об *изоморфизмах*, которые приводят к упрощению полного анализа семейства кодов (1). Рассмотрим автоморфизм σ группы G, переставляющий элементы $h_1, h_2, \ldots, h_n$. Тогда замена h_0 на $\sigma(h_0)$ дает код, который перестановочно эквивалентен исходному коду C. С другой стороны, предположим, что алфавит A симметричен относительно некоторого целого числа r в том смысле, что $r - a$ принадлежит A при всех $a \in A$. Тогда замена h_0 на $r(h_1 + \ldots \ldots h_n) - h_0$ переводит код C в его r-дополнение (где каждый символ v_i заменяется на $r - v_i$). Примером служит двоичный алфавит $A = \{0, 1\}$ при $r = 1$.

Когда это необходимо, будем использовать обозначение C_A для кода (1) над данным алфавитом A. Положим $S = \{0, 1, \ldots \ldots, s-1\}$, где s — период группы G. Код C_S естественно связан с группой G и играет особую роль в теории. На самом деле оказывается, что для данной проверочной $(n+1)$-последовательности $(h_0, \mathbf{h})$ любой код C_A может быть непосредственно получен из кода C_S: а именно, код C_A состоит из векторов, сравнимых по модулю s с векторами кода C_S. Это можно записать формулой

$$C_A = \bigcup_{u \in C_S} \{\mathbf{v} \in A^n : v_i \equiv u_i \pmod{s} \text{ для всех } i\}, \tag{2}$$

Конечно, в случае когда A есть подмножество S, имеем просто $C_A = A^n \cap C_S$. Важно также отметить, что если S рассматривается как циклическая группа порядка s, то C_S имеет структуру *смежного класса группового кода* над S.

В качестве первого результата дадим простую характеристику корректирующей способности кода $C = C_A$ при любом алфавите A.

Теорема 1. *Пусть* $S = \{0, 1, \ldots, s-1\}$, *где* s — *период* G, *и пусть* E — *это подмножество* S^n, *обладающее тем свойством, что элементы* $\mathbf{e}\mathbf{h}^T$ *из* G *всегда различны для разных* $\mathbf{e}$ *из множества* E. *Тогда код* C *с проверочной* $(n+1)$-*последовательностью* $(h_0, \mathbf{h})$ *исправляет все ошибки из множества* E.

Доказательство. Для произвольного передаваемого слова $\mathbf{v}$ по определению $\mathbf{v}\mathbf{h}^T = h_0$. Вектор ошибки $\mathbf{e} \in E$ и полученный вектор $\mathbf{x} \in \mathbb{Z}^n$ связаны соотношением $x_i \equiv v_i + e_i \pmod{s}$. Следовательно, «синдром» $b = \mathbf{x}\mathbf{h}^T - h_0$ равен $\mathbf{e}\mathbf{h}^T$. Теперь, согласно предположению, существует единственный вектор $\mathbf{e} \in E$, удовлетворяющий равенству $\mathbf{e}\mathbf{h}^T = b$. Это показывает, что $\mathbf{e}$ и, следовательно, $\mathbf{v}$ однозначно определяются по полученному вектору, что и доказывает утверждение теоремы. ∎

Амплитуда $\|\mathbf{x}\|$ вектора $\mathbf{x} \in \mathbb{Z}^n$ определяется как его l_1-норма, т. е. $\|\mathbf{x}\| = |x_1| + |x_2| + \ldots + |x_n|$. В этой статье мы в основном интересуемся кодами, исправляющими все ошибки, амплитуда которых не превышает заданного целого числа t. Эти коды будем называть *t-кодами*. Чтобы применить теорему 1 к этому случаю, достаточно положить $E = \{\mathbf{e} \in S^n: \|\mathbf{e}\| \leqslant t\}$.

Легко описать 1-*коды максимальной длины*. Типичный код C этого класса определяется соотношением (1) с проверочной $(n+1)$-последовательностью $(h_0, \mathbf{h})$, где $h_1, h_2, \ldots, h_n$ — различные ненулевые элементы данной абелевой группы G (порядка $n+1$) и h_0 — произвольный элемент G (см. [18] и [3]). Из теоремы 1 непосредственно следует, что C действительно является 1-кодом для любого описанного выше аддитивного канала. Кроме того, оказывается, что все t-коды с $t \geqslant 1$ могут быть получены укорочением 1-кода максимальной длины. (В точности так же, как в классической теории кодирования, все линейные коды с минимальным расстоянием $\alpha \geqslant 3$ могут рассматриваться как укороченные коды Хэмминга.)

Построение t-кодов с $t \geqslant 2$ не так очевидно. Сейчас будет описано важное семейство таких кодов, открытие которых по существу принадлежит Варшамову [20]. Пусть $\alpha_1, \alpha_2, \ldots, \alpha_n$ обозначают n различных ненулевых элементов поля Галуа $F = GF(p^m)$, где p — простое число и m — натуральное число.

Для данного $t \geqslant 1$ построим $t \times n$-матрицу

$$H = \begin{bmatrix} \alpha_1 & \alpha_2 & \dots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \dots & \alpha_n^2 \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \alpha_1^t & \alpha_2^t & & \alpha_n^t \end{bmatrix}. \tag{3}$$

Кроме того, пусть $\lambda = (\lambda_1, \lambda_2, \dots, \lambda_t)$ обозначает любую t-последовательность элементов $\lambda_k \in F$. Тогда *обобщенный код Варшамова* определяется как множество n-последовательностей $\mathbf{v}$ над алфавитом A, удовлетворяющих соотношению $\mathbf{v}H^T = \lambda$. (Варшамов рассматривал случай $m = 1$.) Очевидно, что эти коды охватываются общей конструкцией (1); подходящей группой G является аддитивная группа F^t (т. е. элементарная абелева p-группа порядка p^{mt}), а проверочной $(n+1)$-последовательностью служит матрица (λ^T, H), столбцы которой рассматриваются как элементы G. Отметим, что $s = p$.

Теорема 2. *Обобщенный код Варшамова с проверочной матрицей* (λ^T, H) *является* t*-кодом при условии, что* t *не превышает* $p - 1$.

Доказательство. Представленное ниже рассуждение непосредственно приводит к алгоритму декодирования. Ясно, что оно вдохновлено классическим подходом к кодам БЧХ и близким к ним; в этом отношении особенно см. [1]. Как было определено в доказательстве теоремы 1, синдром, соответствующий вектору ошибки $\mathbf{e}$, есть вектор $(\beta_1, \dots, \beta_t)^T$, задаваемый соотношениями

$$\beta_k = \sum_{i=1}^{n} e_i \alpha_i^k, \quad k = 1, \dots, t. \tag{4}$$

Таким образом, β_k есть k-й степенной момент совокупности, в которой α_i встречается e_i раз $(i = 1, \dots, n)$. Далее, пусть σ_j обозначает j-элементарную симметрическую функцию, определенную на той же самой совокупности. Полагая $w = \|\mathbf{e}\|$ и предполагая, что $w \leqslant t$, получим тождество

$$\sum_{j=1}^{t} (-1)^j \sigma_j z^{t-j} = z^{t-w} \prod_{i=1}^{n} (z - \alpha_i)^{e_i}. \tag{5}$$

(Заметим, что σ_j обращается в нуль при $j > w$.) Теперь $\sigma_1, \sigma_2, \dots, \sigma_t$ могут быть вычислены из компонент синдрома (4) посредством тождеств Ньютона. Наконец, целые числа e_i однозначно определяются из разложения на множители многочлена

$\sigma(z)$, стоящего в левой части (5). Итак, мы показали, что все ошибки $\mathbf{e}$ с амплитудой $\|\mathbf{e}\| \leqslant t$ исправимы, и дали набросок метода декодирования. ∎

Подчеркнем теперь одно интересное применение теоремы 2 в классическом случае, когда $A = S = \{0, 1, \ldots, p-1\}$ и $f_e(a)$ равно вычету $a + e$ по модулю p для a и e из множества S. В этой ситуации обобщенный код Варшамова C является смежным классом укороченного кода БЧХ над полем $GF(p)$ с конструктивным расстоянием Хэмминга $t + 1$. Следовательно, код C способен исправлять все ошибки, вес Хэмминга которых не превышает $t/2$. Теорема 2 утверждает, что код C можно использовать для исправления других типов ошибок, а именно тех, амплитуда которых не превышает t. Например, при $p = 3$ и $t = 2$ описанный выше алгоритм исправляет все одиночные ошибки и, кроме того, все двойные ошибки, ненулевые координаты которых равны 1.

В случае когда алфавит A есть подмножество множества ошибок $S = \{0, 1, \ldots, s-1\}$, а функция ошибок $f_e(a)$ есть сложение целых чисел $a + e$, следует отметить толкование t-кодов, которое можно дать на языке метрических пространств. С этой целью рассмотрим естественное обобщение асимметрического минимального расстояния, введенного Рао и Чоулой [17] для двоичных кодов[1]). Для данных двух n-последовательностей $x = (x_1, \ldots, x_n)$ и $y = (y_1, \ldots, y_n)$ с целочисленными координатами $x_i y_i$ определим асимметрическое расстояние

$$d(\mathbf{x}, \mathbf{y}) = \max\left\{\sum_{x_i \leqslant y_i} (y_i - x_i), \sum_{y_i \leqslant x_i} (x_i - y_i)\right\}, \tag{6}$$

которое действительно обладает свойствами метрики. *Минимальное асимметрическое расстояние* d_C произвольного кода C над алфавитом A есть минимальное значение, принимаемое величиной $d(\mathbf{x}, \mathbf{y})$ по всем парам различных кодовых слов $\mathbf{x}$ и $\mathbf{y}$ из C. Читатель легко убедится сам, что в рассматриваемой ситуации t-коды C — это в точности те коды, у которых минимальное асимметрическое расстояние $d_C \geqslant t + 1$. Из этого свойства t-кодов в двоичном случае (т. е. при $A = \{0, 1\}$) вытекает следующее. Множество D_k всех слов кода C с фиксированным весом Хэмминга k обладает минимальным расстоянием Хэмминга, не меньшим $2d_C \geqslant 2t + 2$. Кроме того, введение дополнительной координаты для объединения множеств D_k и D_{k-1} дает код Γ_k длины $n' = n + 1$ и постоянного веса k с минимальным расстоянием не менее $2t + 2$. Некоторые результаты относительно этой конструкции будут упомянуты впоследствии (см. также [8]).

[1]) Это расстояние было введено ранее Р. Р. Варшамовым (ДАН СССР, 1964, т. 157, 3, 546—548). — *Прим. перев.*

3. НУМЕРАТОРЫ СПЕКТРА

В этом разделе алфавит A предполагается конечным. Рассматривается следующий вопрос: как определить мощность, или, более общо, нумератор спектра кода C, задаваемого соотношением (1). Важность знания величины $|C|$ совершенно очевидна. Поводом для вычисления нумераторов спектра служит в основном построение двоичных кодов постоянного веса, упомянутых в конце разд. 2. Кроме того, укажем, что нумератор спектра кода C_S позволяет непосредственно получить мощность кода C_A для произвольного алфавита A.

Начнем с предварительного результата, который был впервые упомянут Мак-Элисом [14] в случае кодов Варшамова [1]).

Теорема 3. *Для любого алфавита A и любой n-последовательности* $\mathbf{h}$ *над группой G существует элемент $h_0 \in G$, такой что для соответствующего кода* (1) *справедливо соотношение* $|C| \geqslant |A|^n |G|^{-1}$.

Доказательство. Пусть $C[h_0]$ обозначает код (1). Поскольку совокупность всех непустых кодов $C[h_0]$ дает разбиение множества A^n, то средняя мощность $C[h_0]$ равна $|A^n||G|^{-1}$, что и доказывает теорему. ∎

В случае алфавита $A = S = \{0, 1, \ldots, s-1\}$, где s — период группы G, коды $C[h_0]$ являются смежными классами группового кода $C[0]$, так что все эти коды имеют мощность, равную $s^n|G|^{-1}$ [2]). Для алфавита общего вида A задача намного сложнее. Мы получим выражение для мощности C_A, используя *преобразование Фурье*; наш метод непосредственно связан с введенным в классической работе Мак-Вильямс [11].

Определим s-ичный *спектр* вектора $\mathbf{v} = (v_1, v_2, \ldots, v_n) \in \mathbb{Z}^n$ как s-последовательность $c(\mathbf{v}) = (c_0(\mathbf{v}), c_1(\mathbf{v}), \ldots, c_{s-1}(\mathbf{v}))$, где $c_k(\mathbf{v})$ равно числу координат $i \in \{1, \ldots, n\}$, таких что $v_i \equiv k \pmod s$. Тогда нумератор спектра кода C есть однородный многочлен $C(\mathbf{z}) = C(z_0, z_1, \ldots, z_{s-1})$ степени n относительно переменных z_k, в котором коэффициент при одночлене

$$\mathbf{z}^{\tau} = z_0^{\tau_0} z_1^{\tau_1} \ldots z_{s-1}^{\tau_{s-1}} \tag{7}$$

равен числу кодовых слов $\mathbf{v} \in C$, имеющих спектр $c(\mathbf{v}) = \tau = (\tau_0, \tau_1, \ldots, \tau_{s-1})$. Таким образом, используя обозначение (7),

---

[1]) Этот результат, состоящий в том, что средняя мощность кодов (1) равна объему пространства, деленному на число кодов, использовался для 1-кодов еще в работе [21]. — *Прим. перев.*

[2]) Здесь существенно используется сделанное ранее предположение о том, что h_i порождают группу G, иначе некоторые коды $C[h_0]$ будут пустыми. — *Прим. перев.*

можно записать

$$C(\mathbf{z}) = \sum_{v \in C} \mathbf{z}^{c(v)}. \tag{8}$$

В частности, подстановка в (8) $\mathbf{z} = \mathbf{1} = (1, 1, \ldots, 1)$ показывает, что $C(\mathbf{1}) = |C|$. В силу (2) нумератор спектра кода $C = C_A$ легко выводится из нумератора спектра кода C_S. Действительно, пусть $\boldsymbol{\theta} = (\theta_0, \theta_1, \ldots, \theta_{s-1})$ есть s-ичный спектр алфавита A, где по определению θ_k равно числу символов $a \in A$, удовлетворяющих сравнению $a \equiv k \pmod{s}$. Тогда нумератор спектра кода C_A задается формулой

$$C_A(\mathbf{z}) = C_S(\boldsymbol{\theta}\,\mathbf{z}), \tag{9}$$

где $\boldsymbol{\theta}\mathbf{z}$ означает покомпонентное произведение s-последовательностей $\boldsymbol{\theta}$ и $\mathbf{z}$. Доказательство этого результата элементарно и предоставляется читателю. (В действительности (9) немедленно следует из полученного ниже явного выражения.) Подстановка $\mathbf{z} = \mathbf{1}$ в (9) показывает, каким образом значения, принимаемые нумератором кода C_S, дают мощности всех кодов C_A: для произвольного алфавита A со спектром θ выполняется $C_S(\boldsymbol{\theta}) = |C_A|$.

Укажем, в частности, на двоичный случай $A = \{0, 1\}$. Очевидно, что нумератор спектра $C(\mathbf{z})$ зависит только от z_0 и z_1. Используя нестрогие обозначения, запишем $C(z) = C(1, z, *, \ldots, *)$. Этот многочлен $C(z)$ является просто классическим *нумератором весов* двоичного кода C. Он получается из нумератора спектра кода C_S подстановкой в (9) $\boldsymbol{\theta} = (1, 1, 0, \ldots, 0)$.

Будет показано, что полезно располагать матричным представлением проверочной $(n+1)$-последовательности $(h_0, \mathbf{h})$. С этой целью рассмотрим каноническое представление группы G в виде прямого произведения *инвариантных множителей* $G^{(k)}$; см., например, [4]. Существуют однозначно определенные целые числа $s_1, s_2, \ldots, s_r$ ($s_k \geqslant 2$ для всех k), удовлетворяющие условиям делимости $s_k | s_{k-1}$ и такие, что имеет место изоморфизм

$$G \cong G^{(1)} \times G^{(2)} \times \ldots \times G^{(r)}, \tag{10}$$

где G^k обозначает циклическую группу вычетов по модулю s_k. Отметим, что $s_1 = s$ и $\Pi s_k = |G|$. Полагая $c_k = s/s_k$, построим следующим образом r-вектор $\mathbf{b}_0$ и $r \times n$-матрицу B над множеством $S = \{0, 1, \ldots, s-1\}$

$$\mathbf{b}_0 = \begin{bmatrix} c_1 h_0^{(1)} \\ c_2 h_0^{(2)} \\ \cdot \\ \cdot \\ \cdot \\ c_r h_0^{(r)} \end{bmatrix}, \quad B = \begin{bmatrix} c_1 h_1^{(1)} & c_1 h_2^{(1)} & \ldots & c_1 h_n^{(1)} \\ c_2 h_1^{(2)} & c_2 h_2^{(2)} & \ldots & c_2 h_n^{(2)} \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ c_r h_1^{(r)} & c_r h_2^{(r)} & \ldots & c_r h_n^{(r)} \end{bmatrix}. \tag{11}$$

где $h_i^{(k)}$ означает k-ю компоненту элемента $h_i \in G$ в представлении (10)[1]). Определим теперь X как s-ичный групповой код длины n, порожденный по модулю s строками матрицы B, и рассмотрим произвольный вектор $e \in \mathbb{Z}^n$, удовлетворяющий соотношению $\mathbf{e}B^T = \mathbf{b}_0^T \pmod s$. Заметим, что X изоморфен G. Как следует из (1), код C состоит из тех векторов над A, которые являются сдвигами на вектор $\mathbf{e}$ векторов, ортогональных по модулю s коду X. Таким образом,

$$C = \bigcap_{\mathbf{x} \in X} \{\mathbf{v} \in A^n : \langle \mathbf{v} - \mathbf{e}, \mathbf{x} \rangle \equiv 0 \pmod s\}, \tag{12}$$

где $\langle \mathbf{u}, \mathbf{x} \rangle = u_1 x_1 + \ldots + u_n x_n$. В частности, код C_S оказывается смежным классом *ортогонального дополнения* X' к коду X. Действительно, соотношение (12) дает $C_S = X' + \mathbf{e}$.

Теперь мы можем сформулировать весьма общую теорему о спектральных нумераторах. Похожие результаты и похожие методы изложены в работах [12] и [5]. Для двух произвольных элементов x и y группы G определим их скалярное произведение $[x, y]$ как следующее целое число:

$$[x, y] = \sum_{k=1}^{r} c_k x^{(k)} y^{(k)}, \tag{13}$$

где $c_k = s/s_k$, а $x^{(k)}$ и $y^{(k)}$ — это k-е компоненты элементов x и y соответственно в представлении (10).

Теорема 4. *Пусть* ω *— примитивный комплексный корень степени* s *из единицы. Спектральный нумератор* $C(\mathbf{z})$ *кода* (1), *задаваемого над алфавитом* A *с* s*-ичным спектром* $\boldsymbol{\theta} = (\theta_0, \theta_1, \ldots, \theta_{s-1})$ *проверочной* $(n+1)$*-последовательностью* $(h_0, \mathbf{h})$, *дается выражением*

$$c(\mathbf{z}) = |G|^{-1} \sum_{x \in G} \omega^{-[x, h_0]} \prod_{i=1}^{n} \left(\sum_{k=0}^{s-1} \theta_k z_k \omega^{k[x, h_i]} \right) \tag{14}$$

Доказательство. Определим Ω как матрицу преобразования Фурье порядка s с (j, k)-элементом $\Omega_{j,k} = \omega^{jk}$ при $j, k = 0, 1, \ldots, s-1$. Начнем со следующих тождеств, где x обозначает произвольную целочисленную n-последовательность:

$$((\boldsymbol{\theta}\mathbf{z})\,\Omega)^{c(\mathbf{x})} = \prod_{i=1}^{n} \left(\sum_{k=0}^{s-1} \theta_k z_k \omega^{kx_i} \right) =$$

$$= \sum_{\mathbf{u} \in S^n} \omega^{\langle \mathbf{u}, \mathbf{x} \rangle} (\boldsymbol{\theta}^{c(\mathbf{u})} \mathbf{z}^{c(\mathbf{u})}) = \sum_{\mathbf{v} \in A^n} \omega^{\langle \mathbf{v}, \mathbf{x} \rangle} \mathbf{z}^{c(\mathbf{v})}. \tag{15}$$

[1]) В (11) и ниже $h_i^{(k)}$ следует рассматривать как целые числа, а не как элементы $G^{(k)}$. В противном случае, например при $r = 2$, $s_1 = p^2$, $s_2 = p$, получим вторую строку в матрице B из одних нулей и (12) неверно. — *Прим. перев.*

(Первое выражение получается непосредственно; второе следует из дистрибутивности и определения спектра $c(\mathbf{u})$; третье следует из того факта, что каждый вектор $\mathbf{u} \in S^n$ со спектром $c(\mathbf{u}) = \boldsymbol{\tau}$ дает (ввиду (2)) $\boldsymbol{\theta}\boldsymbol{\tau}$ векторов $\mathbf{v} \in A^u$ с $c(\mathbf{v}) = \boldsymbol{\tau}$.) Умножая обе части (15) на $\omega^{-\langle \mathbf{e}, \mathbf{x}\rangle}$ и суммируя по коду X, получим

$$\sum_{\mathbf{x} \in X} \omega^{-\langle \mathbf{e}, \mathbf{x}\rangle} ((\boldsymbol{\theta}\mathbf{z})\,\Omega)^{c(\mathbf{x})} = \sum_{\mathbf{v} \in A^n} \left(\sum_{\mathbf{x} \in X} \omega^{\langle \mathbf{v}-\mathbf{e}, \mathbf{x}\rangle} \right) \mathbf{z}^{c(\mathbf{v})} \tag{16}$$

Векторы $\mathbf{v}$ кода C характеризуются, ввиду соотношения (12), условием $\langle \mathbf{v} - \mathbf{e}, \mathbf{x}\rangle \equiv 0 (\mathrm{mod}\, s)$ для всех $\mathbf{x} \in X$. Следовательно, сумма по множеству X, стоящая в правой части выражения (16), в силу хорошо известного свойства преобразования Фурье равняется $|X|$, когда $\mathbf{v} \in C$, и нулю в противном случае. В результате применение определения (8) дает замечательное тождество

$$C(\mathbf{z}) = |X|^{-1} \sum_{\mathbf{x} \in X} \omega^{-\langle \mathbf{e}, \mathbf{x}\rangle} ((\boldsymbol{\theta}\mathbf{z})\,\Omega)^{c(\mathbf{x})}. \tag{17}$$

Последний шаг доказательства совсем прост: Достаточно применить изоморфизм между группами X и G, устанавливаемый формулой

$$\mathbf{x} = x^{(1)}\mathbf{b}^{(1)} + x^{(2)}\mathbf{b}^{(2)} + \ldots + x^{(r)}\mathbf{b}^{(r)}, \tag{18}$$

где $\mathbf{b}^{(k)}$ означает k-й столбец матрицы B в (11). Тогда соотношение (17) дает желаемое выражение (14) Детали проверки оставляем читателю. ◥

В оставшейся части этой статьи мы применим теорему 3 к вычислению нумератора весов $C(z)$ некоторых двоичных кодов, указанных в разд. 2. Таким образом, нам потребуется частный случай $\boldsymbol{\theta} = (1, 1, 0, \ldots, 0)$ формулы (14), который даст соотношение

$$C(z) = |G|^{-1} \sum_{x \in G} \omega^{-[x, h_0]} \prod_{i=0}^{n} (1 + z\omega^{[x, h_i]}). \tag{19}$$

3.1. ДВОИЧНЫЕ 1-КОДЫ МАКСИМАЛЬНОЙ ДЛИНЫ

Большинство результатов этого раздела не новы (см. разд. 1). Они представлены здесь без подробных доказательств, в основном чтобы проиллюстрировать теорию. Для простоты рассмотрим вначале класс двоичных 1-кодов длины $n = s - 1$, введенных Варшамовым и Тененгольцем [21]. Типичный код C из этого класса по определению состоит из векторов $\mathbf{v} \in \{0, 1\}^n$, удовлетворяющих сравнению $v_1 + 2v_2 + \ldots + nv_n \equiv \lambda (\mathrm{mod}\, s)$, где λ — фиксированное целое число. Таким образом, формула

(19) приводится к виду

$$s(1+z)C(z)=\sum_{x=0}^{s-1}\omega^{-\lambda x}\prod_{i=0}^{s-1}(1+z\omega^{ix}). \tag{20}$$

В выражении (20) i-произведение зависит только от наибольшего общего делителя чисел x и s, обозначаемого в дальнейшем через (x, s). В действительности каждый сомножитель $1+z\omega^{j(x,\,s)}$ встречается в этом произведении (x, s) раз при $j=0, 1, \ldots, d-1$, где $d=s/(x, s)$. Следовательно, выражение (20) принимает вид

$$s(1+z)C(z)=\sum_{d\mid s}\left(\sum_{(k,\,d)=1}\omega^{-\lambda ks/d}\right)\left(\prod_{j=0}^{d-1}(1+z\omega^{js/d})\right)^{s/d}. \tag{21}$$

Очевидно, что в (21) j-произведение равно $1-(-z)^d$. С другой стороны, k-сумма равна $\mu(d_\lambda)\varphi(d)/\varphi(d_\lambda)$, где $d_\lambda=d/(d, \lambda)$, а φ и μ означают *функции Эйлера и Мёбиуса* соответственно. Доказательство этого основывается на элементарной теории чисел; см. [22]. В результате имеем явную формулу для нумератора весов кода C, а именно

$$s(1+z)C(z)=\sum_{d\mid s}\mu(d_\lambda)\frac{\varphi(d)}{\varphi(d_\lambda)}(1-(-z)^d)^{s/d}. \tag{22}$$

Отметим, что $C(z)$ зависит только от (λ, s), что, впрочем, сразу очевидно из замечания об изоморфизмах, сделанного в разд. 2. В качестве иллюстрации получим нумератор всех 1-кодов Варшамова — Тененгольца длины 8 (т. е. $s=9$). Для $(\lambda, 9)$, равного 9, 3 и 1, получим последовательно

$$C(z)=1+4z^2+6z^3+8z^4+6z^5+4z^6+z^8,$$
$$C(z)=z+3z^2+7z^3+7z^4+7z^5+3z^6+z^7,$$
$$C(z)=z+3z^2+6z^3+8z^4+6z^5+3z^6+z^7.$$

Обсудим теперь обобщение, рассмотренное Констэнтином и Рао [3]. Типичный код здесь состоит из векторов $\mathbf{v}\{0, 1\}^n$, удовлетворяющих проверочному уравнению $v_1h_1+\ldots+v_nh_n=\lambda$, где h_i — различные ненулевые элементы некоторой конечной абелевой группы G, а λ — произвольный фиксированный элемент группы G. В теореме 4 содержится явная формула для нумератора весов кода C. Вывод ее из (19) подобен тому, что был показан выше в случае циклической группы G, хотя и несколько сложнее. Детали доказательства опущены.

Теорема 4. *Пусть G — конечная абелева группа порядка q и периода s. Для данных чисел d и k, таких что d делит s, а k делит d, обозначим через $\pi(k, d)$ число элементов $x\in G$ периода d, удовлетворяющих сравнению $[x, \lambda]\equiv s/k \pmod s$, где*

скалярное произведение $[\cdot, \cdot]$ *определено так же, как в* (13). *Тогда нумератор весов* $C(z)$ *кода Константина — Рао* C *задается соотношением*

$$q(1+z)C(z) = \sum_{d \mid s} \Big(\sum_{k \mid d} \mu(d)\pi(k, d) \Big) (1-(-z)^d)^{q/d}. \qquad (23)$$

Замечание. Очевидно, что $\pi(k, d)$ обращается в нуль, когда k не делит период λ. Таким образом, в формуле (23) выражение $k \mid d$ может быть заменено на $k \mid (l, d)$, где l равно периоду λ. В случае циклической группы $\pi(k, d)$ обращается в 0 при $k \neq d_\lambda$ и равняется $\varphi(d)/\varphi(d_\lambda)$ при $k = d_\lambda$; таким образом, (23) сводится к (22).

Рассмотрим в качестве примера группу G, заданную параметрами $r = 2$, $s_1 = p^2$, $s_2 = p$, где p — простое число. Группа G, как оказывается, распадается на четыре орбиты под действием своей группы автоморфизмов. Эти орбиты содержат представленные в виде (10) элементы $\lambda = (0, 0)$, $(p, 0)$, $(0, 1)$ и $(1, 0)$ соответственно, а их мощность равна соответственно 1, $p-1$, $p(p-1)$ и $p^2(p-1)$. Вычисление величин $\pi(k, d)$ показывает, что две последние орбиты дают один и тот же нумератор весов. Используя матричную запись, полученные результаты можно представить в виде

$$p^3(1+z)\begin{bmatrix} C^1(z) \\ C^2(z) \\ C^3(z) \end{bmatrix} = \begin{bmatrix} 1 & p^2-1 & p^2(p-1) \\ 1 & p^2-1 & -p^2 \\ 1 & -1 & 0 \end{bmatrix} \begin{bmatrix} (1+z)^p \\ (1-(-z)^p)^p \\ (1-(-z)^{p^2})^p \end{bmatrix},$$

где $C^1(z)$, $C^2(z)$, $C^3(z)$ обозначают нумераторы весов $C(z)$ при выборе λ равным $(0, 0)$, $(p, 0)$ и $(1, 0)$ (или $(0, 1)$) соответственно.

Приведенный выше пример показывает, что обычно $C(z)$ зависит от λ, а не только от величины l, равной периоду λ, как это было в случае циклической группы, и в то же время различным величинам l может соответствовать один и тот же нумератор $C(z)$.

Достаточно общее разъяснение характера этого явления основано на теории двойственности колец Шура (см. [19] и [5]) и выходит за пределы данной статьи.

Подстановка $z = 1$ в (23) приводит к законченному выражению для мощности $|C| = C(1)$ кода. Из него очевидно, что для любой группы G мощность кода максимальна при выборе $\lambda = 0$ (этот результат был впервые доказан Константином и Рао [3]). Исследуя аналитическое выражение величины $|C|$, Мак-Элис и Родемич [15] и Хелесет и Клеве [9] сумели доказать справедливость сделанного Константином и Рао предположения, что

при заданном порядке q группы и $\lambda = 0$ наибольший код C дает группа G с максимальным числом элементарных делителей (инвариантов).

Теорема 4 позволяет описать хорошие равновесные коды с минимальным расстоянием 4 (соответствующая конструкция объясняется в конце разд. 2). Полученные таким образом результаты согласуются с результатами работы [8]. Поскольку в ней не рассмотрены все возможные группы, укажем на улучшение результатов этой работы для длины $n = 24$, которое дает группа G с $s_1 = 6$, $s_2 = s_3 = 2$ в двух случаях, а именно $|\Gamma_8| = 30789$ и $|\Gamma_{10}| = 112952$. Дадим также получаемые значения мощности максимальных кодов Γ_k длины $n' = 25$: $|\Gamma_2| = 12$, $|\Gamma_3| = 92$, $|\Gamma_4| = 506$, $|\Gamma_5| = 2130$, $|\Gamma_6| = 7034$, $|\Gamma_7| = 19\,228$, $|\Gamma_8| = 43\,263$, $|\Gamma_9| = 81\,719$, $|\Gamma_{10}| = 130\,760$, $|\Gamma_{11}| = 178\,296$, $|\Gamma_{12}| = 208\,012$. Все эти «оптимальные» коды получаются с помощью нециклической группы порядка 25 и выбора $\lambda = 0$.

Сделаем последнее замечание, касающееся численных результатов нашего анализа. Для данного порядка q распределения весов различных кодов близки друг к другу и близки к нормированному биномиальному распределению. То же замечание может быть сделано и относительно кодов, анализируемых ниже.

3.2. ОБОБЩЕННЫЕ 2-КОДЫ ВАРШАМОВА

Последняя часть этой статьи посвящена двоичным обобщенным кодам Варшамова, определенным в разд. 2. Таким образом, типичный код C состоит из векторов $\mathbf{v} \in \{0, 1\}^n$, удовлетворяющих уравнениям $v_1\alpha_1^k + \ldots + v_n\alpha_n^k = \lambda_k$ при $k = 1, \ldots, t$, где α_i — различные ненулевые элементы поля $F = GF(p^m)$, тогда как λ_k — произвольные элементы поля F.

Пусть ψ — гомоморфизм поля F в его простое подполе $S = GF(p)$. Тогда формула (19), примененная к обобщенным кодам Варшамова, превращается в

$$p^{mt}C(z) = \sum_{f \in P} \omega^{-\psi(\langle f, \lambda\rangle)} \prod_{i=1}^{n} \left(1 + z\omega^{\psi(f(\alpha_i))}\right), \tag{24}$$

где через P обозначено пространство многочленов $f(x) = f_1x + f_2x^2 + \ldots + f_tx^t$ с коэффициентами $f_k \in F$, а $\langle f, \lambda\rangle$ означает $\sum f_k\lambda_k$. Когда t относительно мало, формула (24) эффективна с вычислительной точки зрения. Ниже мы будем рассматривать случай 2-кодов полной длины и сосредоточим внимание главным образом на простых полях.

Таким образом, пусть $m = 1$, $t = 2$ и $n = p - 1$, где p — нечетное простое число. Положим для удобства $\lambda = \lambda_1$ и $\mu = \lambda_2$.

Тогда (24) немедленно приводится к виду

$$p^2(1+z)C(z)=\sum_{a,\,b\in S}\omega^{-a\mu-b\lambda}\prod_{x\in S}(1+z\omega^{ax^2+bx}), \qquad (25)$$

где $S=\{0, 1, \ldots, p-1\}$. В правой части выражения (25) сгруппируем слагаемые. Очевидно, что слагаемое, соответствующее $a=b=0$, есть $\Sigma(0,0)=(1+z)^p$. Положив $S^*=\{1, 2, \ldots, p-1\}$, легко устанавливаем, что слагаемые, соответствующие $a=0$, $b\in S^*$, дают вклад, равный $\Sigma(0, S^*)=(p\delta_{0,\lambda}-1)\times(1+z^p)$, где δ — символ Кронекера. Затем пусть через Q и N обозначаются подмножества множества S^*, состоящие из квадратичных и неквадратичных вычетов по модулю p соответственно. Используя тождество $ax^2+bx=a(x+b/2a)^2-b^2/4a$, получим следующее выражение для вклада $a\in Q$, $b\in S$ в (25):

$$\Sigma(Q, S)=\sum_{a\in Q}\sum_{b\in S}\omega^{-a\mu-b\lambda}\left[(1+z\omega^{-b^2/4a})\prod_{c\in Q}(1+z\omega^{c-b^2/4a})^2\right]. \qquad (26)$$

Чтобы продолжить исследование выражения (26), введем многочлен

$$g(z)=(1+z)\prod_{c\in Q}(1+z\omega^c)^2=1+z^p+\sum_{k=1}^{p-1}g_kz^k. \qquad (27)$$

Коэффициенты g_k являются алгебраическими числами, принадлежащими к расширению $\mathbb{Q}[\theta]$ поля рациональных чисел $\mathbb{Q}$, порожденному числом θ

$$\theta=1+2\sum_{c\in Q}\omega^c=\pm(-1)^{(p-1)/4}p^{1/2} \qquad (28)$$

(см., например, Ленг [10]). В выражении (26), множитель, заключенный в квадратные скобки, есть $g(z\omega^{-b^2/4a})$. Следовательно, применяя (27), получим

$$\Sigma(Q, S)=(1+z^p)\sum_{a\in Q}\omega^{-a\mu}\sum_{b\in S}\omega^{-b\lambda}+$$
$$+\sum_{k=1}^{p-1}g_kz^k\left(\sum_{a\in Q}\sum_{b\in S}\omega^{(-a\mu-b\lambda-kb^2/4a)}\right). \qquad (29)$$

Первый член в выражении (29) легко определяется; ввиду хорошо известного свойства квадратичных вычетов он сводится к

$$\frac{p}{2}(1+z^p)\delta_{0,\lambda}(p\delta_{0,\mu}-1+\chi(-\mu)\theta), \qquad (30)$$

где $\chi(u)$ — *квадратичный характер* (т. е. символ Лежандра), принимающий значения 0, 1 и —1 в зависимости от того, обра-

щается ли вычет числа u по модулю p в нуль, принадлежит ли он Q или N. Рассмотрим теперь во втором члене выражения (29) сумму по a и b при фиксированном значении k. Положив $x = b + 2a\lambda/k$, выводим следующее выражение для искомой суммы:

$$\sum_{a\in Q}\sum_{x\in S}\omega^{a(-\mu+\lambda^2/k)-kx^2/4a} = \sum_{a\in Q}\omega^{a(-\mu+\lambda^2/k)}\left(1+2\sum_{c\in Q}\omega^{-kc}\right) =$$
$$= \tfrac{1}{2}(p\delta_{\mu,\,\lambda^2/k} - 1 + \chi(-\mu+\lambda^2/k)\,\theta)\,\chi(-k)\,\theta. \quad (31)$$

Это завершает вычисление $\Sigma(Q, S)$.

Вклад $\Sigma(N, S)$, вносимый в (25) элементами $a \in N$, $b \in S$, подсчитывается точно таким же образом, как и выше. Единственное отличие состоит в том, что число θ должно быть заменено на алгебраически сопряженное $\bar{\theta} = -\theta$; тем самым g_k должны быть заменены на $\bar{g}_k$. Собирая вместе все члены и используя формулы (30) и (31), получим

$$p^2(1+z)C(z) = \Sigma(0,0) + \Sigma(0, S^*) + \Sigma(Q, S) + \Sigma(N, S) =$$
$$= (1+z)^p + (p^2\delta_{0,\,\lambda}\delta_{0,\,\mu} - 1)(1+z^p) +$$
$$+ p\sum_{k=1}^{p-1}\chi(\lambda^2 - k\mu)\,R(g_k)\,z^k +$$
$$+ \sum_{k=1}^{p-1}\chi(-k)\,(p\delta_{\lambda^2,\,k\mu} - 1)\,R(\theta g_k)\,z^k, \quad (32)$$

где $R(\xi)$ обозначает $(\xi + \bar{\xi})/2$ для $\xi \in \mathbb{Q}[\theta]$.

Объясним теперь, как вычислить коэффициенты, появившиеся в (32). Определим многочлены $a(z)$ и $b(z)$ формальной степени $(p-1)/2$ с рациональными коэффициентами посредством уравнения

$$f(z) = \prod_{c\in Q}(1 + z\omega^c) = a(z) + \theta b(z). \quad (33)$$

Как оказывается, многочлены $2a(z)$ и $2b(z)$ имеют целые коэффициенты. По определению, $g(z) = (1+z)f^2z$. Отсюда, используя тождество $f(z)\bar{f}(z) = (1+z^p)/(1+z)$, из (33) и (28) легко выводится, что

$$R(\theta g(z)) = 2\chi(-1)\,p\,(1+z)\,a(z)\,b(z),$$
$$R(g(z)) = 1 + z^p + 2\chi(-1)\,p\,(1+z)\,b^2(z). \quad (34)$$

Подставляя в (32) значения $R(\theta g_k)$ и $R(g_k)$, полученные из (34), мы, наконец, придем к довольно удовлетворительному

выражению для многочлена $C(z)$, которое сформулируем в виде теоремы.

Теорема 5. *Пусть x_k и y_k обозначают коэффициенты при z^k в многочленах с рациональными коэффициентами $(1+z)a(z)b(z)$ и $(1+z)b^2z$ соответственно, где многочлены $a(z)$ и $b(z)$ определены в* (33). *Тогда нумератор спектра $C(z)$ двоичного 2-кода Варшамова длины $n=p-1$ при обозначениях $\lambda=\lambda_1$ и $\mu=\lambda_2$ задается выражением*

$$p^2(1+z)C(z)=(1+z)^p+(p^2\delta_{0,\lambda}\delta_{0,\mu}-1)(1+z^p)-2p\sum_{k=1}^{p-1}\chi(k)x_kz^k+$$

$$+2p^2\sum_{k=1}^{p-1}(\chi(k\mu-\lambda^2)y_k+\delta_{\lambda^2,\,k\mu}\chi(k)x_k)z^k. \qquad (35)$$

Интересно отметить, что каждый из четырех членов в (35) должен делиться на $1+z$. Для четвертого члена, в частности, это приводит к соотношению

$$x_j=-\sum_{k=1}^{p-1}(-1)^{k-j}\chi(k-j)y_k, \quad 1\leqslant j\leqslant p-1. \qquad (36)$$

Поэтому в многочлене $b(z)$ содержится вся необходимая информация для вычисления $C(z)$ при помощи (35). Приведем эту информацию для значений $p=17$, 19, 23 соответственно:

$$2b(z)=z-z^2+z^3-2z^4+z^5-z^6+z^7,$$
$$2b(z)=z-z^3-z^4+z^5+z^6-z^8,$$
$$2b(z)=z-z^2+z^4-2z^5+2z^6-z^7+z^9-z^{10}.$$

Отметим, что $b(-z)=\chi(-1)b(z)$. Укажем в качестве иллюстрации значения c мощности 2-кодов Варшамова длины 16 (при $p=17$) и количества $N(c)$ кодов данной мощности:

c:	223	224	226	227	228	230	231
$N(c)$:	8	65	64	16	96	32	8

Поразительным является малое отличие значений c от среднего значения, даваемого теоремой 3 (с другой стороны, небольшое число различных значений легко объяснить, приняв во внимание изоморфизмы в классе кодов Варшамова). Кроме того, выпишем опять для $n=16$ нумератор весов максимальных кодов (задаваемых в этом случае условием $\lambda=0$ и $\mu\in Q$):

$$C(z)=z^2+z^{14}+2(z^3+z^{13})+7(z^4+z^{12})+16(z^5+z^{11})+$$
$$+26(z^6+z^{10})+42(z^7+z^9)+43z^8.$$

Читатель может проверить эти результаты при помощи соотношения (35), используя (при $p = 17$) данное выше выражение для $b(z)$.

Возможно также получить явное выражение для (24) и когда $m \geqslant 2$, а остальные условия прежние: $t = 2$, p — простое нечетное число и $n = p^m - 1$. Приведем без доказательства результат для случая $m = 2$. (Приведенная ниже формула представляется с вычислительной точки зрения более простой, чем формула из теоремы 5.)

Теорема 6. *Пусть* $a_k(\zeta)$ *обозначает при* $k = 0, 1, \ldots, p-1$ *многочлен с целыми коэффициентами формальной степени* $p-1$, *определяемый из разложения*

$$\left(\frac{1+z^p}{1+z}\right)^p = \sum_{k=0}^{p-1} z^k a_k(z^p). \tag{37}$$

С другой стороны, пусть χ *обозначает квадратичный характер поля Галуа* $F = GF(p^2)$, *определенный следующим образом:* $\chi(0) = 0$ *и* $\chi(u)$ *равно* 1 *или* -1 *в зависимости от того, является или нет элемент и квадратом в* F. *Нумератор весов* $C(z)$ *двоичного обобщенного 2-кода Варшамова полной длины* $n = p^2 - 1$ *при обозначениях* $\lambda_1 = \lambda$ *и* $\lambda_2 = \mu$ *задается выражением*

$$\begin{aligned} 2p^4(1+z)C(z) = {} & 2(1+z)^{p^2} + (1+z^p)^p\left(p^2\delta_{0,\lambda}\left(p^2\delta_{0,\mu} + 1 + p\chi(\mu)\right) - 2\right) + \\ & + p^2(1+z^p)\,\delta_{0,\lambda}\left(p^2\delta_{0,\mu} - 1 - p\chi(\mu)\right)a_0(z^p) + \\ & + p(1+z^p)^{p-1}\sum_{k=1}^{p-1}\binom{p}{k}\left(p^2\delta_{\lambda,k\mu} - 1 + p\chi(k\mu - \lambda^2)\right)z^k - \\ & - p(1+z^p)\sum_{k=1}^{p-1}\left(p^2\delta_{\lambda^2,k\mu} - 1 - p\chi(k\mu - \lambda^2)\right)z^k a_k(z^p). \end{aligned} \tag{38}$$

Авторы исследовали формулы (35) и (38), с тем чтобы указать хорошие равновесные коды с минимальным расстоянием 6 (см. теорему 2 и последний абзац разд. 2). Полученные таким образом численные значения совпали с приведенными Грехэмом и Слоэном [8] для длин $n' \leqslant 24$. Поскольку случай $n' = 25$ не был упомянут этими авторами, приведем мощности максимальных кодов, получаемых из теоремы 6 (при $p = 5$); имеем $|\Gamma_3| = 8$, $|\Gamma_4| = 22$, $|\Gamma_5| = 100$, $|\Gamma_6| = 288$, $|\Gamma_7| = 796$, $|\Gamma_8| = 1738$, $|\Gamma_9| = 3299$, $|\Gamma_{10}| = 5360$, $|\Gamma_{11}| = 7152$, $|\Gamma_{12}| = 8323$.

Наконец упомянем, что выбор пары (λ, μ), приводящей к максимальному коду C, в отличие от разд. 2, совсем не очеви-

ден. Для нескольких значений длины n ниже указан оптимальный выбор (который в рассмотренных примерах оказывается единственным с точностью до изоморфизма) вместе с соответствующим значением $|C|$:

n:	12	16	18	22	24
(λ, μ):	(1, 0)	(0, 1)	(0, 0)	(1, 2)	(0, 0)
(C):	29	231	748	7946	26 956

Благодарности

Авторы благодарны Р. Д. Мак-Элису и рецензенту, обратившим наше внимание на некоторые полезные ссылки.

ЛИТЕРАТУРА

[1] Berlecamp E. R. Algebraic Coding Theory, McGraw-Hill, New York, 1968.
[2] Bose B. and Rao T. R. N On the Theory of Theory Unidirectional Error Correcting/Detecting Codes. Dept. Comp. Sci., Southern Methodist University, Dallas, Texas, Technical Report, CS7817, 1978.
[3] Constantin S. D., Rao T. R. N. On the theory of binary asymmetric error correcting codes. Inform. Contr., 40, 1979, 20—36.
[4] Curtis C. W., Reiner I. Representation Theory of Finite Groups and Associative Algebras, Wiley-Interscience, New York, 1962. [Имеется перевод: Кэртис Ч., Райнер И. Теория представлений конечных групп и ассоциативных алгебр. — М.: Наука, 1969.]
[5] Delsarte P. An Algebraic Approach to the Association Schemes of Coding Theory. Philips Res. Repts Suppl. 10, 1973. [Имеется перевод: Дельсарт Ф. Алгебраический подход к схемам отношений теории кодирования. — М.: Мир, 1976.]
[6] Delsarte P. Partial-optimal piecewise decoding of linear codes. IEEE Trans. Inform. Theory, IT-24, 1978, 70—75.
[7] Гинзбург Б. Д. Об одной теоретико-числовой функции, имеющей приложение в теории кодирования. — В сб.: Проблемы кибернетики, вып. 19. — М.: Наука, 1967, 249—252.
[8] Graham R. L., Sloane N. J. A. Lower bouds for constant weight codes, IEEE Trans. Inform. Theory, IT-26, 1980, 37—41.
[9] Helleseth T., Klove T. On group theoretic codes for asymmetric channels, Inform. Contr., 1981.
[10] Lang S. Algebra, Addison-Wesley, Reading, Mass, 1965. [Имеется перевод: Ленг С. Алгебра. — М.: Мир, 1968.]
[11] MacWilliams F. J. A theorem on the distribution of weights in a systematic code, Bell System Tech. J., 42, 1963, 79—94.
[12] MacWilliams F. J., Sloane N. J., Goethals J. M. The MacWilliams identities for nonlinear codes, Bell System Tech. J, 51, 1972, 803—819.
[13] Мазур Л. Е. О некоторых кодах, исправляющих несимметрические ошибки. — Проблемы передачи информации, 10, 4, 1974, 40—46.
[14] McEliece R J. A comment on «A class of codes for asymmetric channels and a problem from the additive theory of numbers», IEEE Trans. Inform. Theory IT-19, 1973, 137.
[15] McElice R. J., Rodemich E. R. The Constantin-Rao construction for binary asymmetric error-correcting codes, Inform Contr., 44, 1980, 187—196.

[16] Налбандян М. Н. Заметка о двух классах нелинейных кодов. — Проблемы передачи информации, 10, 2, 1974, 61—63.
[17] Rao T. R. N., Chawla A. S. Asymmetric error correcting codes for some LSI semiconductor memories, in «The Annual Southestern Symposium of System Theory, 1975, 170—171.
[18] Stanley R. P., Yoder M. F. Study of Varshamov Codes for Asymmetric Channes, Jet Propulsion Laboratory Technical Report 32-1526, Vol. 14, 1973, 117—122.
[19] Tamaschke O. Zur Theorie der Permutationsgruppen mit regulärer Untergruppe, I and II, Math. 80, 1963, 328—354, 433—465
[20] Varshamov R. R. A class of codes for asymmetric channels and a problem from the additive theory of numbers, IEEE Trans. Inform Theory, IT-19, 1973, 92—95.
[21] Варшамов Р. Р., Тененгольц Г. М. Код, исправляющий одиночные несимметрические ошибки. — Автоматика и телемеханика, 26, № 2, 1965, 288—292.
[22] Виноградов И. М. Основы теории чисел. — М.: Наука, 1981.

О семействах Шпернера, удовлетворяющих дополнительному условию[1)]

П. Франкл[2)]

Пусть $\mathcal{F}$ — семейство Шпернера, состоящее из подмножеств конечного множества X мощности n, такое что объединение любых трех множеств, принадлежащих семейству $\mathcal{F}$, отлично от X.

В этой статье мы доказываем, что для достаточно больших n (например, для $n > 1000$)

$$|\mathcal{F}| \leqslant \binom{n-1}{\left[\frac{n-1}{2}\right]} + \varepsilon,$$

где $\varepsilon = 0$ при нечетных n и $\varepsilon = 1$ при четных n. Найдены также экстремальные семейства.

Кроме того, мы доказываем обобщение теоремы Эрдеша — Ко — Радо, которое используется при доказательстве основной теоремы.

1. ВВЕДЕНИЕ И ОБОЗНАЧЕНИЯ

Пусть X — конечное множество. Семейство $\mathcal{F}$ подмножеств множества X называется семейством Шпернера, если ни одно из подмножеств, принадлежащих семейству $\mathcal{F}$, не содержится в другом.

Обозначим мощность $\mathcal{F}$ через $|\mathcal{F}|$, а мощность X через $|X|$.

Шпернер [6] доказал, что если $|X| = n$, то для любого семейства Шпернера $\mathcal{F}$, состоящего из подмножеств множества X, имеет место соотношение[3)]

$$|\mathcal{F}| \leqslant \binom{n}{\left[\frac{n}{2}\right]}.$$

[1)] Frankl P. On Sperner families satisfying an additional condition — J. Comb. Theory (A) 20, 1 — 11 (1976).

[2)] Eötvös L. Univ. Budapest, Hungary. 1088.

[3)] Причем равенство имеет место тогда и только тогда, когда $\mathcal{F}$ состоит либо только из всех $\binom{}{\left[\frac{n}{2}\right]}$-элементных подмножеств множества X, либо только из всех $\binom{}{n-\left[\frac{n}{2}\right]}$-элементных подмножеств. — *Прим. перев.*

Э. Милнер доказал следующее обобщение теоремы Шпернера (оно содержится в [5], хотя там оно сформулировано для случая пересечений, а не объединений).

Пусть $\mathcal{F}$ — семейство Шпернера подмножеств множества X, $|X| = n$, и пусть для любых двух множеств A, B, принадлежащих семейству $\mathcal{F}$, выполняется соотношение

$$|A \cup B| \leqslant k \leqslant n;$$

тогда

$$|\mathcal{F}| \leqslant \binom{n}{\left[\frac{k}{2}\right]}.$$

После этого естественно задать вопрос: а что получится, если рассматривать объединения из трех (четырех и т. д.) множеств? В этой статье мы дадим ответ на этот вопрос для случая, когда $k = n - 1$, а n достаточно велико, доказав следующую теорему.

Теорема. *Пусть X — конечное множество мощности n, а $\mathcal{F}$ — семейство Шпернера, состоящее из подмножеств множества X и удовлетворяющее следующему дополнительному условию: для любых трех множеств F, G, $H \in \mathcal{F}$ выполняется соотношение*

$$F \cup G \cup H \neq X.$$

Тогда для достаточно больших n имеет место неравенство

$$|\mathcal{F}| \leqslant \binom{n-1}{\left[\frac{n-1}{2}\right]} + \varepsilon,$$

где $\varepsilon = 0$ при нечетных и $\varepsilon = 1$ при четных n.

Пусть $\mathcal{F}$ — семейство множеств. Будем обозначать через $\mathcal{F}_i$ подсемейство семейства $\mathcal{F}$, состоящее из всех i-элементных множеств, принадлежащих $\mathcal{F}$. Обозначим через $\mathcal{F}'$ семейство, состоящее из всех таких множеств F, для которых существует множество $G \in \mathcal{F}$, такое что $F \subset G$ и $|G \setminus F| = 1$.

2. НЕКОТОРЫЕ СВЕДЕНИЯ

Теорема Эрдеша — Ко — Радо [1]) утверждает, что если дано семейство Шпернера, состоящее из подмножеств множества мощности n, такое что мощность каждого из множеств, принадлежащих этому семейству, не превосходит i, где $2i \leqslant n$, и любые два множества, принадлежащие данному семейству, имеют непустое пересечение, то мощность этого семейства не превосходит

$$\binom{n-1}{i-1}.$$

[1]) См. также [7]. — *Прим. перев.*

Катона [1] дал простое доказательство этой теоремы. Мы обобщим его метод, чтобы получить следующую лемму.

Лемма. *Пусть X — конечное множество мощности n, k натуральное, и пусть $\mathcal{F}$ — такое семейство i-элементных подмножеств множества X, что для любых k множеств $F_1, \ldots, F_k \in \mathcal{F}$ $\bigcap_{j=1}^{k} F_j \neq \varnothing$. Тогда если $ki/(k-1) \leqslant n$, то*

$$|\mathcal{F}| \leqslant \binom{n-1}{i-1}.$$

Доказательство. Пусть $\mathcal{F}^c = \{K \mid X \setminus K \in \mathcal{F}\}$. Тогда каждое множество, принадлежащее $\mathcal{F}^c$, имеет мощность $n - i \geqslant n/k$, и для любых k множеств $K_1, \ldots, K_k \in \mathcal{F}^c$ выполняется соотношение $\bigcup_{j=1}^{k} K_j \neq X$. Пусть $x_1, x_2, \ldots, x_n, x_1$ — некоторое циклическое упорядочение элементов множества X. Будем оценивать число тех множеств $K \in \mathcal{F}^c$, которые состоят из $n - i$ последовательных элементов относительно этого циклического упорядочения. Если существует по крайней мере одно такое множество K, то можно считать, что его последним элементом является x_n (здесь и в дальнейшем «последний» означает такой элемент, что его сосед справа не содержится в K). С каждым множеством $K \in \mathcal{F}^c$, состоящим из последовательных элементов относительно данного упорядочения, свяжем число — индекс его последнего элемента, а с множеством, последним элементом которого является x_n, свяжем совокупность всех целых чисел s из промежутка $n \leqslant s \leqslant k(n-i)$. Если имеется r множеств, состоящих из последовательных элементов относительно данного циклического упорядочения, то с ними окажутся связанными $r + k(n - i) - n$ чисел из промежутка $[1, k(n-i)]$. Разобьем элементы этого промежутка на $n - i$ непересекающихся классов так, чтобы элементы каждого из этих классов имели одинаковый остаток по модулю $n - i$. Допустим, что среди множеств, принадлежащих семейству $\mathcal{F}^c$, можно найти k множеств, состоящих из последовательных элементов относительно данного циклического упорядочения, таких, чтобы связанные с ними индексы полностью покрывали один из классов. Тогда, как легко видеть, объединение этих множеств покроет множество X. Следовательно, в силу сделанного допущения в каждом из этих классов существует элемент, с которым не связано ни одно из множеств семейства $\mathcal{F}^c$. А так как с различными множествами связаны различные индексы, имеем $-n + r + k(n-i) + n - i \leqslant k(n-i)$, откуда $r \leqslant i$.

Подсчитывая число пар, состоящих из циклического упоря-

дочения и множества последовательных элементов относительно этого упорядочения, получим [1])

$$|\mathcal{F}^c|i!(n-i)! \leqslant (n-1)!i.$$

Следовательно, $|\mathcal{F}^c| \leqslant \binom{n-1}{i-1}$, что и требовалось доказать.

Мы будем использовать следующую оценку, которая следует из теоремы Краскала — Катоны (см. [2] и [3]) [2]).

Пусть $\mathcal{A}$ — семейство k-элементных подмножеств конечного множества X. Если $|\mathcal{A}| \leqslant \binom{s}{k}$, то

$$|\mathcal{A}'| \geqslant |\mathcal{A}| \binom{s}{k-1} \Big/ \binom{s}{k}, \qquad (*)$$

причем равенство имеет место тогда и только тогда, когда $\mathcal{A}$ состоит из всех k-элементных множеств некоторого s-элементного подмножества X. Теперь мы можем доказать следующее обобщение теоремы Эрдеша — Ко — Радо.

Теорема 1. *Пусть X — конечное множество мощности n. Пусть $\mathcal{A}$ — семейство Шпернера, состоящее из подмножеств множества X, мощность каждого из которых не превышает i. Пусть любые k множеств, принадлежащих семейству $\mathcal{A}$, имеют непустое пересечение и $ki \leqslant (k-1)n$. Тогда справедливо неравенство*

$$\sum_{j=1}^{i} \frac{|\mathcal{A}_j|}{\binom{n-1}{j-1}} \leqslant 1. \qquad (1)$$

Замечание. Эта теорема действительно является обобщением теоремы Эрдеша — Ко — Радо, так как при $k = 2$ имеем $2i \leqslant n$, откуда, учитывая, что величина $\binom{n-1}{i-1}$ монотонно возрастает при $i \leqslant n/2$, получаем

$$1 \geqslant \sum_{j=1}^{n} \frac{|\mathcal{A}_j|}{\binom{n-1}{j-1}} \geqslant \frac{\sum_{j=1}^{i} |\mathcal{A}_j|}{\binom{n-1}{i-1}} = \frac{|\mathcal{A}|}{\binom{n-1}{i-1}}.$$

[1]) С одной стороны, каждое множество, принадлежащее $\mathcal{F}^c$, может составить пару с $i!(n-i)!$ циклическими упорядочениями, и поэтому семейство $\mathcal{F}^c$ может составить $|\mathcal{F}^c|i!(n-i)!$ пар. С другой стороны, по доказанному выше, каждое циклическое упорядочение может составить пару с не более чем i множествами из $\mathcal{F}^c$, а всего циклических упорядочений $(n-1)!$ — *Прим. перев.*

[2]) См. также [8]. — *Прим. перев.*

Доказательство. При $i = 1$ теорема тривиальна. Будем вести индукцию по числу непустых подсемейств $\mathscr{A}_j$.

Если это число равно 1, то теорема следует из приведенной выше леммы. В противном случае пусть p и r, $p < r$, — два наименьших индекса, для которых $|\mathscr{A}_j| \neq 0$. Определим семейство $\mathscr{C}_r$ соотношением

$\mathscr{C}_r = \{\mathscr{C} \mid |\mathscr{C}| = r$ и существует $A \in \mathscr{A}_p$, такое что $\mathscr{C} \supset A\}$. Тогда, используя оценку $(*)$ $(r - p)$ раз, получаем [1])

$$|\mathscr{C}_r| \geqslant |\mathscr{A}_p| \frac{\binom{n-1}{r-1}}{\binom{n-1}{p-1}}.$$

Семейство $\mathscr{B} = (\mathscr{A} \setminus \mathscr{A}_p) \cup \mathscr{C}_r$ также удовлетворяет условиям теоремы, причем число непустых подсемейств $\mathscr{B}_j$ в этом семействе на единицу меньше, чем в семействе $\mathscr{A}$. Следовательно, в силу индуктивного предположения имеем [2])

$$\sum_{j=1}^{i} \frac{|\mathscr{A}_j|}{\binom{n-1}{j-1}} \leqslant \sum_{j=1}^{i} \frac{|\mathscr{B}_j|}{\binom{n-1}{j-1}} \leqslant 1,$$

что и требовалось доказать.

Пусть теперь $\mathscr{F}$ такое семейство Шпернера, что объединение любых k множеств, принадлежащих $\mathscr{F}$, отлично от X и пусть при этом его мощность $|\mathscr{F}|$ является наибольшей из возможных.

Пусть $\mathscr{G}$ — подсемейство, состоящее из множеств, принадлежащих $\mathscr{F}$ и имеющих мощность, меньшую n/k. Поскольку $\mathscr{G}$ — семейство Шпернера, оно удовлетворяет неравенству Любеля [3])

$$\sum_{j=1}^{n} \frac{|G_j|}{\binom{n}{j}} = \sum_{j=1}^{\left[\frac{n-1}{k}\right]} \frac{|G_j|}{\binom{n}{j}} \leqslant 1.$$

Следовательно,

$$|G| \leqslant \binom{n}{\left[\frac{n-1}{k}\right]}. \tag{2}$$

---

[1]) Так как в силу леммы $|\mathscr{C}_s| \leqslant \binom{n-1}{s-1}$ для всех $s = p+1, \ldots, r$. — *Прим. перев.*

[2]) Здесь используется тот факт, что множества семейства $\mathscr{A}$ несравнимы, и поэтому $|\mathscr{B}_r| = |\mathscr{C}_r| + |\mathscr{A}_r|$. — *Прим. перев.*

[3]) См. [9], стр. 11—12. — *Прим. перев.*

Из (1) и (2) следует, что

$$|\mathcal{F}| \leqslant \binom{n-1}{\left[\frac{n-1}{2}\right]} + \binom{n}{\left[\frac{n-1}{k}\right]}, \tag{3}$$

или, более точно, обозначая $|\mathcal{F}_j| \Big/ \binom{n-1}{j}$ через r_j,

$$|\mathcal{F}| \leqslant \binom{n}{\left[\frac{n-1}{k}\right]} + \sum_{j=\left[\frac{n+k-1}{k}\right]}^{n} r_j \binom{n-1}{j}, \tag{4}$$

где в силу (1)

$$\sum_{j=\left[\frac{n+k-1}{k}\right]}^{n} r_j \leqslant 1. \tag{5}$$

Неравенство (3) не дает желаемого результата, но оно верно для всех значений n.

Докажем теперь, что при $2t \geqslant n$ подсемейство $\mathcal{F}_t$ пусто.

Пусть T — наибольший индекс, для которого $\mathcal{F}_T$ непусто. Если $2T > n$ или если $2T = n$ и при этом $|\mathcal{F}_T| \neq \binom{n-1}{T}$, то из (*) следует, что семейство $(\mathcal{F} \setminus \mathcal{F}_T) \cup \mathcal{F}'_T$ имеет мощность, бóльшую $|\mathcal{F}|$, и удовлетворяет условию теоремы, что противоречит максимальности $\mathcal{F}$. Остается рассмотреть лишь случай, когда $\mathcal{F}_{n/2}$ состоит из всевозможных $(n/2)$-элементных подмножеств множества X, которые не содержат фиксированного элемента x из X; иначе не могло бы быть равенства в соотношении (*)[1]). В этом случае семейство $\mathcal{F}$ не может содержать никаких других множеств. Действительно, с одной стороны, если бы для некоторого $F \in \mathcal{F}$ оказалось, что $x \in F$, то нашлись бы два $(n/2)$-элементных подмножества G и H, не содержащих x и таких, что

$$G \cup H = X \setminus x,$$

и тогда $G \cup H \cup F = X$, что противоречит нашим предположениям. С другой стороны, если $x \notin F$, то F содержит некоторое $(n/2)$-элементное подмножество множества $X \setminus x$ или содержится в таком множестве, а это невозможно в силу свойства Шпернера. Таким образом, имеем $\mathcal{F} = \mathcal{F}_{n/2}$, а это означает, что

$$|\mathcal{F}| = \binom{n-1}{n/2},$$

[1]) То есть снова $(\mathcal{F} \setminus \mathcal{F}_T) \cup \mathcal{F}'_T$ имело бы мощность, большую $|\mathcal{F}|$, так как $|\mathcal{F}'_T| > |\mathcal{F}_T| \binom{n-1}{T-1} \Big/ \binom{n-1}{T} = |\mathcal{F}_T|$; равенство в (*) возможно лишь в единственном случае. — *Прим. перев.*

в то время как семейство, состоящее из всех $(n/2-1)$-элементных подмножеств множества $X \setminus x$ и множества $\{x\}$, также удовлетворяет условиям теоремы и имеет мощность, большую $|\mathcal{F}|$. Мы получили противоречие.

Итак, мы можем принять, что $\mathcal{F}_t = \varnothing$ при $2t \geqslant n$.

3. ДОКАЗАТЕЛЬСТВО ОСНОВНОЙ ТЕОРЕМЫ

Переформулируем теорему в следующей более сильной форме.

Теорема 2. *Пусть X — конечное множество мощности $2t + \varepsilon = n$, где ε равно 0 или 1. Пусть $\mathcal{F}$ — семейство Шпернера, состоящее из подмножеств множества X. Пусть, далее, для любых $F, G, H \in \mathcal{F}$ выполняется соотношение*

$$F \cup G \cup H \neq X$$

и величина $|F|$ максимальна при этих условиях. Тогда если $\varepsilon = 1$ и n достаточно велико, то существует элемент x из X, такой что $\mathcal{F}$ состоит в точности из всех t-элементных подмножеств множества $X \setminus x$. Если же $\varepsilon = 0$ и n достаточно велико, то существует элемент x из X, такой что $\mathcal{F}$ состоит в точности из всех $(t-1)$-элементных подмножеств множества $X \setminus x$ и множества $\{x\}$.

Доказательство. В силу (4) имеем

$$\begin{aligned}
|\mathcal{F}| &\leqslant r_{\left[\frac{n-1}{2}\right]}\binom{n-1}{\left[\frac{n-1}{2}\right]} + \\
&+ \left(1 - r_{\left[\frac{n-1}{2}\right]}\right)\binom{n-1}{\left[\frac{n-1}{2}\right]-1} + \binom{n}{\left[\frac{n-1}{3}\right]} = \\
&= \binom{n-1}{\left[\frac{n-1}{2}\right]} + \binom{n}{\left[\frac{n-1}{3}\right]} - \\
&- \left(1 - r_{\left[\frac{n-1}{2}\right]}\right)\left(\binom{n-1}{\left[\frac{n-1}{2}\right]} - \binom{n-1}{\left[\frac{n-1}{2}\right]-1}\right) \leqslant \\
&\leqslant \binom{n-1}{\left[\frac{n-1}{2}\right]} + \binom{n}{\left[\frac{n-1}{3}\right]} - \left(1 - r_{\left[\frac{n-1}{2}\right]}\right)\frac{2}{n+1}\binom{n-1}{\left[\frac{n-1}{2}\right]}. \qquad (6)
\end{aligned}$$

Оценим теперь отношение $\binom{n-1}{\left[\frac{n-1}{2}\right]}\Big/\binom{n}{\left[\frac{n-1}{3}\right]}$:

$$\frac{\binom{n-1}{\left[\frac{n-1}{2}\right]}}{\binom{n}{\left[\frac{n-1}{3}\right]}}=\frac{(n-1)!\left[\frac{n-1}{3}\right]!\left(n-\left[\frac{n-1}{3}\right]\right)!}{n!\left[\frac{n-1}{2}\right]!\left[\frac{n}{2}\right]!}=$$

$$=\left(\prod_{i=0}^{n-3-\left[\frac{n-1}{3}\right]-\left[\frac{n-1}{2}\right]}\frac{n-\left[\frac{n-1}{3}\right]-i}{\left[\frac{n-1}{2}\right]-1-i}\right)\left(\frac{\left[\frac{n}{2}\right]+1}{\left[\frac{n-1}{2}\right]}\right)\left(\frac{\left[\frac{n}{2}\right]+2}{n}\right)>$$

$$>\frac{1}{2}\prod_{i=0}^{n-3-\left[\frac{n-1}{3}\right]-\left[\frac{n-1}{2}\right]}\frac{\frac{2n}{3}-i-1}{\frac{n}{2}-i-\frac{3}{2}}>\frac{1}{2}\prod_{i=0}^{\frac{n}{6}-3}\frac{4}{3}\geqslant\frac{1}{5}\left(\frac{4}{3}\right)^{\frac{n}{6}}. \quad (7)$$

Рассмотрим отдельно случаи нечетного и четного n.

(а) n нечетно.

Если бы оказалось, что в $\mathcal{F}$ имеются два непересекающихся t-элементных подмножества множества X, то из этого немедленно следовало бы, что любое множество, входящее в $\mathcal{F}$, содержится в их объединении, которое представляет собой $2t$-элементное подмножество множества X. Следовательно, $\mathcal{F}$ оказалось бы семейством Шпернера, состоящим из всех подмножеств $2t$-элементного множества[1]), и утверждение теоремы было бы доказано.

Таким образом, можно считать, что любые два множества, принадлежащие подсемейству $\mathcal{F}_t$, имеют непустое пересечение, откуда в силу леммы имеем

$$|\mathcal{F}_t|\leqslant\binom{n-1}{\frac{n-1}{2}-1}=\binom{n-1}{\frac{n-1}{2}}-\frac{2}{n+1}\binom{n-1}{\frac{n-1}{2}}.$$

Используя (6), получаем для величины $|F|$ оценку

$$|\mathcal{F}|\leqslant\binom{2t}{t}+\binom{n}{\left[\frac{n-1}{3}\right]}-\frac{4}{(n+1)(n+1)}\binom{n-1}{\frac{n-1}{2}},$$

[1]) Так как максимальное семейство Шпернера может быть только таким (см. примечание на стр. 105). — *Прим. перев.*

которая в силу (7) при достаточно больших n (например, при $n \geqslant 300$) меньше, чем $\binom{2t}{t}$, что и требовалось доказать.

(б) n четно.

Вернемся к циклическим упорядочениям. Предположим, что для циклического упорядочения $x_1, \ldots, x_n, x_1$ существует в точности $t+1$ [1]) множеств, входящих в F_{t-1} и состоящих из последовательных элементов относительно этого циклического упорядочения. Образуем для каждого i пары $(t-1)$-элементных множеств

$$A_i = \{x_{i+1}, x_{i+2}, \ldots, x_{i+t-1}\}, \quad B_i = \{x_{i+t}, \ldots, x_{i+2t-2}\},$$

где $i = 1, \ldots, n$; x_j обозначает x_{j-n} при $n < j \leqslant 2n$. Каждое множество, состоящее из $t-1$ последовательных элементов относительно рассматриваемого упорядочения, содержится ровно в 2 парах. Существует всего $2t$ таких пар, а множеств, принадлежащих $\mathcal{F}_{t-1}$, имеется всего $t+1$. Следовательно, найдется пара, каждый член которой содержится в $\mathcal{F}_{t-1}$. Можно считать, что эта пара есть

$$A_n = \{x_1, \ldots, x_{t-1}\}, \quad B_n = \{x_t, \ldots, x_{2t-2}\}.$$

Никакое из множеств A_i при $t+1 \leqslant i \leqslant 2t-2$ не может принадлежать $\mathcal{F}_{t-1}$, так как в противном случае объединение множеств A_n, B_n и этого множества давало бы X. Таким образом, имеем $t-2$ множеств. Если A_t и A_{2t-1} одновременно принадлежат $\mathcal{F}_{t-1}$, то аналогичными рассуждениями устанавливается, что никакое из множеств $A_1, \ldots, A_{t-2}$ не принадлежит $\mathcal{F}_{t-1}$. Но это невозможно при $t \geqslant 4$, так как существует самое большее $t-1$ множеств A_i, которые не принадлежат $\mathcal{F}_{t-1}$. Так как нас не интересует случай $n < 8$, мы можем считать, что единственным недостающим множеством является или A_t, или A_{2t-1}. В обоих случаях приходим к тому, что существует элемент x_r из X, такой что множество A_j принадлежит $\mathcal{F}_{t-1}$ тогда и только тогда, когда оно не содержит x_r. Если $r_{t-1} \leqslant 1 - 1/n(t+1)$, то, используя (6) и (7) так же, как в случае нечетного n, приходим к тому, что это невозможно для достаточно больших n (например, при $n > 1000$).

Следовательно, можно считать, что

$$r_{t-1} > 1 - \frac{1}{n(t+1)}. \tag{8}$$

[1]) Из рассуждений, приведенных в доказательстве леммы, следует, что для любого циклического упорядочения число множеств из $\mathcal{F}_{t-1}$, состоящих из последовательных элементов относительно этого упорядочения, не превышает $n-(t-1) = t+1$. — *Прим. перев.*

Пусть c обозначает число циклических упорядочений, по отношению к которым существует ровно $t+1$ $(t-1)$-элементных множеств, принадлежащих $\mathscr{F}_{t-1}$ и состоящих из последовательных элементов по отношению к этому упорядочению. Тогда имеем

$$|\mathscr{F}_{t-1}| \leqslant \frac{c\,(t+1)+((n-1)!-c)\,t}{(t-1)!\,(t+1)!} = \binom{n-1}{t-1}\left(1-\frac{1}{t+1}\,\frac{(n-1)!-c}{(n-1)!}\right).$$

Используя (8), получаем

$$c > (n-1)!\,\frac{n-1}{n} > (n-2)\,(n-2)! \tag{9}$$

Если для некоторого циклического упорядочения существует элемент y из X, такой что любое множество, состоящее из $(t-1)$ последовательных элементов относительно этого упорядочения, принадлежит $\mathscr{F}_{t-1}$ тогда и только тогда, когда оно не содержит y, то мы будем говорить, что y *соответствует* этому циклическому упорядочению. В силу (9) можно считать, что существует циклическое упорядочение $x_1, \ldots, x_n, x_1$, которому соответствует некоторый элемент y из X. В силу симметрии можно считать, что $y = x_1$.

Если дано циклическое упорядочение $y_1, y_2, \ldots, y_{n-1}, y_1$ элементов множества $X \setminus x$, то, помещая x на все возможные $n-1$ мест, можно построить $n-1$ различных циклических упорядочений элементов множества X. Таким образом, можно разбить все циклические упорядочения элементов множества X на $(n-2)!$ непересекающихся $(n-1)$-элементных блоков. В силу (9) найдется такой блок, что каждому циклическому упорядочению из этого блока соответствует некоторый элемент[1]) y. Пусть $y_1, y_2, \ldots, y_{n-1}, y_1$ — циклическое упорядочение, из которого построены упорядочения, входящие в рассматриваемый блок. Допустим сначала, что некоторому циклическому упорядочению из этого блока соответствует элемент $u \neq x_1$. Пусть $u = y_j$. В данном упорядочении или y_j и y_{j+1}, или y_{j-1} и y_j являются последовательными элементами. В силу симметрии можно считать, что y_j и y_{j+1} — последовательные элементы, т. е. x_1 не лежит между ними. Тогда объединение $(t-1)$-элементного множества, оканчивающегося на соседний слева от y_j элемент, и $(t-1)$-элементного множества, начинающегося с соседнего справа от y_{j+1} элемента, есть $X \setminus \{y_j, y_{j+1}\}$. Так как оба этих множества принадлежат $\mathscr{F}_{t-1}$, никакое мно-

[1]) Вообще говоря, разным упорядочениям из этого блока могут соответствовать разные элементы. — *Прим. перев.*

жество из $\mathscr{F}$ не содержит одновременно y_j и y_{j+1}. Из этого следует, что любому упорядочению из рассматриваемого блока, в котором x_1 не лежит между y_j и y_{j+1}, соответствует или y_j, или y_{j+1}. Выберем один из элементов x_2, x_{t+1}, x_{2t}, который отличается как от y_j, так и от y_{j+1}, и обозначим этот элемент через z. Одна из трех пар $(\{x_3, x_4, \ldots, x_{t+1}\}, \{x_{t+2}, \ldots, x_{2t}\})$, $(\{x_2, \ldots, x_t\}, \{x_{t+2}, \ldots, x_{2t}\})$, $(\{x_2, \ldots, x_t\}, \{x_{t+1}, \ldots, x_{2t-1}\})$ дает множество $X \setminus \{x_1, z\}$ при объединении ее членов, которые принадлежат $\mathscr{F}_{t-1}$ [1]). Рассмотрим теперь циклическое упорядочение из рассматриваемого блока, в котором x_1 и z являются соседями. Так как ни x_1, ни z не соответствуют этому упорядочению, найдется множество, принадлежащее $\mathscr{F}_{t-1}$, которое содержит оба этих элемента. Таким образом, окажется, что нашлись 3 множества, принадлежащие семейству $\mathscr{F}$ и дающие при их объединении множество X, что приводит к противоречию.

Следовательно, можно считать, что x_1 является соответствующим элементом для каждого циклического упорядочения, входящего в рассматриваемый блок. Допустим, что существует множество из $\mathscr{F}$, содержащее элемент x_1 и некоторый другой элемент z. Пусть $z = y_j$. Так как x_1 соответствует циклическому упорядочению $y_1, y_2, \ldots, y_j, x_1, y_{j+1}, \ldots, y_{n+1}, y_1$, множества $\{y_{j+1}, \ldots, y_{j+t-1}\}$ и $\{y_{j+t}, \ldots, y_{j+2t-2}\}$, где индексы берутся по модулю $n-1 = 2t-1$, принадлежат F_{t-1}. Объединение этих двух множеств и множества, содержащего как z, так и x_1, есть X, что приводит к противоречию.

Пока что мы доказали, что любое множество, принадлежащее семейству $\mathscr{F}$, является либо подмножеством множества $X \setminus x_1$, либо $\{x_1\}$. Мы доказали также, что при $s \geqslant t$ $\mathscr{F}_s$ пусто. Множества, отличные от $\{x_1\}$ и принадлежащие $\mathscr{F}$, образуют семейство Шпернера подмножеств множества $X \setminus x_1$. Следовательно, в силу неравенства Любеля [4] имеем

$$\sum_{A \neq \{x_1\}} \frac{1}{\binom{n-1}{|A|}} \leqslant 1,$$

откуда

$$|\mathscr{F}| \leqslant 1 + \binom{n-1}{\left[\frac{n-1}{2}\right]},$$

и равенство может иметь место только тогда, когда $\mathscr{F} \setminus \{x_1\}$ состоит в точности из всех $(t-1)$-элементных подмножеств множества $X \setminus x_1$, что и требовалось доказать.

В заключение этой статьи упомянем две задачи.

[1]) В силу выбора элемента x_1 (см. стр. 114). — *Прим. перев.*

Задача 1. Пусть $\mathcal{F}$ — семейство, состоящее из s-элементных подмножеств конечного множества X, $|X| = n$. Предположим, что для любых множеств $F, G, H \in \mathcal{F}$ имеет место соотношение

$$|F \cap G \cap H| \geqslant 2.$$

Существует ли положительное ε, такое что при $s \leqslant (1/2 + \varepsilon)n$ выполняется неравенство $|\mathcal{F}| \leqslant \binom{n-2}{s-2}$?

Задача 2. Пусть $\mathcal{F}$ — семейство Шпернера, состоящее из подмножеств конечного множества X, $|X| = n$. Предположим, что объединение любых трех множеств из $\mathcal{F}$ имеет мощность, меньшую или равную $n-2$. Пусть ε — произвольное положительное действительное число. Верно ли, что при $n \geqslant n_0(\varepsilon)$ имеет место неравенство

$$|\mathcal{F}| \leqslant (1+\varepsilon)\binom{n-2}{\left[\frac{n-2}{2}\right]}?$$

Замечание. Если ответ на вопрос задачи 1 утвердителен, то также утвердителен ответ на вопрос задачи 2.

ЛИТЕРАТУРА

[1] Katona G. O. H. A simple proof of the Erdös — Chao — Ko — Rado theorem. — J. Combinatorial Theory V 13 (1972), 183—184.

[2] Katona G. O. H. A theorem of finite sets, Theory of Granphs Proc. Coll. held at Tihany, 1966. Akadémiai Kiadó, 1968, p. 187—207.

[3] Kruskal J. B. The number of simplicies in a complex. — In: Mathematical Optimization Techniques, Univ. of Calif. Press, Berkeley and Los Angeles, 1963, pp. 251—278.

[4] Lubell D. A short proof of Sperner's lemma, — J. Combinatorial Theory 1 (1966), 299.

[5] Milner E. C. A combinatorial theorem on systems of sets. — J. London Math. Soc. 43 (1968), 204—206.

[6] Sperner E. Ein Satz über Untermengen einer endlichen Menge. — Mat. Z. 27 (1928), 544—548.

[7]* Дейкин Д. Теорема Эрдеша — Ко — Радо из теоремы Краскала — Катоны, в сб. «Киберн. сборник», вып. 19 (н. с.). — М.: Мир, 1983, 168—169.

[8]* Дейкин Д. Простое доказательство теоремы Краскала — Катоны, в сб. «Киберн. сборник», вып. 19 (н. с.). — М.: Мир, 1983, 166—167.

[9]* Сачков В. Н. Введение в комбинаторные методы дискретной математики. — М.: Наука, 1982.

* Добавлено переводчиком.

Оценки памяти для одной игры на графах[1)]

В. Пауль[2)], *Р. Э. Тарьян*, *Дж. Р. Селони*[3)]

1. ВВЕДЕНИЕ

Пусть $G=(V, E)$ есть ациклический ориентированный граф со множеством вершин V и множеством ребер E. Если (i, j) является ребром G, то i мы называем *предшественником* j, а j — *последователем* i. Число предшественников вершины называется ее *входной степенью*, а число последователей — *выходной степенью*. Вершина с входной степенью, равной нулю, называется *источником*, а вершина с выходной степенью, равной нулю, — *стоком*[4)]. Через $\mathcal{G}(n, d)$ мы будем обозначать класс ациклических направленных графов с n вершинами, каждая из которых имеет входную степень, не превосходящую d. Буквы c, c_1, c_2, ... ниже обозначают положительные константы; $c(d)$, $c_1(d)$, $c_2(d)$ будут обозначать положительные константы, зависящие от d, но не от n. Наконец, через $[i, j]$ обозначается множество целых чисел $\{k \mid i \leqslant k \leqslant j\}$.

В этой статье мы изучаем игру одного лица на графах. Игра состоит в помещении камней в вершинах графа $G \in \mathcal{G}(n, d)$ в соответствии с некоторыми правилами. Каждый шаг игры состоит либо в помещении камня в пустую вершину v графа G (так называемое *заполнение* v'), либо в удалении камня с предварительно заполненной вершины. Вершину можно заполнить *только* если во всех ее предшественниках находятся камни. (В частности, всякий источник в любой момент можно заполнить.) Цель игры состоит в заполнении некоторой заданной вершины графа G так, чтобы в любой момент в вершинах графа находилось не более заданного числа камней. В дополнение к этому можно потребовать, чтобы заполнение происходило за наименьшее число шагов. Любую вершину графа G можно заполнить за n шагов, используя n камней, при заполнении вер-

---

[1)] Paul W., Tarjan R., Celoni J. Space bounds for a game on graphs, Math. Syst. Theory, 1977, 10, 239—251

[2)] Computer Science Department Cornell University; Ithaca, New York.

[3)] Computer Science Department Stanford University, Stanford, California.

[4)] В работах [3, 5, 9] используются термины вход (input) и выход (output). — *Прим. ред*

шин в топологическом порядке [1]) без удаления камней. Нас интересуют способы заполнения, при которых используется меньше n камней, но которые, возможно, требуют много больше чем n шагов.

Число камней, используемых в игре, моделирует объем памяти, требуемый для выполнения вычисления, в следующем смысле. Каждая вершина представляет значение некоторой величины. Это значение вычисляется применением специальной операции к значениям, представленным предшественниками этой вершины. Источники представляют входные величины. Каждый камень представляет некоторую ячейку памяти. Заполнение вершины соответствует вычислению значения, представленного этой вершиной, и запоминанию вычисленного значения в ячейке, представленной этим камнем. Удаление камня с вершины соответствует очищению ячейки памяти, представленной камнем; при этом значение, представленное вершиной, становится недоступным для дальнейшего вычисления. В том случае, когда это значение потребуется снова, его нужно вычислить вновь. Игру в камни можно рассматривать как модель для задач распределения регистров [7] и как средство изучения соотношений между границами времени и памяти для машин Тьюринга [1, 3].

Известны следующие результаты об игре в камни:

Теорема А. *Если* $G \in \mathcal{G}(n, d)$ *и максимальная выходная степень* G *равна единице* (*т. е.* G *дерево*), *то каждую вершину* G *можно заполнить за время* n, *используя* $c_2(d)\log n$ *камней* [1]. *Для всякого* n *существует граф* $G \in \mathcal{G}(n, 2)$ *с максимальной выходной степенью единица, требующий* $c_1 \log n$ *камней для заполнения некоторой вершины* [5].

Теорема В. [1] *Для каждого* n *существует граф* $G \in \mathcal{G}(n, 2)$, *требующий с* $\sqrt{n}$ *камней для заполнения некоторой вершины.*

Теорема С. [3] *Если* $G \in \mathcal{G}(n, d)$, *то всякую вершину графа* G *можно заполнить, используя* $c_3(d)\, n/\log n$ *камней.*

В разд. 2 для всякого n мы построим граф $G(n) \in \mathcal{G}(n, 2)$, такой что $G(n)$ требует $c_5 n/\log n$ камней для заполнения некоторой вершины. Это показывает, что граница в теореме С достигается с точностью до постоянного множителя. В разд. 3 мы дадим верхние границы для различных способов заполнения, включая способ, при котором достигается граница памяти $O(n/\log n)$ теоремы С. В разд. 4 будут приведены некоторые дальнейшие замечания.

[1]) То есть если $(ij) \in E$, то вершина i заполняется раньше, чем вершина j[4].

2. НИЖНЯЯ ОЦЕНКА

Мы докажем обещанную нижнюю оценку построением соответствующего семейства графов. Будет использован следующий результат Вэлианта [8]. Для каждого i существует граф $D(i)$ с $c_1 2^i$ ребрами, 2^i источниками и 2^i стоками, обладающий следующим свойством.

Пусть $j \in [1, 2^i]$; тогда для каждого подмножества S, состоящего из j источников, и каждого подмножества T, состоящего из j стоков, в графе $D(i)$ существует j не имеющих общих вершин путей, ведущих из S в T.

Вершины в этом графе могут иметь произвольную входную степень. Заменяя каждую вершину входной степени $d > 2$ бинарным деревом с d листьями, мы самое большее удваиваем число ребер. В новом графе каждая вершина имеет входную степень два и граф по-прежнему обладает тем же свойством. Так мы приходим к следующей лемме.

Лемма 1. *Для каждого значения i существует граф $C(i) \in \mathcal{G}(c2^i, 2)$ с 2^i источниками и 2^i стоками, такой что для каждого $j \in [1, 2^i]$, для любого подмножества S, состоящего из j источников, и любого подмножества T, состоящего из j стоков, в $C(i)$ найдутся без общих вершин j путей, ведущих из S в T.*

Следствие 1. *Для всякого $j \in [0, 2^{i-1}]$, если j камней помещены в любые j вершин графа $C(i)$, а T — любое подмножество, состоящее из не менее чем $j+1$ стоков, то не менее чем $2^i - j$ источников связаны с T путями, свободными от камней.*

Доказательство. Пусть $j \in [0, 2^i - 1]$. Предположим, что на $C(i)$ размещены j камней, а T есть множество, состоящее из не менее чем $j+1$ стоков. Любое подмножество S из $j+1$ источников связано с T $j+1$ путями, не имеющими общих вершин, причем по крайней мере один из этих путей свободен от камней. Поэтому число источников, не связанных с T путями, свободными от камней, не превосходит j. ∎

Используя копии $C(i)$, определим рекурсивно множество графов $\{G(i) \mid i = 8, 9, 10, \ldots\}$. $\mathcal{G}(8) = C(8)$. Образуем граф $G(i+1) = (V(i+1), E(i+1))$ из двух копий $G(i)$ и двух копий $C(i)$ следующим способом. Пусть граф $G(i) = (V(i), E(i))$ имеет множество источников $S(i) = \{s(i, j) \mid j \in [1, 2^i]\}$ и множество стоков $T(i) = \{t(i, j) \mid j \in [1, 2^i]\}$. Пусть $SC(i) = \{sc(i, j) \mid j \in [1, 2^i]\}$ суть источники, а $TC(i) = \{tc(i, j) \mid j \in [1, 2^i]\}$ суть стоки графа $C(i)$. Далее, пусть $G_1(i)$ и $G_2(i)$ суть две копии $G(i)$, а $C_1(i)$ и $C_2(i)$ — две копии $C(i)$. Пусть $S(i+1) = \{s(i+1, j) \mid j \in [1, 2^{i+1}]\}$ и $T(i+1) = \{t(i+1, j) \mid j \in [1, 2^{i+1}]\}$ суть два новых множества вершин. Теперь положим

$G(i+1)=(V(i+1), E(i+1))$, где

$V(i+1)=S(i+1)\cup T(i+1)\cup V_1(i)\cup V_2(i)\cup VC_1(i)\cup VC_2(i)$, а

$$\begin{aligned}E(i+1)=&E_1(i)\cup E_2(i)\cup EC_1(i)\cup EC_2(i)\\ &\cup\{(s(i+1, j), t(i+1, j)): j\in[1, 2^{i+1}]\}\\ &\cup\{(s(i+1, j), sc_1(i, j)): j\in[1, 2^i]\}\\ &\cup\{(s(i+1, j+2^i), sc_1(i, j)): j\in[1, 2^i]\}\\ &\cup\{(tc_1(i, j), s_1(i, j)): j\in[1, 2^i]\}\\ &\cup\{(t_1(i, j), s_2(i, j)): j\in[1, 2^i]\}\\ &\cup\{(t_2(i, j), sc_2(i, j)): j\in[1, 2^i]\}\\ &\cup\{(tc_2(i, j), t(i+1, j)): j\in[1, 2^i]\}\\ &\cup\{(tc_2(i, j), t(i+1, j+2^i)): j\in[1, 2^i]\}.\end{aligned}$$

Граф $G(i+1)$ изображен на рис. 1. Пусть $m(i)=|S(i)|=|T(i)|=2^i$, а $n(i)=|V(i)|$. Тогда $n(8)=c/2^8$, а $n(i+1)=2n(i)+(2c+4)2^i$, где c — константа из леммы 1. Индукцией по i докажем, что $n(i)=(i-7)c\cdot 2^i+(i-8)2^{i+1}$ следующим образом. Имеем $n(8)=c\cdot 2^8=1\cdot c\cdot 2^8+0\cdot 2^9$. Если $n(i)=(i-7)c2^i+(c\cdot 2^i-8)2^{i+1}$, то $n(i+1)=(i-7)c\cdot 2^{i+1}+c\cdot 2^{i+1}+(i-8)2^{i+2}+2^{i+2}=(i-6)c\cdot 2^{i+1}+(i-7)2^{i+2}$. Очевидно, что $G(i)\in\mathcal{G}(n(i), 2)$. Положим $c_1=14/256$, $c_2=3/256$, $c_3=34/256$, $c_4=1/256$. Очевидно, выполнены следующие неравенства: $c_3m(i)/2\geqslant c_2m(i+1)+1$, $(1-2c_2)\geqslant c_3$, $c_2m(i)\geqslant c_4m(i+1)+1$, $c_im(i)/2\geqslant c_2m(i+1)+1$, $1-c_2\geqslant c_1$; $c_1(c_3/2)-c_2\geqslant c_1$.

Лемма 2. *Пусть во время игры на графе $G(i)$ были заполнены в любом порядке*[1]) *не менее $c_1m(i)$ стоков, причем в исходной конфигурации камни были размещены не более чем в $c_2m(i)$ вершинах. Тогда найдется промежуток игрового времени $[t_1, t_2]$, в течение которого будут заполнены не менее чем $c_3m(i)$ источников и по крайней мере $c_4m(i)$ (не обязательно одних и тех же) камней всегда находятся на графе.*

Доказательство. Проводится индукцией по i. Пусть $i=8$. Рассмотрим исходную конфигурацию на $G(8)$ с не более чем тремя заполненными вершинами и предположим, что 14 стоков заполняются в промежутке времени $[0, t]$. Первоначально каждые четыре из этих стоков связаны посредством свободных от камней (далее — «свободных») путей с не менее чем 253 источниками (согласно следствию 1). Таким образом, по крайней

[1]) То есть каждый сток заполняется в некоторый момент времени, но не требуется, чтобы все стоки были заполнены одновременно.

мере один из этих стоков, скажем v, связан первоначально свободными путями с не менее чем 64 источниками. Когда сток v заполнен, то ни один из этих 64 источников не связан с v свободным от камней путем. Далее, множество источников, связанных с v свободным от камней путем, может убывать самое большее на один источник за каждый шаг времени. Пусть $t_1 - 1$ является последним моментом времени, когда 64 источника связаны с v путями, свободными от камней. В течение промежутка $[t_1, t]$ $63 \geqslant 34$ источника графа G (8) должны быть заполнены, причем на графе всегда находится по крайней мере 1 камень. Это доказывает лемму для $i = 8$. Предположим, что лемма верна для i. Чтобы доказать ее для $i+1$, рассмотрим исходную конфигурацию $G(i+1)$ с не более чем $c_2 m(i+1)$ заполненными вершинами и предположим, что самое меньшее $c_1 m(i+1)$ стоков заполняются в промежутке $[0, t]$. Мы рассмотрим несколько случаев.

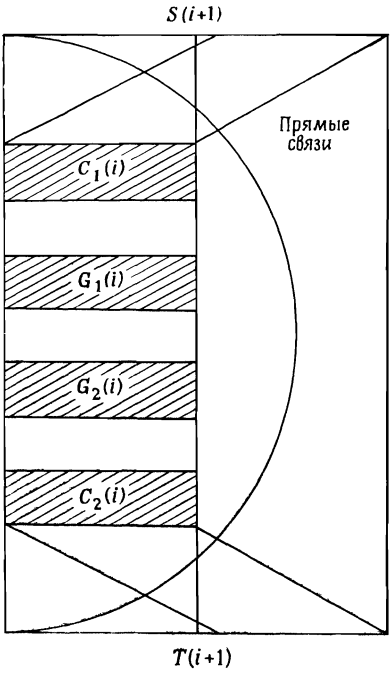


Рис. 1. $G(i+1)$.

Случай 1. Существует временной промежуток $[t_1, t_2] \subseteq [0, t]$, в течение которого заполняются самое меньшее $c_3 m(i)/2$ источников графа $G_1(i)$ и самое меньшее $c_2 m(i)$ камней всегда находятся на графе. Подграф графа $G(i+1)$, состоящий из всех вершин и ребер на путях из множества источников $\{s(i+1, j) \mid j \in [1, 2^i]\}$ к множеству стоков $\{s_1(i, j) \mid j \in [1, 2^i]\}$, а также подграф, состоящий из всех вершин и ребер на путях из множества $\{s(i+1, j+2^i) \mid j \in [1, 2^i]\}$ к множеству $\{s_1(i, j) \mid j \in [1, 2^i]\}$, удовлетворяют лемме 1 и следствию 1. Обозначим через t_0 последний момент до t_1, в который на графе находятся не более $c_2 m(i+1)$ камней. Ввиду того что $c_3 m(i)/2 \geqslant c_2 m(i+1) + 1$ (см. следствие 1) в этот момент найдется по крайней мере $2(m(i) - c_2 m(i+1)) = (1 - 2c_2) m(i+1) \geqslant c_3 m(i+1)$ источников графа $G(i+1)$, связанных свободными путями с $c_3 m(i)/2$ источниками графа $G_1(i)$, заполненными в период от t_1 до t_2. В про-

межутке $[t_6, t_2]$ по крайней мере эти источники графа $G(i+1)$ должны быть заполнены и по меньшей мере $c_2m(i) - 1 \geqslant c_4m(i+1)$ камней должны постоянно находиться на графе. Таким образом, для этого случая лемма справедлива.

Случай 2. Существует временной промежуток $[t_1, t_2] \subseteq [0, t]$, в течение которого по крайней мере $c_3m(i)/2$ источников графа $G_2(i)$ заполнены и $c_2m(i)$ камней всегда находятся на графе. Для этого случая лемма доказывается так же, как в случае 1.

Случай 3. Существует временной промежуток $[t_1, t_2] \subseteq [0, t]$, в течение которого по меньшей мере $c_1m(i+1)/2$ стоков графа $G(i+1)$ заполнены и по меньшей мере $c_2m(i)$ камней всегда находятся на графе. В течение $[t_1, t_2]$ либо заполняются $c_1m(i+1)/4$ стоков из множества $\{t(i+1, j) \mid j \in [1, 2^i]\}$, либо заполняются $c_1m(i+1)/4$ стоков из множества $\{t(i+1, j+2^i) \mid j \in [1, 2^i]\}$. В этом случае лемма доказывается аналогично случаю 1 с использованием неравенств $c_1m(i+1)/4 \geqslant c_2m(i+1)+1$, $1 - 2c_2 \geqslant c_3$ и $c_2m(i) - 1 \geqslant c_4m(i+1)$.

Случай 4. Не имеет места ни один из рассмотренных выше случаев. Поскольку не имеет места случай 3, должен найтись такой момент $t_1 \in [0, t]$, что меньше $c_1m(i+1)/2$ стоков графа $G(i+1)$ заполняются за время $[0, t_1]$, а число камней в момент t_1 на графе $G(i+1)$ не превосходит $c_2m(i)$. За время $[t_1, t]$ заполняются по меньшей мере $c_1m(i)$ стоков графа $G(i+1)$. Поскольку $c_1m(i)/2 \geqslant c_2m(i+1)+1 \geqslant c_2m(i)+1$, число стоков графа $G_2(i)$, связанных с упомянутыми стоками графа $G(i+1)$ свободными путями, не меньше $(1-c_2)m(i)$. Таким образом, по крайней мере $(1-c_2)m(i) \geqslant c_1m(i)$ стоков графа $G_2(i)$ заполняются за время $[t_4, t]$, начиная с исходной конфигурации, в которой в вершинах находятся не более $c_2m(i)$ камней. По индуктивному предположению найдется временной промежуток $[t_2, t_3] \subseteq [t_1, t]$, в течение которого заполняются $c_3m(i)$ источников графа $G_2(i)$ и $c_4m(i)$ камней все время присутствуют на графе $G_2(i)$.

Поскольку не имеет места случай 2, должен найтись момент $t_4 \in [t_2, t_3]$, такой что менее $c_3m(i)/2$ источников графа $G_2(i)$ заполняются за время $[t_2, t_4]$ и число камней на графе $G(i+1)$ в момент t_4 не превосходит $c_2m(i)$. За время $[t_4, t_3]$ заполняются по меньшей мере $c_3m(i)/2$ источников графа $G_2(i)$. В момент t_4 не меньше $c_1m(i)$ стоков графа $G_1(i)$ связаны свободными путями с этими источниками графа $G_2(i)$. На протяжении $[t_4, t_3]$ эти стоки графа $G_1(i)$ должны быть заполнены, начиная с не более чем $c_2m(i)$ заполненных вершин. По индуктивному предположению найдется промежуток $[t_5, t_6] \subseteq [t_4, t_3]$, в течение ко-

торого заполняются $c_3 m(i)$ источников графа $G_1(i)$ и $c_4 m(i)$ камней все время находятся на $G_1(i)$.

Поскольку не имеет места случай 1, должен найтись момент $t_7 \in [t_5, t_6]$, такой что меньше чем $c_3 m(i)/2$ источников графа $G_2(i)$ заполняются в течение $[t_5, t_7]$ и число камней на графе $G(i+1)$ в момент t_7 не превышает $c_2 m(i)$. В течение $[t_7, t_6]$ заполняются по меньшей мере $c_3 m(i)/2$ источников $G_1(i)$. В момент t_7 не меньше чем $(1-2c_2)m(i+1) \geqslant c_3 m(i+1)$ источников $G(i+1)$ связаны свободными путями с этими источниками графа $G_1(i)$. Таким образом, в течение $[t_7, t_6] \subseteq [t_5, t_6] \subseteq [t_2, t_3]$ заполняются по крайней мере $c_3 m(i+1)$ источников графа $G(i+1)$ и не менее $c_4 m(i) + c_4 m(i) = c_4 m(i+1)$ камней все время находятся на графе. Доказательство завершено. ∎

Теорема 1. *Для каждого $n \geqslant 2$ найдется граф $G \in \mathcal{G}(n, 2)$, такой что заполнение некоторой вершины G требует по меньшей мере $c_5 n/\log n$ камней.*

Доказательство. Для $n \geqslant c2^8$ через j обозначим максимальное число, такое что $n(j) \leqslant n$ и обозначим через G граф, образованный из графа $G(i)$ добавлением $n - n(j)$ дополнительных вершин без добавления дополнительных ребер. Поскольку заполнение всех стоков графа $G(j)$ из начальной конфигурации без камней в вершинах требует $c_4 m(j)$ камней, должен найтись некоторый сток, заполнение которого требует $c_4 m(j)$ камней. (В противном случае процедура последовательного заполнения стоков с использованием минимального числа камней для каждого стока и удалением всех камней после того, как очередной сток окажется заполненным, заполняла бы все стоки с применением меньше чем $c_4 m(j)$ камней.)

Поскольку $n(i+1) \leqslant 4n(i)$, для всех i должно выполняться неравенство $n/4 \leqslant n(j) \geqslant n$, т. е. $n/4 \leqslant (j-7)c2^j + (j-8)2^{j+1} \leqslant n$. Таким образом, $\log n \geqslant j$ и число требуемых камней не меньше $m(j) = 2^j \geqslant c_5 n/\log n$ при некоторой постоянной c_5. Выбрав $c_5 \leqslant 1/c \cdot 2^8$, можно установить истинность теоремы для $2 \leqslant n < c2^8$, а также для $n \geqslant c2^8$, поскольку для *всякого* графа требуется по меньшей мере один камень.

3. ВЕРХНИЕ ГРАНИЦЫ

В этом разделе мы выведем верхние границы для числа камней, необходимых при различных способах заполнения. Пусть *удаление* (**set** S) обозначает процедуру, удаляющую все камни из вершин S. Большинство наших результатов зависит от следующего алгоритма, который заполняет вершины по методу «из глубины наружу». (Здесь и ниже $B(v)$ обозначает множество

предшественников вершины v, т. е. вершин, из которых имеется путь в v. — *Ред.*).

procedure *заполнение из глубины наружу* (**graph** G, **vertex** v, **set** S);
  begin
    for $u \in B(v)$ **do if** u не заполнена **then**
      заполнение из глубины наружу $(G, u, B(v) \cup S)$;
    заполнить v;
    удаление $(V - (S \cup \{v\}))$
  end *заполнение из глубины наружу*;

Следующая лемма неявно содержится в [3].

Лемма 3. *Пусть* $G = (V, E) \in \mathcal{G}(n, d)$, v — *вершина* G. *Если каждый путь в вершину* v *содержит не более* l *вершин, то процедура заполнения «из глубины наружу»* $(G, v, \varnothing)$ *заполняет* v, *используя не больше* $(d-1)(l-1)+2$ *камней.*

Доказательство. Проводится индукцией по числу вершин l в длиннейшем пути к v. Если вершина v является источником, то процедура требует $1 \leqslant (d-1) \cdot 0 + 2$ камней. Предположим, что лемма верна для l и пусть длиннейший путь к v содержит не больше $l+1$ вершин. Тогда наша процедура использует $\max\{(d-1)+(d-1)(l-1)+2,\ d+1\} = (d-1)l+2$ камней. В следующей более общей процедуре используются «постоянные» камни, которые, будучи однажды помещены в вершинах множества P, никогда не удаляются из них.

procedure *постоянное заполнение*[1]) (**graph** G, **vertex** v, **set** P)
  begin
    for $u \in P$ в топологическом порядке **do** *заполнение из глубины наружу* (G, u, P);
    заполнение из глубины наружу (G, v, P)
  end *постоянное заполнение*;

Лемма 4. *Если* $|P| = k$ *и* $G = (V, E) \in \mathcal{G}(n, d)$ *обладает тем свойством, что каждый путь в вершину* v, *не содержащий вершин из* P, *содержит не более* l *вершин, то постоянное заполнение* (G, v, P) *требует не более* $k + (d-1)(l-1)+2$ *камней.*

Доказательство. Когда процедура «*постоянное заполнение*» производит вызов «*заполнение из глубины наружу* (G, u, P)», где $u \in P \cup \{v\}$, каждый свободный от камней путь в вершину u содержит не более l вершин, поскольку все вершины из P на любом пути в u уже заполнены. Таким образом, оценка следует из леммы 3.

[1]) Более точным названием этой процедуры является «заполнение с постоянными камнями», но для краткости мы пишем «постоянное заполнение». — *Прим. перев.*

Эрдеш, Грехем и Семереди [2] доказали, что в любом ациклическом ориентированном графе с n вершинами найдется подмножество P из $c_1 n \log\log n/\log n$ вершин, такое что каждый путь, не содержащий вершин из P, имеет длину, не превосходящую $c_2 n \log\log n/\log n$. (Более того, они указали простой способ нахождения такого множества P.) Этот результат вместе с леммой 4 приводит к следующей теореме.

Теорема 2. *Если* $G=(V,E)\in\mathcal{G}(n,d)$ *и* P *выбрано, как указано выше, то постоянное заполнение* (G,v,P) *использует не больше* $cdn\log\log n/\log n$ *камней.*

Чтобы ближе подойти к оценке теоремы С, нам потребуется алгоритм, несколько более сложный, чем *постоянное заполнение*. Обсуждение этого алгоритма мы отложим до конца раздела.

Теорема 1 и лемма 4 дают также

Следствие 2 [2]. *Для бесконечно многих* n *существует граф* $G\in\mathcal{G}(n,2)$, *такой что в каждом подмножестве* P *вершин графа* G *со свойством «каждый путь, не проходящий через* P, *имеет длину не больше, чем* $|P|$» *содержится не меньше* $c_1 n/\log n$ *вершин.*

Теперь мы приведем хорошие способы заполнения двух специальных классов графов. Назовем граф $G=(V,E)\in\mathcal{G}(n,d)$ *уровневым графом*, если можно разбить на *уровни* $L(1)$, $L(2)$, ..., $L(m)$ так, что из $(v,w)\in E$ и $v\in L(i)$ следует $w\in L(i+1)$. Пусть G — уровневый граф, а x — положительное вещественное число. Назовем уровень i *большим*, если $|L(i)|\geqslant x$, и *малым* в противном случае. Пусть $\{i(j)\mid 1\leqslant j\leqslant l\}$ — множество индексов малых уровней, взятых в возрастающем порядке. Пусть $i(0)=0$, $i(l+1)=m+1$, и $L(0)=\varnothing$. Пусть также $v\in L(j)$ — любая вершина, а l' — такое целое число, что $i(l')>j$ и $i(l'+1)\geqslant j$.

Следующий алгоритм эффективно заполняет v.

```
procedure уровневое заполнение (graph G, set array L, array j,
vertex v, integer l');
  begin
    for j := 1 until l' do
      begin
        for u ∈ L(i(j)) do
          заполнение из глубины наружу (G, u, L(i(j − 1)) ∪
          ∪ L(i(j)));
        удаление (L(i(j − 1)))
      end
    заполнение из глубины наружу (G, v, L(i(l')))
  end уровневое заполнение;
```

Лемма 5. *Пусть* $G=(V,E)\in\mathcal{G}(n,d)$ — *уровневый граф,* x — *произвольное натуральное число, не превышающее количество уровней в* G. *Тогда процедура «уровневое заполнение» заполняет любую вершину* G, *используя не более* $2x+(d-1)n/x$ *камней.*

Доказательство. На протяжении всего процесса заполнения не больше двух малых уровней содержат камни одновременно. Следовательно, не больше $2x-2$ камней все время находятся на малых уровнях. Число уровней, расположенных между двумя малыми уровнями, не превосходит n/x, так как имеется не больше n/x больших уровней. Таким образом, число камней, используемых в последнем вызове заполнения «из глубины наружу», не превосходит $(d-1)n/x+2$, а общее число используемых камней не больше $2x+(d-1)n/x$.

Теорема 3. *Если* $G\in\mathcal{G}(n,d)$ — *уровневый граф, то любая вершина* G *может быть заполнена с* $\sqrt{8(d-1)n}$ *камнями.*

Доказательство. Немедленно получается из леммы 5, если взять $x=\sqrt{(d-1)n/2}$. ∎

Граница в теореме 3 является неулучшаемой (с точностью до постоянного множителя, зависящего от d), так как графы Кука, использованные в доказательстве теоремы B, являются уровневыми.

Класс *m-головочных графов* $\mathcal{H}(n,m)$ является подмножеством $\mathcal{G}(n,m+1)$, содержащим графы $G=(V,E)$ следующего типа:

$V=\{(i,\ j_1(i),\ \ldots,\ j_m(i))\mid 1\leqslant i\leqslant n,\ \ j_k(1)=1$ для всех k и $j_k(i+1)\in\{j_k(i)-1,\ j_k(i),\ j_k(i)+1\}$ для всех $i<n$ и всех $k\}$;

$E=\bigcup_{k=1}^{m}\{((i',\ j_1(i'),\ \ldots,\ j_m(i')),\ (i,\ j_1(i),\ \ldots,\ j_m(i))\mid i'=\max\{l<i\mid j_k(l)=j_k(i)\}\}\cup$

$\cup\{((i,\ j_1(i),\ \ldots,\ j_m(i)),\ (i+1,\ j_1(i+1),\ \ldots,\ j_m(i+1))\mid 1\leqslant i<n\}$.

Эти графы названы m-головочными потому, что их можно использовать для представления движения головок, имеющего место в процессе вычисления на машине Тьюринга. Игра в камни на m-головочных графах применялась в [1, 3] как средство установления соотношения между временем и памятью при вычислении на машинах Тьюринга. Мы покажем, что каждый граф $\mathcal{H}(n,1)$ можно заполнить с $O(\sqrt{n})$ камнями, приспособив для этого доказательство Патерсона об эффективном по объему памяти моделировании одноленточных машин Тьюринга [6].

Пусть $G=(V,E)\in\mathcal{H}(n,1)$. Для каждого j положим $H(j)=\{(i,j_1(i))\in V\mid j_1(i)=j\}$. Для всякого $S\subseteq V$ положим *ширина* $(S)=\max\{|j_1(i)-j_1(i')|\ (i,j_1(i)),\ \ (i',j_1(i'))\in S\}$. Для каждого $S\subseteq V$ должно найтись некоторое j, такое что $\max\{|j_1(i)-j|\mid(i,j_1(i))\in S\}\leqslant(2/3)\cdot$*ширина*$(S)+1$ и $|H(j)|\leqslant 3n/$*ширина* (S). (Если бы такого j не существовало, то число вершин

в G было бы не меньше, чем $((1/3)\cdot$ *ширина* $(S)+1)\cdot(3n/$*ширина* $(S))>n$.) Удаление вершин в $H(j)$ расщепляет S на две части: $S_1=\{(i, j_1(i))\in S \mid j_1(i)<j\}$ и $S_2=\{(i, j_1(i))\in S \mid j_1(i)>j\}$. Всякий путь в G содержит также вершину из S_1 и вершину из S_2; помимо этого, он должен содержать промежуточную вершину из $H(j)$.

Следующий рекурсивный алгоритм эффективно заполняет вершину v в $G=(V, E)\in\mathscr{H}(n, 1)$. Параметр x является положительным вещественным числом, значение которого мы выберем позднее.

procedure *одноголовочное заполнение* (**graph** G, **set** S, **vertex** v)
  if *ширина* $(S)<x$ **then**
    begin
      for $(i, j_1(i))\in S$ в топологическом поряке **while** $i\leqslant v$ **do**
        begin
          заполнить $(i, j_1(i))$;
          положить $(i', j_1(i'))$ равным вершине в S (если такая найдется) с наибольшим $i'<i$, для которого $j_1(i')=j_1(i)$;
          удалить камень из $(i', j_1(i'))$;
        end;
      удаление $(S-\{v\})$
    end
  else
    begin
      set S_1, S_2;
      найти j, такое что $\max\{|j_1(i)-j|:\ (i, j_1(i))\in S\}\leqslant$
        $\leqslant\frac{2}{3}\cdot$ *ширина* $(S)+1$ и $|H(j)|\leqslant 3n/$*ширина* (S);
      $S_1:=\{(i, j_1(i))\in S:\ j_1(i)<j\}$;
      $S_2:=\{(i, j_1(i))\in S:\ j_1(i)>j\}$;
      for $(i, j_1(i))\in H(j)$ в топологическом порядке **do**
        begin
          if $(i-1, j_1(i-1))\in S_1$ **then**
          одноголовочное заполнение $(G, S_1, (i-1, j_1(i-1)))$
          else if $(i-1, j_1(i-1))\in S_2$ **then**
          одноголовочное заполнение $(G, S_2, (i-1, j_1(i-1)))$;
          заполнить $(i, j_1(i))$
        end;
      if $v\in S_1$ **then** *одноголовочное заполнение* (G, S_1, v)
        else if $v\in S_2$ **then** *одноголовочное заполнение* (G, S_2, v);
      end
  удаление $(S\setminus\{v\})$
  end *одноголовочное заполнение*;

Лемма 6. *Процедура «одноголовочное заполнение» заполняет каждую вершину графа* $G=(V,E)\in\mathscr{H}(n,1)$, *используя* $9n/x+x$ *камней* (x — *произвольное положительное число — параметр алгоритма*).

Доказательство. Пусть $p(n,y)$ обозначает число камней, используемых при выполнении *одноголовочного заполнения* (G,S,n), когда $G=(V,E)\in\mathscr{H}(n,1)$ и для $S\subseteq V$ *ширина* $(S)=y$. Тогда $p(n,y)\leqslant x$, если $y<x$ и $p(n,y)\leqslant p(n,\lfloor(2/3)y\rfloor)+3n/y$, если $y\geqslant x$. Пусть y таково, что $y<x$, $(3/2)\,y\geqslant x$. Тогда

$$p(n,(3/2)^l y)\leqslant\sum_{i=1}^{l}\frac{3n}{(3/2)^i y}+x\leqslant\frac{3n}{(3/2)\,y}\sum_{i=0}^{l-1}(2/3)^i+x\leqslant\frac{9n}{x}+x$$

для каждого положительного x. Максимальное число камней, требуемых для заполнения каждого графа из $\mathscr{H}(n,1)$, не превосходит $p(n,n)\leqslant 9n/x+x$. ∎

Теорема 4. *Если* $G\in\mathscr{H}(n,1)$, *то каждую вершину графа* G *можно заполнить с использованием* $6\sqrt{n}$ *камней.*

Доказательство. Немедленно следует из леммы 6, если выбрать $x=3\sqrt{n}$. ∎

Графы Кука вкладываются в одноголовочные графы с умножением числа вершин лишь на постоянный множитель. Таким образом, граница в теореме 4 является неулучшаемой с точностью до постоянного множителя. Приспособив построение из разд. 2, можно показать, что двухголовочные графы требуют $cn/(\log n)^2$ камней в худшем случае, и мы уверены, хотя и не можем доказать, что эту нижнюю границу можно поднять до $cn/\log n$.

В [6] Патерсон показывает, что при $t(n)\geqslant n^2$ каждую $t(n)$-ограниченную по времени недетерминированную одноленточную машину Тьюринга можно промоделировать на $\sqrt{t(n)}$ ленточно-ограниченной детерминированной машине Тьюринга. Могут спросить, приводит ли комбинация теоремы 4 с результатами работы [3] об эффективном по объему памяти моделировании машин Тьюринга к результату Патерсона. Мы можем привести $t(n)^{2/3}$-ленточно-ограниченное моделирование, но не видим очевидного способа получить границу Патерсона на основе нашего результата.

Последним результатом этого раздела является алгоритм, основанный на доказательстве в [3] теоремы C, который эффективно заполняет каждую вершину произвольного графа. Этот алгоритм является рекурсивным и работает на графе G следующим образом.

Если граф G мал, то вершины G заполняются в топологическом порядке без удаления камней. Если граф G большой, то он расщепляется на два графа G_1 и G_2, таких что ни одно ребро не ведет из G_2 в G_1, причем число ребер, ведущих в G_2 как из G_1, так и из G_2, находится между $l/2-d$ и $l/2+d$, где l есть число ребер в G. Это разбиение можно осуществить нумерацией вершин в топологическом порядке, помещением всех вершин первоначально в G_1 и добавлением вершин к G_2 одну за другой в обратном топологическом порядке до тех пор, пока у G_2 не окажется достаточно входящих ребер (с каждой вершиной, добавляемой к G_2, добавляется от нуля до d входящих ребер).

После того как граф G будет расщеплен, применяется следующий метод заполнения вершин G. Если некоторая вершина в G_1 должна быть заполнена, то этот метод рекурсивно применяется к G_1. Если должна быть заполнена некоторая вершина в G_2 и число ребер, ведущих из G_1 в G_2, мало (т. е. меньше $l/\log l$), то мы рассматриваем множество P, состоящее из всех вершин графа G_1, имеющих последователей в G_2, рекурсивно применяем к (G, P) процедуру *заполнения с постоянными камнями*, а затем удаляем все камни, кроме постоянных (т. е. камней на вершинах из P). После этого нужная вершина в G_2 заполняется рекурсивным применением описываемого метода к G_2. Если должна быть заполнена некоторая вершина в G_2 и число ребер из G_1 в G_2 велико, алгоритм рекурсивно применяется к G_2. Каждый раз, когда следующая вершина v, которая должна заполняться в G_2, имеет несколько предшественников $u_1, \ldots, u_k$ в G_1, алгоритм рекурсивно применяется к G_1 с заполнением $u_1, \ldots, u_k$ и все камни в G_1, кроме камней на $u_1, \ldots, u_k$, удаляются. После заполнения вершины v все камни с G_1 удаляются и алгоритм продолжает работу с G_2. Более точно этот алгоритм приводится ниже. Параметр $\mathscr{S}$ является разбиением множества вершин V графа G, производимым при вложенных рекурсивных вызовах этой процедуры. Множество T есть множество вершин, которые, будучи однажды заполнены, не должны освобождаться от камней на протяжении текущего рекурсивного вызова процедуры *наилучшее заполнение*. Целое число k является подходящим положительным значением, зависящим от d; мы определим k позднее. В результате вызова процедуры «*наилучшее заполнение*» $(G, \{V\}, v, \varnothing)$ заполняется вершина v графа $G=(V, E)\in\mathscr{G}(n, d)$.

```
procedure наилучшее заполнение (graph G, partition 𝒮,
vertex v, set T);
  begin
    взять S ∈ 𝒮, такое что v ∈ S;
    l := |{(u, w): u, w ∈ S}|;
```

```
if l < k then
  begin
    for u ∈ B(v) do
    if u не заполнено
    then наилучшее заполнение (G, S, u, T ∪ B(v));
    заполнить v;
    удаление (V − (T ∪ {v}))
  end
else
  begin
    разделить S на такие части S1, S2, что
        1) ((u, w) ∈ E & u ∈ S2) ⊃ w ∈ S2
        2) l/2 − d ≤ |{(u, w): w ∈ S2}| ≤ l/2 + d;
    if |{(u, w): u ∈ S1, w ∈ S2}| ≤ l/log l then
      begin
        set C;
        C := {u | ∃(u, w) ∈ E с u ∈ S1, w ∈ S2};
        for u ∈ C do
          if u не заполнено
          then наилучшее заполнение (G, 𝒮 − {S} ∪ {S1, S2},
          u, T ∪ C);
          наилучшее заполнение (G, 𝒮 − {S} ∪ {S1, S2},
          v, T ∪ C);
          удаление (V − (T ∪ {v}))
      end
    else наилучшее заполнение (G, 𝒮 − {S} ∪ {S1, S2}, v, T)
  end
end наилучшее заполнение;
```

Теорема 5. *Процедура наилучшего заполнения заполняет каждую вершину графа* $G = (V, E) \in \mathcal{G}(n, d)$, *используя* $c_6(d \log d)(n/\log n)$ *камней.*

Доказательство. Пусть $q(m)$ обозначает максимальное число камней, используемых процедурой *наилучшее заполнение*, чтобы заполнить любую вершину любого графа с m или меньшим числом ребер и максимальной входной степенью d. Тогда

$$q(m) \leqslant m, \quad \text{если} \quad m \leqslant k \quad \text{и}$$

$$q(m) \leqslant \max\left\{ q\left(\frac{m}{2} + d\right) + \frac{m}{\log m},\ 2q\left(\frac{m}{2} + d - \frac{m}{\log m}\right)\right\},$$

$$\text{если} \quad m > k.$$

Придадим k минимальное значение из таких, что $\log k \geqslant 10$ и $k/(16 \log k) \geqslant d$. (Так, k есть $O(d \log d)$.) Индукцией мы покажем, что $q(m) \leqslant cm/\log m$ для всех $m \geqslant 2$, где $c = \log k$. Для

$m \leqslant k$ этот результат справедлив, поскольку $q(m) \leqslant m \leqslant cm/\log k \leqslant cm/\log m$. Пусть $m > k$. Предположим, что $q(m') \leqslant cm'/\log m'$ для всех $m' < m$. Тогда

$$q\left(\frac{m}{2}+d\right)+\frac{m}{\log m} \leqslant c\left(\frac{m}{2}+d\right)/\log\left(\frac{m}{2}+d\right)+$$

$$+\frac{m}{\log m} \leqslant c\left(\frac{9}{16}m\right)/(\log m-1)+\frac{m}{\log m}$$

$$\left(\text{поскольку из } m > k \text{ следует } \frac{m}{16} > d\right)$$

$$\leqslant \frac{cm\left(\frac{9}{16}+\frac{1}{m}\right)}{(\log m-1)} \leqslant \frac{5}{9}\,cm/(\log m-1) \quad (\text{так как } m > k \geqslant 16)$$

$$\leqslant cm/\log m \left(\text{так как } \frac{\log m}{\log m-1} \leqslant \frac{10}{9} \leqslant \frac{8}{5}\right).$$

Кроме того,

$$2q\left(\frac{m}{2}+d-\frac{m}{\log m}\right) \leqslant$$

$$\leqslant 2c\left(\frac{m}{2}+d-\frac{m}{\log m}\right)/\log\left(\frac{m}{2}+d-\frac{m}{\log m}\right) \leqslant$$

$$\leqslant 2c\left(\frac{m}{2}+d-\frac{m}{\log m}\right)/\left(\log m-\frac{3}{2}\right),$$

так как $\log m \geqslant 10$ и

$$\log(2/5m) \geqslant \log m - 3/2$$

$$\leqslant 2c\left(\frac{m}{2}-\frac{3}{4}\frac{m}{\log m}\right)/\left(\log m-\frac{3}{2}\right),$$

так как $d \leqslant \frac{m}{4\log m}$

$$= \frac{cm}{\log m}.$$

Из этого по индукции следует, что $q(m) \leqslant cm/\log m$ для всех $m \geqslant 2$. Таким образом, для всех $m \geqslant 2$ $q(m) \leqslant c_6(\log d) \times \times (m/\log m)$ для некоторой положительной постоянной c_6. Так как $m < dn$, то $q(m) \leqslant c_6(\log d)\,(dn/\log dn) \leqslant c_6(d \log d) \times \times (n/\log n)$. ∎

4. ЗАМЕЧАНИЯ

Теорема 1 дает нижнюю границу $cn/\log n$ для числа камней, необходимых для заполнения каждого графа в $\mathcal{G}(n, 2)$. Из этого результата следует, что верхняя граница в теореме С неулучшаема с точностью до постоянного множителя. Этот результат показывает также, что эффективное по объему памяти модели-

рование многоленточных машин Тьюринга, данное в [3], нельзя улучшить без применения новых методов.

Много вопросов об игре в камни остается пока без ответа, и некоторые области применения ждут своих исследователей. Например, насколько большим временем приходится жертвовать, чтобы достичь данной экономии в камнях? Сколько камней можно сэкономить, если сохраняется полиноминальное время работы? Сколько времени можно сэкономить при сохранении границы числа камней $cn/\log n$?

Возможная область приложений лежит в выводе нижних границ для времени, необходимого при различных вычислениях. Например, предположим, что мы хотим доказать нижнюю границу $cn \log n$ объема булевой схемы, необходимого для некоторого вычисления. Если мы докажем, что всякая такая схема имеет объем $c_1 n \log n$ или требует одновременного запоминания $c_2 n$ промежуточных результатов, то доказываемая граница будет следовать из теоремы С.

Благодарность. За полезные и вдохновляющие обсуждения мы признательны профессорам В. Клаусу и К. Мельхорну.

ЛИТЕРАТУРА

[1] Cook S. A. An observation on time-storage trade off. — Proc. of the Fifth Annual ACM Symp. on Theory of Computing (1973), 29—33

[2] Erdös P., Graham R. L., Szemerédy E. Onsparse graphs with dense long paths. — STAN-CS-75-504, Computer Science Department, Stanford University (1975).

[3] Hopcroft J., Paul W., Valiant L. On time versus space and related problems. — Sixteenth Annual Symp. on Foundations of Computer Science (1975), 57—64.

[4] Knuth D. The Art of Computer Programming, Vol. I: Fundamental Algoritms. — Addison-Wesley, Reading, Mass. (1968), 258—265. [Имеется перевод: Кнут Д. Искусство программирования для ЭВМ. Том I: Основные алгоритмы. — М.: Мир, 1976.]

[5] Paterson M. S., Hewitt C. E. Comparative Schematology. — Record of Project MAC Conf. on Concurrent Systems and Parallel Computation (1970), 119—128.

[6] Paterson M. S. Tape bounds for time-bounded Turing machines. — J. of Comp. and Sys. Sci. 6 (1972), 116—124. [Имеется перевод: Патерсон М. С. Ограничение на ленту для ограниченных во времени машин Тьюринга. — Сб. «Сложность вычислений и алгоритмов», — М.: Мир, 1974, 213—221.]

[7] Sethi K. Complete register allocation problems. — Proc. of the Fifth Annual ACM Symp. on Theory of Computing (1973), 182—195.

[8] Valiant L. On non-linear lower bounds on computational complexity. — Proc. of the Seventh Annual ACM Symp. on Theory of Computing (1975), 45—53.

О соотношении времени и памяти в игре в камни[1)]

В. Пауль[2)], *Р. Э. Тарьян*[3)]

В различных контекстах изучается одна игра в камни на графах в качестве модели для соотношений между временем вычислений [1, 2, 3, 8] и объемом памяти. В этой заметке показано, что существует семейство ориентированных ациклических графов G_n и постоянных c_1, c_2, c_3, таких что
(1) G_n содержит n вершин и каждая вершина в G_n имеет входную степень не больше 2.
(2) Каждый граф G_n можно заполнить с $c_1\sqrt{n}$ камнями за n шагов.
(3) Каждый граф G_n можно также заполнить с $c_2\sqrt{n}$ камнями, $c_2 < c_1$, но каждая выигрышная стратегия при этом требует не меньше чем $2^{c\sqrt{n}}$ шагов игры.

Пусть $S(k, n)$ — множество всех ориентированных ациклических графов с n вершинами, каждая из которых имеет входную степень не больше k. На графах $G \in S(k, n)$ рассматривается следующая игра одного лица. Игра состоит в помещении камней в вершины графа G в соответствии со следующими правилами:

(I) во входную вершину (т. е. вершину без предшественников) можно всегда поместить камень;
(II) если все непосредственные предшественники вершины c заняты камнями, то в c можно поместить камень;
(III) всегда можно удалить камень из вершины.

Цель игры состоит в помещении камня в некоторую выходную вершину (без потомков) графа G таким образом, что общее число камней, одновременно находящихся в вершинах графа, было минимальным. Игра моделирует потребности вычислений во времени и памяти в следующем смысле. Вершины G соответствуют операциям, а камни — ячейкам памяти (storage locations). Если камень находится в вершине, это означает, что результат операции, которой соответствует вершина, запоминается в некоторой ячейке памяти. Таким образом, правила имеют следующий смысл:

[1)] Paul W., Tarjan R. Time — space trade — offs in a pebble game. Acta Informat., v. 10 (1978), 111—115.

[2)] Fakultät für Mathematik der Universität Bielefeld, Germany (Fed. Rep.).
[3)] Computer Science Department, Stanford University, Stanford, USA.

(I) входная информация всегда доступна;
(II) если все операции известны и размещены где-нибудь в памяти, то можно выполнить эту операцию, а результат поместить в новую ячейку в памяти;
(III) всякое место в памяти можно всегда очистить от содержимого. По этим правилам одна и та же вершина может быть много раз занята камнями. Это соответствует повторному вычислению промежуточных результатов.

В частности, эта игра используется для моделирования времени и памяти машин Тьюринга [1, 2], а также соотношения длины неветвящихся программ и необходимого при их работе объема памяти [8].

Приведем известные результаты об игре в камни.

(A) Каждый граф $G \in S(k, n)$ может быть заполнен[1]) с $c_k n/\log n$ камнями, где постоянная c_k зависит только от k [2].

(B) Существует постоянная c и семейство графов $G_n \in S(2, n)$, такие что для бесконечно многих n G_n нельзя заполнить с менее чем $cn/\log n$ камнями [4].

(О других результатах см. [1, 3, 4, 7, 8].)

Помещая камни в вершины графа G в топологическом порядке (т. е. если существует ребро из вершины c в вершину c', то сначала камень помещается в вершину c), можно заполнить каждый граф $G \in S(k, n)$ с n камнями и за n шагов. Однако известны стратегии, которые обходятся $O(n/\log n)$ камнями на каждом графе, но затрачивают экспоненциальное время. Таким образом, естественно спросить, существуют ли графы $G_n \in S(k, n)$, такие что каждая стратегия, на которой достигается минимум числа камней, необходимо требует экспоненциального времени. Это действительно имеет место.

Теорема. *Существует семейство графов* $G_n \in S(2, n)$, $n = 1, 2, \ldots$, *и положительные постоянные* c_1, c_2, c_3, $c_2 < c_1$, *такие что для бесконечно многих* n

(1) *граф* G_n *можно заполнить с* $c_1\sqrt{n}$ *камнями за* n *шагов*;
(2) *граф* G_n *можно заполнить и с* $c_2\sqrt{n}$ *камнями*;
(3) *каждая стратегия, заполняющая* G_n *только с* $c_2\sqrt{n}$ *камнями, требует по крайней мере* $2^{c_3\sqrt{n}}$ *шагов.*

Таким образом, экономия лишь фиксированной части всех камней приводит к увеличению времени игры от линейного до $2^{O(\sqrt{n})}$.

Доказательство. В качестве «строительных блоков» для графов G_n нам потребуются некоторые специальные графы. *Ориен-*

[1]) Речь идет о заполнении некоторой выделенной выходной вершины графа G. — *Прим. перев.*

тированным двудольным графом называется граф, все вершины которого можно так разбить на два непересекающихся множества N_1, N_2, что все ребра ведут из N_1 в N_2. Ориентированный двудольный граф называется *n-i/j-эспандером*, если $|N_1| = |N_2| = n$ ($|A|$ обозначает мощность A) и для всех подмножеств N' мощности n/i множества N_2 имеет место следующее:

$$|\{c \mid c \in N_1 \text{ и существует ребро из } c \text{ в некоторую вершину } N'\}| > n/j.$$

Лемма 1. *Для достаточно больших n существует n-8/2-эспандер, у которого входная степень каждой вершины из N_2 равна* 16.

Доказательство. Сопоставим каждой функции f: $\{1, \ldots, cn\} \to \{1, \ldots n\}$ двудольный граф $G_f \in S(c, 2n)$ с n входами и n выходами следующим образом: входы и выходы графа нумеруются числами от 1 до n и вход i соединяется с выходом ($j \bmod n$) тогда и только тогда, когда $f(j) = i$ ($j = 1, \ldots, cn$; $i = 1, \ldots n$). Различные функции могут порождать один и тот же граф. Функцию f назовем *плохой*, если существует множество I, состоящее из $n/2$ входов, и множество O из $n/8$ выходов, такие что все ребра, ведущие в O, выходят из вершин I. В противном случае функция f считается *хорошей*. Очевидно, что для хороших функций f граф G_f является n-8/2-эспандером с требуемыми свойствами. С целью доказать существование хорошей функции установим, что доля плохих функций среди всех таких функций стремится к 0 с ростом n [5, 6].

Существует n^{cn} функций f: $\{1, \ldots cn\} \to \{1, \ldots n\}$. Существует $\binom{n}{n/2} \cdot \binom{n}{n/8}$ способов выбрать множество I из $n/2$ входов и множество O из $n/8$ выходов. Для всякого выбора I и O найдется $(n/2)^{cn/8 \cdot n^{7cn/8}}$ функций f, таких что f — плохая функция, поскольку в G_f все ребра, входящие в O, исходят из I. Следовательно, существует самое большее $\binom{n}{n/2} \cdot \binom{n}{n/8} \times (n/2)^{cn/8 \cdot n^{7cn/8}}$ плохих функций. Следовательно, доля, которую мы хотим оценить, равна

$$\binom{n}{n/2} \cdot \binom{n}{n/8} \cdot (n/2)^{cn/8} \cdot n^{7cn/8} / n^{cn} = \binom{n}{n/2} \cdot \binom{n}{n/8} \Big/ 2^{cn/8} =$$

$$= O(1) \quad \text{для} \quad c \geqslant 16. \ \blacksquare$$

Пусть E'_n есть n-8/2-эспандер из леммы 1. Построим, исходя из E'_n, граф E_n, заменяя для каждой выходной вершины v

16 входящих в нее ребер полным бинарным деревом с 16 листьями, идентифицируя v с корнем дерева, а предшественников v — с листьями. Очевидно, что $E_n \in S(2, 16n)$.

Пусть $H_{b,d}$ есть графа, состоящий из d экземпляров E_b: $E_b^1, \ldots, E_b^d$, где для $2 \leqslant i \leqslant d$ выходные вершины графа E_b^i идентифицированы с выходными вершинами E_b^{i-1}. Итак, $H_{b,d} \in S(2, (15d+1)b)$. Множество выходных вершин E_b^i назовем i-м уровнем. Входные вершины графа E_b^i образуют нулевой уровень (рис. 1).

Лемма 2. *Граф $H_{b,d}$ можно заполнить с $2b+16$ камнями за $(15d+1)b$ ходов.*

Доказательство. Назовем уровень i *полным*, если во всех вершинах этого уровня находятся камни. Стратегия состоит в том, чтобы заполнить эти уровни один за другим. Каждый уровень является множеством сечения. Таким образом, как только

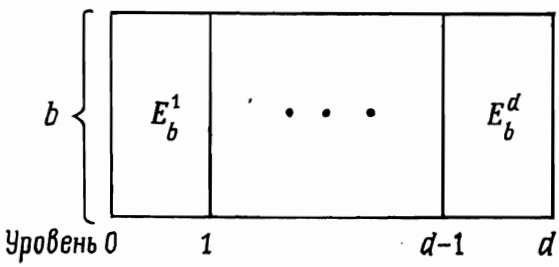


Рис. 1. Граф $H_{b,d}$.

будет заполнен новый уровень i, все камни, лежащие выше этого уровня, можно удалить. Значит, не больше чем $2b$ камней должно сохраниться на двух последовательных уровнях. В процессе заполнения $(i+1)$-го уровня, если i-й уровень заполнен, на деревьях между уровнями используются 16 дополнительных камней. Так как деревья попарно не пересекаются (за исключением листьев), то в каждую вершину камень помещается ровно один раз.

Лемма 3. *Граф $H_{b,d}$ можно заполнить с $4d+2$ камнями*[1]).

Доказательство. Глубиной вершины v называется число ребер в длиннейшем пути, ведущем в v. В графе $G \in S(2n)$ в каждую вершину глубины t можно поместить камни, имея $t+2$

[1]) То есть заполнить любую выходную вершину графа $H_{b,d}$. — *Прим. перев.*

камней (это легко устанавливается индукцией по t). Каждая вершина в графе $H_{b,d}$ имеет глубину не больше $4d$. ■

Центральным местом доказательства является

Лемма 4. *Для всех $i \in \{0, 1, \ldots d\}$ и всякого b, делящегося на 8, справедливо следующее утверждение. Пусть c — произвольная конфигурация на графе $H_{b,d}$, содержащая не более $b/8$ камней; N — произвольное множество вершин уровня i, содержащее $b/4$ вершин. Пусть далее M — произвольная последовательность ходов, начинающаяся с конфигурации c, такая что*

а) ни в одной промежуточной конфигурации не используется более $b/8$ камней;

б) в каждую вершину из N хотя бы раз помещается камень. Тогда M содержит по крайней мере 2^i ходов.

Доказательство. Индукция по i. Для $i = 0$ доказывать нечего. Предположим, что лемма верна для $i - 1$. В конфигурации c на графе размещено не более чем $b/8$ камней. Таким образом, для не менее чем $b/8$ вершин v из N ни один камень не лежит ни в v, ни в какой-либо другой вершине дерева, соединяющего v с $(i-1)$-м уровнем, за возможным исключением только листьев. Пусть N' есть некоторое подмножество этого множества вершин мощности $b/8$, а P — множество вершин уровня $i-1$, связанных с N'. По построению $H_{b,d}$, $|P| \geqslant b/2$. Так как ни в одной из вершин N' и в деревьях этих вершин (за исключением листьев) нет камней, в процессе выполнения M в каждой вершине множества P должен в некоторый момент оказаться камень (возможно, в самом начале процесса).

Разделим стратегию M на два этапа M_1, M_2, причем начальный этап M_1 такой, что на протяжении M_1 в некоторых $b/4$ вершинах P находятся или находились камни, а в остальных $b/4$ вершинах P нет и не было камней (в течение M_1). Этап M_2 — остальная часть M. Для M_1 выполняется предположение леммы; таким образом, в M_1 не меньше 2^{i-1} ходов. Поскольку M_1 оставляет на графе не больше чем $b/8$ камней, а M_2 также никогда не использует больше чем $b/8$ камней, то предположение леммы справедливо и для M_2. Следовательно, в M_2 также 2^{i-1} ходов, откуда следует утверждение леммы. ■

Выберем b так, чтобы $4d + 2 \leqslant b/8$, например $b = 32d + 16$. Тогда всякая стратегия, при которой камни помещаются в какие-нибудь $b/4$ выходных вершин $H_{b,d}$ и при этом используется не больше $4d + 2$ камней, содержит не меньше 2^d ходов. Таким образом, по крайней мере для одной из этих вершин v, помещение в v камня с использованием только $4d + 2$ камней требует $2d/(b/4) \geqslant 2^{(1-\varepsilon)d}$ ходов, так как $b = 0(d)$. Число вершин в $H_{b,d}$ равно $n = (15d + 1)b$. Итак, положим $d = O(\sqrt{n})$, $b = O(\sqrt{n})$, и теорема доказана. ■

Из приведенного выше построения получаем также

Следствие. Для всякой функции $f(n) = o(n \log n)$ существует семейство графов $G_n \in S(2, n)$, таких что любая стратегия, которая заполняет G_n, используя $f(n)$ камней, требует сверхполиномиального количества ходов.

Доказательство. Пусть $\gamma(n) = (n/f(n) \log n)^{1/2}$ и, следовательно, $f(n) = n/(\log n \cdot \gamma^2(n))$. Положим $G_n = H_{b,d}$ с $b = n/(\log n \times \times \gamma(n))$ и $d = O(\log n \cdot \gamma(n))$. ∎

Открытой является следующая интересная проблема: существует ли семейство графов $G_n \in S(2, n), n = 1, 2, \ldots$, таких что заполнение графа G_n с $O(n/\log n)$ камнями требует сверхполиномиального количества ходов? В качестве первого шага в решении этой проблемы Пиппенджер в [7] построил семейство графов, требующих для заполнения с $O(n/\log n)$ камнями нелинейного числа ходов.

ЛИТЕРАТУРА

[1] Cook S. A. An observation on time-storage trade off. — Proceedings Fifth Annual ACM Symp. on Theory of Computing, 1973, 29—33.
[2] Hopcroft J., Paul W., Waliant L. On time versus space. — J. ACM 24, 77, 332—337.
[3] Paterson M. S., Hewitt C. E. Comparative Schematology. Record of Project MAC Conference on Concurent Systems and Parallel Computation, pp. 119—128, 1970.
[4] Paul W., Tarjan R. E., Celoni J. R. Space bounds for a game on graphs. — Math. Systems Theory 10, 239—251 (1977).
[5] Пинскер М. С. On the complexity of a concentrator. — 7th International Teletraffic Congress, Stockholm, 1973.
[6] Pippenger N. Superconcentrators. — Technical Report, IBM Thomas J. Watson Research Center, Yorktown Heights, N. Y., 1976.
[7] Pippenger N A time-space trade off. — Journ. ACM 25, No. 3. 509—515 (1978).
[8] Sethi R. Complete register allocation problems. — SIAM J. Comput. 4, 226—248 (1975).

Доказательство теорем с помощью абстракций[1)]

Д. А. Плейстид[2)]

Определяется класс функций, называемых абстракциями, и приводятся их примеры. Эти функции отображают множество дизъюнктов S на, быть может, более простое множество дизъюнктов T, а резолюционные доказательства из S — на, быть может, более простые резолюционные доказательства из T. Для того чтобы найти доказательство дизъюнкта C из S, достаточно найти доказательство из T и попытаться обратить функцию абстракции. Предлагается несколько стратегий доказательства теорем, основанных на этой идее. Большинство стратегий полные. Приводится также метод употребления нескольких абстракций одновременно, что требует использования «мультидизъюнктов», которые являются мультимножествами литер, и связанных с ними «функций m-абстракции». Некоторые абстракции особенно интересны, поскольку они соответствуют отдельным интерпретациям множества дизъюнктов S. Применение абстракций дает возможность реализовать преимущества стратегий поддержки в произвольных полных резолюционных стратегиях.

0. СПИСОК ИСПОЛЬЗУЕМЫХ ОБОЗНАЧЕНИЙ

T — резолюционное доказательство,
C — произвольный дизъюнкт в доказательстве,
D — абстракция C,
Result (T) — результат (заключительный дизъюнкт) доказательства T,
C' — результат доказательства,
D' — абстракция C',
B — дизъюнкт, полученный из абстракций с помощью резолюции,
f — функция абстракции,
Res (T) — множество резолюций в доказательстве T,

[1)] David A. Plaisted. Theorem Proving with Abstraction, Artificial Intelligence, 16, № 1 (1981), 47—108.

[2)] Department of Computer Science, University of Illinois at Urbana — Champaign, Urbana, IL 61801, U. S. A.

Nodes (T) — множество узлов в доказательстве T,
N, N' — узлы в доказательстве,
d — глубина узла в доказательстве,
label (N) — метка узла N,
$\langle N1, N2, N3\rangle$ — резолюция (тройка узлов),
R, M — отношения между дизъюнктами, доказательствами,
$T \rightarrow_f U$ — доказательство U является абстракцией доказательства T, полученной с помощью функции абстракции f,
V, W, X, Y, Z — доказательства (как правило, абстрактные).

1. ВВЕДЕНИЕ

Использование аналогий, по-видимому, полезно во многих областях «решения задач»[1]) [5, 8, 13]. Мы введем специальный вид аналогии, который применяется для доказательства теорем в исчислении предикатов первого порядка. В частности, если дана задача A, мы преобразуем ее в более простую задачу B. Если A имеет решение, то имеет решение и B, и одно из решений задачи B будет обладать структурой, подобной структуре решения задачи A. Следовательно, мы можем использовать решения для B в качестве ориентиров в поисках решений для A. На этом пути мы даже не рассматриваем те возможные решения задачи A, которые не соответствуют никакому решению задачи B. Разумеется, B может иметь решения, даже если A их не имеет.

Мы используем эту идею для резолюционного доказательства теорем в исчислении предикатов первого порядка [3]. Данный подход представляется достаточно общим для того, чтобы применяться и к другим системам правил вывода, а также к логикам более высокого порядка. Мы определяем класс отображений, называемых «функциями абстракции», которые удовлетворяют некоторым условиям. Эти отображения преобразуют множество дизъюнктов A в более простое множество дизъюнктов B, причем доказательства из A соответствуют доказательствам из B, имеющим подобную структуру. Абстракции обсуждаются также в [9], но данные там стратегии отличаются от тех, которые представлены ниже. Мы приведем несколько функций абстракции и дадим метод получения таких отображений. Рассматриваются как синтаксические, так и семантические отображения Один полезный класс нетривиальных семантических абстракций может быть порожден полностью автоматически. Далее мы предлагаем неполную стратегию доказательства теорем, основанную на абстракциях. Эта стратегия может управ-

[1]) Решение задач (в оригинале «problem solving») — одно из направлений «искусственного интеллекта». — *Прим. перев.*

лять поиском доказательства как отдельных следствий множества дизъюнктов, так и поиском доказательства противоречивости множества дизъюнктов.

Затем обсуждаются некоторые новые правила вывода, связанные с резолюцией. В частности, мы вводим «m-дизъюнкты», которые являются мультимножествами литер, т. е. вместе с каждой литерой в m-дизъюнкте содержится число ее вхождений в этот m-дизъюнкт. Для m-дизъюнктов определяется вариант резолюции, называемый m-резолюцией, а также определяются m-абстракции. Они отображают множество m-дизъюнктов A на более простое множество B, такое что m-резолюционные доказательства из A отображаются на имеющие подобные структуры m-резолюционные доказательства из B. Преимущество m-абстракций состоит в том, что они сохраняют гораздо больше информации о структуре доказательства, чем обычные абстракции. И как следствие имеются простые полные стратегии доказательства теорем, основанные на m-абстракциях. Имеются также стратегии, использующие несколько m-абстракций одновременно, что соответствует одновременному применению нескольких аналогий. Таким образом, мы получаем сильно ограничивающую поиск стратегию, которая не имеет известных аналогов для обычной резолюции и обычных абстракций. Все предлагаемые стратегии, основанные на m-абстракциях, полны.

Затем обсуждаются ограниченные m-дизъюнкты. Это такие m-дизъюнкты, которые содержат меньше информации о числе вхождений литер в дизъюнкт. Дается определение абстракции и предлагаются полные стратегии доказательства теорем, основанные на ограниченных m-дизъюнктах. Преимущество ограниченных m-дизъюнктов состоит в том, что абстрактное поисковое пространство часто конечно и может быть полностью просмотрено без особых усилий.

Далее мы предлагаем специальный вид абстракции, который, вероятно, соответствует «неполностью определенной диаграмме», т. е. эти абстракции связаны с такими интерпретациями множества дизъюнктов, в которых часть интерпретации описана не полностью. Употребление таких абстракций, по-видимому, соответствует свойственному человеку подходу к решению задач, при котором рисуются диаграммы из точек и неопределенных областей, обозначающих несущественные детали. Отмечаются также и другие классы m-абстракций и ограниченных m-абстракций.

Применение абстракции и связанных с ними методов аналогии позволяет использовать семантическую информацию и специальные знания в общих иерархических стратегиях доказательства теорем. Основная идея заключается в построении сначала наброска доказательства и уточнении всех деталей впо-

следствии. Такая стратегия универсальна и лишена «близорукости» большинства систем доказательства теорем. Каждый шаг поиска в ней осмыслен и нетривиальным образом управляется структурой задачи в целом, а не только такой локальной информацией, как способность двух дизъюнктов резольвировать согласно определенной стратегии. Мы считаем такое локальное поведение одним из самых слабых мест современных систем доказательства теорем. Использование аналогии имеет дополнительное преимущество в том, что стратегия поиска становится все более и более ограничивающей по мере увеличения глубины вывода. При завершении поиска число возможных выборов более ограничено, чем в середине, несмотря на то что стратегия основана на прямых рассуждениях. Это резко отличается от ситуации, характерной для традиционных стратегий, в которых поисковое пространство с ростом глубины, по-видимому, экспоненциально увеличивается в объеме. Применение абстракции допускает также возможность нескольких уровней абстракции, причем каждый последующий уровень содержит меньше информации, чем предыдущий. Поиск на каждом уровне может управляться поиском на следующем, более высоком уровне абстракции.

Одно из преимуществ абстракции состоит в том, что она автоматически выбирает из входных дизъюнктов те, которые представляются относящимися к данной задаче. Таким образом, мы получаем преимущества стратегий поддержки. Однако поисковые стратегии, основанные на абстракции, оказываются совместимыми с другими полными резолюционными стратегиями, такими как лок-резолюция и $P1$-вывод. Следовательно, мы можем получить преимущества стратегии поддержки в резолюционных стратегиях, которые непосредственно не совместимы с ограничениями множества поддержки. Эта совместимость должна быть особенно полезна, когда имеется очень большое число входных дизъюнктов, из которых не все имеют отношение к данной задаче.

Мы используем довольно стандартную запись программ [1]). Для циклов мы применяем конструкции

loop ... while ... repeat

и *loop ... until ... repeat* [2]).

[1]) Для записи программ автор использует как обычные конструкции некоторых языков программирования, так и специальные обозначения, определяемые ниже. В тексте все ключевые слова этих конструкций будут даны на языке оригинала, а для читателя, не знакомого с языками программирования, дается подстрочный перевод. — *Прим. перев.*

[2]) цикл ... пока ... повторить и цикл ... до ... повторить. — *Прим. перев.*

while- и *until*-предложения могут встречаться в начале или в конце цикла. Если $A(x_1, \ldots, x_n)$ — *булево* выражение со свободными переменными $x_1, \ldots, x_n$, то мы используем выражение «*there exist* $x_1, \ldots, x_n$ *such that* $A(x_1, \ldots, x_n)$»[1]) в следующем смысле: его значение есть ИСТИНА, если формула $\exists x_1 \ldots \exists x_n A(x_1, \ldots, x_n)$ истинна, и ЛОЖЬ в противном случае. Если это значение — ИСТИНА, то $x_1, \ldots, x_n$ — те значения, при которых $A(x_1, \ldots, x_n)$ истинно. Таким образом, мы можем писать «*if there exist* $x_1, \ldots, x_n$ *such that* $A(x_1, \ldots, x_n)$ *then* (сделать что-либо с $x_1, x_2, \ldots, x_n$) *else* ... *fi*»[2]). Это соглашение позволяет нам писать программы без уточнения того, как в действительности найти $x_1, x_2, \ldots, x_n$, удовлетворяющие $A(x_1, x_2, \ldots, x_n)$, если они существуют.

2. ОБЫЧНЫЕ АБСТРАКЦИИ

Мы принимаем стандартную терминологию резолюционного доказательства теорем [12]. В частности, мы говорим, что дизъюнкт $C1$ *поглощает*[3]) дизъюнкт $C2$, если существует такая подстановка θ, что $C1\theta$ является подмножеством $C2$. Дизъюнкты C и D называются *вариантами*, если они являются примерами друг друга, т. е. если C и D совпадают с точностью до переименования переменных.

Определение. *Абстракция* есть сопоставление каждому дизъюнкту C множества дизъюнктов $f(C)$, такое что f обладает следующими свойствами:

(1) Если дизъюнкт $C3$ является резольвентой для $C1$ и $C2$ и $D3 \in f(C3)$, то существуют $D1 \in f(C1)$ и $D2 \in f(C2)$, такие что некоторая резольвента для $D1$ и $D2$ поглощает $D3$.

(2) $f(\mathrm{NIL}) = \{\mathrm{NIL}\}$. (NIL — пустой дизъюнкт.)

(3) Если $C1$ поглощает $C2$, то для каждой абстракции $D2$ дизъюнкта $C2$ существует абстракция $D1$ дизъюнкта $C1$, поглощающая $D2$.

Если f — отображение с указанными свойствами, то мы называем f *функцией абстракции*. Если, кроме того, $D \in f(C)$, мы называем D абстракцией от C. Как правило, абстракции удовлетворяют также свойству: каждый дизъюнкт D из $f(C)$ — тавтология, если таковыми является C.

Следующий результат дает нам довольно общий метод построения абстракций. Впоследствии мы приведем и другие методы.

---

[1]) Существуют $x_1, \ldots, x_n$, такие что $A(x_1 \ldots, x_n)$. — *Прим. перев.*

[2]) Если существуют $x_1, \ldots, x_n$, такие что $A(x_1, \ldots, x_n)$, то ... иначе ... ...; *fi* (*if* наоборот) обозначает конец оператора *if*. — *Прим. перев.*

[3]) В оригинале "subsume". Дизъюнкт $C1$ называют также поддизъюнктом $C2$, а дизъюнкт $C2$ — наддизъюнктом $C1$. — *Прим. перев.*

Теорема 2.1. *Предположим, что* φ *— отображение литер в литеры. Расширим* φ *до отображения дизъюнктов в дизъюнкты, положив по определению* $\varphi(C)=\{\varphi(L)\colon L\in C\}$. *Пусть* φ *удовлетворяет также следующим двум свойствам*:

(1) $\varphi(\bar{L})=\overline{\varphi(L)}$, *т. е.* φ *сохраняет отрицания.*

(2) *Если* C *и* D *— дизъюнкты и* D *является примером* C, *то* $\varphi(D)$ *— также пример* $\varphi(C)$, *т. е.* φ *сохраняет примеры. Тогда* φ *— функция абстракции. А точнее,* f *является функцией абстракции, где* $f(C)=\{\varphi(C)\}$.

Доказательство. Все свойства, кроме (1), легко проверяются. Для установления (1) мы поступим следующим образом.

Допустим, что $C3$ есть резольвента для $C1$ и $C2$. Тогда существуют такие дизъюнкты $A1$ и $A2$, что $A1\subset C1$, $A2\subset C2$, и такие подстановки $\alpha1$ и $\alpha2$, что $A1\alpha1=\{L\}$ и $A2\alpha2=\{\bar{L}\}$ для некоторой литеры L. Пусть $\alpha1$ и $\alpha2$ — наиболее общие такие подстановки, и пусть $C3=(C1-A1)\alpha1\cup(C2-A2)\alpha2$. Мы хотим показать, что некоторая резольвента для $\varphi(C1)$ и $\varphi(C2)$ поглощает $\varphi(C3)$. Имеем $\varphi(C1\alpha1)=\varphi((C1-A1)\alpha1)\cup\{\varphi(L)\}$, $\varphi(C2\alpha2)=\varphi((C2-A2)\alpha2)\cup\{\varphi(\bar{L})\}$ и по свойствам отображения φ $\varphi(\bar{L})=\overline{\varphi(L)}$. Поэтому дизъюнкт $\varphi((C1-A1)\alpha1)\cup\varphi((C2-A2)\alpha2)$ либо сам есть резольвента $\varphi(C1\alpha1)$ и $\varphi(C2\alpha2)$, либо содержит собственное подмножество, являющееся резольвентой для $\varphi(C1\alpha1)$ и $\varphi(C2\alpha2)$ (когда, к примеру, $\varphi(L)\in\varphi((C1-A1)\alpha1)$). Заметим, что $\varphi(C3)=\varphi((C1-A1)\alpha1)\cup\varphi((C2-A2)\alpha2)$. Следовательно, некоторая резольвента $\varphi(C1\alpha1)$ и $\varphi(C2\alpha2)$ поглощает $\varphi(C3)$. А так как по свойствам φ $\varphi(C1\alpha1)$ — это пример $\varphi(C1)$, а $\varphi(C2\alpha2)$ — пример $\varphi(C2)$, то по свойствам резолюции некоторая резольвента $\varphi(C1)$ и $\varphi(C2)$ поглощает $\varphi(C3)$. Потребовав выполнения соответствующих условий, мы могли бы доказать подобную теорему и тогда, когда φ — отношение между литерами. ∎

Более общим является следующий результат.

Теорема 2.2. *Предположим, что* F *— множество отображений литер в литеры и для любого* $\varphi\in F$ *и любой литеры* $L\varphi(\bar{L})=\overline{\varphi(L)}$. *Если* C *— дизъюнкт, то, как и прежде, положим* $\varphi(C)=\{\varphi(L)\colon L\in C\}$. *Допустим, что если дизъюнкт* D *является примером дизъюнкта* C, *то для каждого* $\varphi2\in F$ *существует* $\varphi1\in F$, *такое что* $\varphi2(D)$ *есть пример* $\varphi1(C)$. *Определим* f *посредством* $f(C)=\{\varphi(C)\colon\varphi\in F\}$. *Тогда* f *— функция абстракции.*

Доказательство. Свойства (2) и (3), как и раньше, легко проверяются. Мы покажем, что f удовлетворяет свойству (1).

Пусть $C3$ — резольвента для $C1$ и $C2$. Тогда существуют множества литер $A1$ и $A2$ и подстановки $\alpha1$ и $\alpha2$, такие что $A1\subset C1$, $A2\subset C2$ и $C3=(C1-A1)\alpha1\cup(C2-A2)\alpha2$. Кроме

того, для некоторой литеры L $A1\alpha1 = \{L\}$ и $A2\alpha2 = \{\bar{L}\}$. Мы хотим показать, что для любого $\varphi3 \in F$ найдутся $\varphi1 \in F$ и $\varphi2 \in F$, такие что некоторая резольвента $\varphi1(C1)$ и $\varphi2(C2)$ поглощает $\varphi3(C3)$.

Пусть $\varphi1$ и $\varphi2$ таковы, что $\varphi3(C1\alpha1)$ есть пример $\varphi1(C1)$, а $\varphi3(C2\alpha2)$ — пример $\varphi2(C2)$. Такие $\varphi1$ и $\varphi2$ благодаря наложенным на F условиям обязательно найдутся. Имеем

$$\varphi3(C3) = \varphi3((C1 - A1)\alpha1) \cup \varphi3((C2 - A2)\alpha2),$$
$$\varphi3(A1\alpha1) = \{\varphi3(L)\} \text{ и } \varphi3(A2\alpha2) = \{\varphi3(\bar{L})\} = \{\overline{\varphi3(L)}\}.$$

Как и выше, отсюда вытекает, что $\varphi3(C3)$ обладает подмножеством (возможно, собственным подмножеством), которое является резольвентой для $\varphi3(C1\alpha1)$ и $\varphi3(C2\alpha2)$. Так как $\varphi3(C1\alpha1)$ — пример $\varphi1(C1)$, а $\varphi3(C2\alpha2)$ — пример $\varphi2(C2)$, то некоторая резольвента от $\varphi1(C1)$ и $\varphi2(C2)$ поглощает $\varphi3(C3)$. ∎

Если f — функция абстракции, удовлетворяющая условиям доказанной выше теоремы, то мы говорим, что f определена посредством отображений литер. Оказывается, не все абстракции определены посредством отображений литер.

2.1. Примеры абстракций

Используя эти теоремы, мы можем построить много абстракций. Сейчас мы дадим несколько примеров абстракций, каждая из которых может быть получена из приведенных выше теорем. Первый пример синтаксической абстракции может быть получен из теоремы 2.2, другие — из теоремы 2.1. Пример семантической абстракции получается из теоремы 2.2.

2.1.1. Примеры синтаксических абстракций

(1) Основная абстракция. Если C — дизъюнкт, то $f(C) = \{C'\colon C'$ — основной пример $C\}$. Заметим, что $f(C)$ будет, как правило, бесконечным множеством дизъюнктов.

(2) Пропозициональная абстракция. Если C — дизъюнкт $\{L_1, L_2, \ldots, L_k\}$, то $f(C)$ есть $\{C'\}$, где C' — дизъюнкт $\{L_1', L_2', \ldots, L_k'\}$, а L_i' для $1 \leqslant i \leqslant k$ определяется следующим образом:

Если L_i имеет вид $P(t_1, \ldots, t_n)$, то L_i' есть P. Если же L_i имеет вид $\neg P(t_1, \ldots, t_n)$, то L_i' есть $\neg P$.

Таким образом, $f(C)$ является дизъюнктом в пропозициональном исчислении.

(3) Переименование предикатных и функциональных символов. Для дизъюнкта C $f(C) = \{C'\}$, где C' — дизъюнкт, полученный из C переименованием всех функциональных и предикатных

символов каким-либо систематическим образом. Переименование не обязательно должно быть взаимно однозначным; два различных предикатных или функциональных символа могут быть заменены на один и тот же символ. Однако замена сразу и предикатного, и функционального символа на один и тот же символ недопустима.

(4) Изменение знаков литер. Пусть Q — некоторое множество предикатных символов. Если C — дизъюнт $\{L_1, \ldots, L_k\}$, то $f(C)$ есть $\{C'\}$, где C' — дизъюнкт $\{L'_i, \ldots, L'_k\}$, а L'_i для $1 \leqslant i \leqslant k$ определяются следующим образом:

Если L_i имеет вид $P(t_1, \ldots, t_n)$ и $P \in Q$, то L'_i есть $\neg P(t_1, \ldots, t_n)$. Если же L_i имеет вид $\neg P(t_1, \ldots, t_n)$ и $P \in Q$, то L'_i есть $P(t_1, \ldots, t_n)$. В остальных случаях L'_i есть L_i.

(5) Перестановка аргументов. Для дизъюнкта C $f(C) = \{C'\}$, где C' есть C с измененным каким-либо систематическим образом порядком аргументов некоторых функциональных или предикатных символов.

(6) Вычеркивание аргументов. Для дизъюнкта C $f(C) = \{C'\}$, где C' получается из C вычеркиванием некоторых аргументов каких-либо функциональных или предикатных символов. Например, $g(t_1, \ldots, t_n)$ можно везде заменить на $g(t_2, \ldots, t_n)$. Заметим, что пропозициональная абстракция представляет собой частный случай вычеркивания аргументов (вычеркиваются все аргументы у всех предикатных символов).

2.1.2. Пример семантической абстракции

Каждому дизъюнкту C мы следующим образом сопоставим множество дизъюнктов $f(C)$.

Пусть $\mathcal{I}$ — интерпретация множества дизъюнктов над некоторым множеством функциональных и предикатных символов, а $\mathcal{D}$ — область интерпретации $\mathcal{I}$. В интерпретации $\mathcal{I}$ равенство может трактоваться как любой другой предикатный символ, т. е. отношение $a_1 = a_2$ может быть истинным, в $\mathcal{I}$, даже если a_1 и a_2 — различные элементы из $\mathcal{D}$.

Каждой основной литере вида $P(t_1, \ldots, t_n)$ мы сопоставим литеру $P(a_1, \ldots, a_n)$, где $a_i \in \mathcal{D}$ и a_i является значением t_i в интерпретации $\mathcal{I}$ при $1 \leqslant i \leqslant n$. Каждой литере $\bar{P}(t_1, \ldots, t_n)$ мы сопоставим $\bar{P}(a_1, \ldots, a_n)$.

Каждому основному дизъюнкту $C = \{L_1, \ldots, L_k\}$ мы сопоставим $C' = \{L'_1, \ldots, L'_k\}$, где L'_i — литера, сопоставленная L_i указанным выше способом. Если $C1$ — произвольный дизъюнкт, то $f(C1) = \{D: D$ сопоставлен некоторому основному примеру C дизъюнкта $C1\}$. Мы называем f *$\mathcal{I}$-абстракцией* или *абстракцией, полученной из $\mathcal{I}$*.

Пример. Если $\mathcal{I}$ — обычная интерпретация арифметики, то с дизъюнктом $\{\neg(x \leqslant y), \neg(y \leqslant z), x \leqslant z\}$ мы ассоциируем дизъюнкты $\{\neg(1 \leqslant 2), \neg(2 \leqslant 3), 1 \leqslant 3\}$, $\{\neg(1 \leqslant 5), \neg(5 \leqslant 2), 1 \leqslant 2\}$, $\{\neg(8 \leqslant 7), \neg(7 \leqslant 6), 8 \leqslant 6\}$ и т. д. Дизъюнкты из $f(C)$ будут истинны в $\mathcal{I}$, если C истинен, но не обязаны быть таковыми в общем случае.

Заметим, что $f(C)$ может содержать бесконечно много дизъюнктов, если f является $\mathcal{I}$-абстракцией и $\mathcal{D}$ бесконечна. Однако если область $\mathcal{D}$ конечна, то $f(C)$ будет конечным для каждого дизъюнкта C. Такие абстракции представляются наиболее полезными. Действительно, они могут порождаться автоматически при выборе $\mathcal{D} = \{1, 2, \ldots, n\}$ для некоторого малого n и при интерпретации каждого функционального символа из C произвольной функцией из $\mathcal{D}$ в $\mathcal{D}$ для всех $C \in S$. Заметим, что все вхождения одного и того же функционального символа во всех дизъюнктах из S должны интерпретироваться одной и той же функцией из $\mathcal{D}$ в $\mathcal{D}$. Если абстракция f определена таким способом, то по данному дизъюнкту C легко вычисляется $f(C)$.

2.2. Алгебраические свойства абстракции

Определение. Пусть f_1 и f_2 — абстракции. *Композиция* f_1 и f_2, обозначаемая через $f_2 f_1$, определяется формулой $f_2 f_1(C) = \bigcup \{f_2(D) : D \in f_1(C)\}$.

Определение. *Тождественная* абстракция — это отображение f, такое что $f(C) = \{C\}$ для всех дизъюнктов C.

Определение. Абстракции f_1 и f_2 называются взаимно *обратными*, если $f_1 f_2 = f$ и $f_2 f_1 = f$, где f — тождественная абстракция. Если абстракция имеет обратную, то она, по существу, не приводит к потере информации о множестве дизъюнктов. Например, взаимно однозначное переименование предикатных символов является обратимой абстракцией.

Теорема 2.3. *Композиция двух абстракций есть абстракция.*

Доказательство. Пусть f_1 и f_2 — абстракции. Мы покажем, что $f_2 f_1$ также абстракция.

Пусть $C3$ — резольвента от $C1$ и $C2$, $E3 \in f_2 f_1(C3)$ и $D3$ — такой элемент из $f_1(C3)$, что $E3 \in f_2(D3)$. Так как f_1 — абстракция, обязаны найтись дизъюнкты $D1 \in f_1(C1)$ и $D2 \in f_1(C2)$, такие что некоторая резольвента D дизъюнктов $D1$ и $D2$ поглощает дизъюнкт $D3$. Так как f_2 — абстракция, а D поглощает $D3$, то существует дизъюнкт $E' \in f_2(D)$, поглощающий $E3$. Кроме того, поскольку f_2— абстракция, существуют $E1 \in f_2(D1)$ и $E2 \in f_2(D2)$, такие что некоторая резольвента E дизъюнктов $E1$ и $E2$ поглощает E'. Следовательно, E поглощает $E3$. Так

как $E1 \in f_2 f_1(C1)$ и $E2 \in f_2 f_1(C2)$, мы показали, что $f_2 f_1$ обладает свойством (1) (рис. 1). Заметим, что для получения этого доказательства нам не понадобилось свойство (3) отображения f_2. Легко показать, что $f_2 f_1$ обладает свойствами (2) и (3). Таким образом, $f_2 f_1$ — функция абстракции. ∎

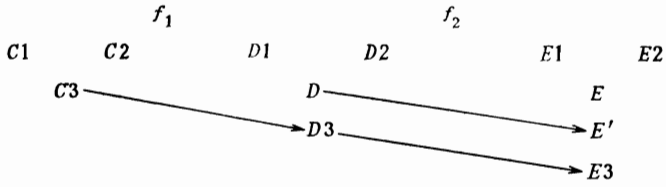


Рис. 1. Композиция двух абстракций.

Определение. Если f_1 и f_2 — абстракции, то их *объединение* f для всех дизъюнктов C определяется формулой $f(C) = f_1(C) \cup f_2(C)$.

Теорема 2.4. *Если f_1 и f_2 — абстракции, то их объединение также абстракция.*

Доказательство очевидно.

Нетрудно показать, что если f_1 и f_2 — абстракции, определенные посредством отображений литер, то объединение и композиция f_1 и f_2 также определяются посредством отображений литер.

2.3. Объяснение определений

Определение абстракции может показаться несколько необычным и, вероятно, заслуживает некоторого интуитивного оправдания. Могло бы показаться разумным потребовать в определении абстракции, чтобы дизъюнкт $D3$ сам был резольвентой $D1$ и $D2$, а не содержал такую резольвенту в качестве подмножества. Однако при переходе от $C1$ к $D1$ может случиться так, что различные литеры из $C1$ соответствуют одинаковым литерам в $D1$ и аналогично для $C2$ и $D2$. Применяя к $D1$ и $D2$ правило резолюции, мы можем удалить «слишком много» литер, поскольку мы можем вычеркнуть и такие из них, которые соответствуют литерам, все еще остающимся в $C1$ или $C2$. Приведем пример.

Рассмотрим пропозициональную абстракцию. Пусть $C1$ — дизъюнкт $\{\bar{P}_1(a), \bar{P}_1(b), P_2(C))$, C_2 — это $\{P_1(a)\}$, и пусть $C3$ — дизъюнкт $\{\bar{P}_1(b), P_2(C)\}$, являющийся резольвентой для $C1$ и $C2$. Единственными абстракциями $C1$, $C2$ и $C3$ являются соответственно $\{\bar{P}_1, P_2\}$, $\{P_1\}$ и $\{\bar{P}_1, P_2\}$. Однако $\{\bar{P}_1, P_2\}$ не есть резольвента от $\{\bar{P}_1, P_2\}$ и $\{P_1\}$, но содержит собственное под-

множество (а именно, P_2), являющееся резольвентой для $\{\bar{P}_1, P_2\}$ и $\{P_1\}$.

Следующий пример показывает, почему в определении абстракции требуется, чтобы некоторая резольвента от $D1$ и $D2$ поглощала $D3$, а не являлась его подмножеством. Предположим, что абстракция вычеркивает второй аргумент предиката $P1$. Пусть $C1$, $C2$, $C3$, $D1$, $D2$ и $D3$ имеют следующий вид:

$$C1\colon \{P1(z, f(x))\}, D1, \qquad \{P1(z)\},$$
$$C2\colon \{\bar{P}1(y, y), P2(y)\}, \quad D2\colon \{\bar{P}1(y), P2(y)\},$$
$$C3\colon \{P2(f(x))\}, \qquad D3\colon \{P2(f(x))\}.$$

Единственная резольвента от $D1$ и $D2$ есть дизъюнкт $\{P2(y)\}$, который не является подмножеством $\{P2(f(x))\}$. Однако $\{P2(y)\}$ поглощает $\{P2(f(x))\}$.

2.4. Абстракции резолюционных доказательств

Мы теперь покажем, как абстракции могут использоваться для управления поиском доказательства дизъюнкта C из множества дизъюнктов S. Сначала мы покажем, что если доказательство C из S существует, то существует и «абстрактное доказательство» некоторого дизъюнкта, поглощающего абстракцию C, из абстракций дизъюнктов из S. Затем мы опишем процедуры, которые по данному абстрактному доказательству пытаются восстановить исходное доказательство. Хотя последнее не всегда возможно, мы можем указать полную стратегию доказательства теорем, которая использует абстрактные доказательства как ориентир в поисках доказательства C из S.

Если f — функция абстракции, а S — множество дизъюнктов, то мы пишем $f(S)$ для обозначения $\bigcup\{f(C)\colon C \in S\}$.

Теорема 2.5. *Пусть S — множество дизъюнктов, f — функция абстракции для S, и пусть C' - дизъюнкт, выводимый резолюцией из S, и $D' \in f(C')$. Тогда найдется дизъюнкт B', выводимый резолюцией из $f(S)$ и поглощающий D'.*

Доказательство. Проводится индукцией по глубине вывода C'. Если $C' \in S$, то теорема справедлива, поскольку мы можем положить B' равным D'. Допустим, что C' есть резольвента от $C1$ и $C2$, причем $C1$ и $C2$ могут быть выведены из S с помощью доказательств, глубина которых меньше глубины доказательства C'. Пусть $D1$ и $D2$ — абстракции дизъюнктов $C1$ и $C2$ соответственно, такие что некоторая резольвента D дизъюнктов $D1$ и $D2$ поглощает D'. Дизъюнкты $D1$ и $D2$ обязаны существовать по определению абстракции. По индуктивному предположению должны существовать дизъюнкты $B1$ и $B2$, выводимые из $f(S)$,

причем $B1$ поглощает $D1$, а $B2$ поглощает $D2$. Отсюда из определения поглощения следует, что либо $B1$, либо $B2$, либо некоторая их резольвента B поглощает D. А значит, либо $B1$, либо $B2$, либо некоторая их резольвента B' поглощает дизъюнкт D'. На этом доказательство теоремы завершается. ∎

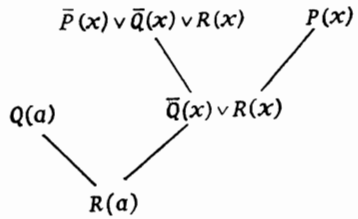
Заметим, что вывод B' из $f(S)$ будет иметь глубину, не бо́льшую, чем глубина вывода C из S.

Следствие. *Если S противоречиво, то противоречиво также и $f(S)$.*

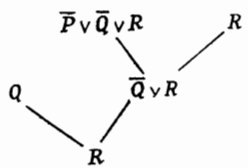
Доказательство. Положим C' равным NIL. Тогда по свойствам функций абстракции D' также есть NIL. Так как B' — подслучай D', то и B' есть NIL. А поскольку B' выводим из $f(S)$, то $f(S)$ противоречиво. ∎

Данная теорема может использоваться для доказательства непротиворечивости S, но основную ценность для нас представляет та информация, которую нам может дать доказательство из $f(S)$ о структуре возможного доказательства из S. Приведем несколько примеров.

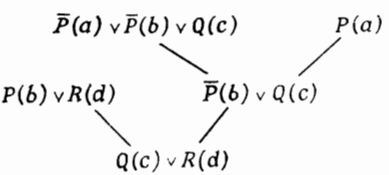
Пример 1. Рассмотрим следующее доказательство:



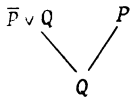
Пусть f — пропозициональная абстракция, т. е. $P(t_1, \ldots, t_n)$ заменяется на P, $\bar{P}(t_1, \ldots, t_n)$ заменяется на $\bar{P}$ и т. д. Мы имеем следующее абстрактное доказательство:



Пример 2. Рассмотрим следующее доказательство:

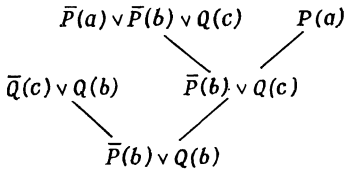


Пусть P — пропозициональная абстракция. Получаем следующее абстрактное доказательство:

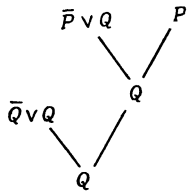


Заметим, что мы потеряли литеру $\bar{P}$ из $\{\bar{P}, Q\}$ в результате резолюции с P, хотя литера $\bar{P}(b)$ остается в $\{P(b), Q(c)\}$.

Пример 3.



Пусть, как и раньше, f — пропозициональная абстракция. Получаем следующее абстрактное доказательство:



Заметим, что мы включаем дизъюнкт $\bar{Q} \vee Q$ в абстрактное доказательство, хотя он и тавтологичен. В данном случае это не является необходимым, но окажется полезным в дальнейшем, когда нам понадобится, чтобы абстрактное доказательство имело такой же вид, как и исходное доказательство.

2.5. Терминология, относящаяся к доказательствам

Мы сейчас введем некоторую терминологию, которая поможет описывать и анализировать различные процедуры использования абстрактных доказательств в качестве ориентира в поиске доказательства дизъюнкта C из множества дизъюнктов S. Мы считаем дизъюнкты, являющиеся вариантами друг друга, идентичными. Этого можно достичь, выбирая переменные в дизъюнктах некоторым каноническим способом. Хотя проверка, являются ли два дизъюнкта вариантами, в общем случае полиномиально эквивалентна задаче обнаружения изоморфизма графов, на практике это сделать нетрудно. Если бы варианты не

считались идентичными, то появилось бы гораздо больше возможных резольвент двух дизъюнктов, поскольку многие резольвенты могли бы быть вариантами друг друга.

Определение. Резолюционное доказательство T — это конечное множество узлов вместе с множеством троек таких узлов. Каждый узел N имеет к тому же метку, обозначаемую $\mathrm{label}(N)$, которая является дизъюнктом. Никакие две различные метки узлов T не могут быть вариантами, однако один и тот же дизъюнкт может служить меткой более одного узла в T. Если $\langle N1, N2, N3\rangle$ — тройка узлов T, то мы требуем, чтобы $\mathrm{label}(N3)$ была резольвентой для $\mathrm{label}(N1)$ и $\mathrm{label}(N2)$. Узлы $N1$ и $N2$ могут совпадать. Мы обозначаем множество троек T посредством $\mathrm{Res}(T)$, а множество узлов посредством $\mathrm{Nodes}(T)$. Каждая тройка называется *резолюцией*. Если $\langle N1, N2, N3\rangle \in \mathrm{Res}(T)$, то мы требуем, чтобы и $\langle N2, N1, N3\rangle \in \mathrm{Res}(T)$. Узел в T, не являющийся третьей компонентой никакой тройки T, называется *начальным узлом* T. Метка такого узла называется *начальным дизъюнктом* T. Узел, не являющийся ни первой, ни второй компонентой никакой тройки из T, называется *конечным узлом* T. Метка такого узла называется конечным дизъюнктом T. Наконец, мы требуем, чтобы имелась функция «depth», сопоставляющая узлам T неотрицательные целые числа так, что

|a| $\mathrm{depth}(N) = 0$ для всех начальных узлов N из T,

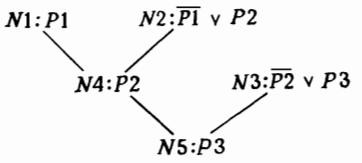
|b| $\mathrm{depth}(N) = 1 + \min\{\max(\mathrm{depth}(N1), \mathrm{depth}(N2)) : \langle N1, N2, N\rangle \in \mathrm{Res}(T)\}$.

Мы называем $\mathrm{depth}(N)$ глубиной узла N. Таким образом, резолюционное доказательство является специальным видом «гиперграфа» с помеченными узлами. Заметим, что единственный узел с меткой — это допустимое резолюционное доказательство. Наличие функции глубины гарантирует, что никакой узел не используется для своего собственного вывода, т. е. доказательство не имеет циклов. Мы иногда будем обозначать тройку $\langle N1, N2, N3\rangle$ доказательства T посредством $\langle C1, C2, C3\rangle$, где $C1 = \mathrm{label}(N1)$, $C2 = \mathrm{label}(N2)$ и $C3 = \mathrm{label}(N3)$.

Определение. Если S — множество дизъюнктов, то *резолюционное доказательство из* S — это такое резолюционное доказательство, в котором метки всех начальных узлов являются дизъюнктами из S.

Приведем пример резолюционного доказательства из $\{P1, \overline{P1} \vee P2, \overline{P2} \vee \overline{P3}\}$. Пусть доказательство T имеет узлы $N1$, $N2$, $N3$, $N4$, $N5$ с метками $P1$, $\overline{P1} \vee P2$, $\overline{P2} \vee P3$, $P2$ и $P3$ соответственно. Мы пишем $N : C$ для указания того, что C яв-

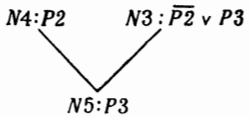
ляется меткой узла N. Тройки доказательства T суть $\{\langle N1, N2, N4\rangle, \langle N2, N1, N4\rangle, \langle N4, N3, N5\rangle, \langle N3, N4, N5\rangle\}$. Это соответствует следующему доказательству:



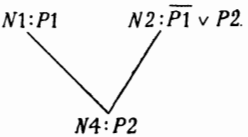
Здесь $N1$, $N2$ и $N3$ — начальные узлы, а $P1$, $\overline{P1} \vee P2$ и $\overline{P2} \vee P3$ — начальные дизъюнкты. Единственным конечным узлом в данном примере является $N5$, а $P3$ — единственный конечный дизъюнкт. (В некоторых примерах может присутствовать более одного конечного узла или дизъюнкта.) Глубина $N1$, $N2$ и $N3$ есть 0, глубина $N4$ — это 1 и глубина $N5$ равна 2.

Определение. Если $T1$ и $T2$ — резолюционные доказательства, то мы пишем $T1 \subset T2$ для обозначения того, что $\mathrm{Res}(T1)$ — подмножество $\mathrm{Res}(T2)$ и $\mathrm{Nodes}(T1)$ — подмножество $\mathrm{Nodes}(T2)$. Мы называем $T1$ *поддоказательством* $T2$. Если все начальные узлы $T1$ являются также начальными узлами $T2$, мы называем $T1$ *начальным поддоказательством* $T2$.

Следующее доказательство является поддоказательством предыдущего:



А это — начальное поддоказательство:



Определение. Если T — резолюционное доказательство и $\langle N1, N2, N3\rangle \in \mathrm{Res}(T)$, то мы называем $N1$ и $N2$ *предшественниками* $N3$, а $N3$ *последователем* $N1$ и $N2$.

Определение. *Глубина* резолюционного доказательства T — это максимальная глубина узлов T. Глубина резолюции $\langle N1, N2, N3\rangle$ в T — это глубина $N3$ в T. Если $\mathrm{label}(N) = C$, то мы часто говорим о глубине C вместо глубины N. Заметим, что C может иметь более одной глубины в T.

Если T — резолюционное доказательство и дизъюнкт C является меткой некоторого узла из T, то мы говорим, что C

имеет вхождения в T. Говоря неформально, мы называем C *элементом* T.

Определение. Если конечный дизъюнкт C резолюционного доказательства T единственный, то мы полагаем $\mathrm{Result}(T)$ равным C. Заметим, что C может встречаться более чем при одном узле T, однако C должен являться меткой конечного узла T.

Определение. Предположим, что S — множество дизъюнктов, а C — дизъюнкт. *Резолюционное доказательство* C из S — это такое резолюционное доказательство T из S, что C является меткой некоторого узла в T.

Определение. Предположим, что T — резолюционное доказательство из S. Мы говорим тогда, что T — минимальное доказательство из S, если:

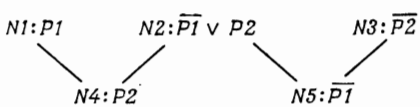
(a) T имеет ровно один конечный узел и

(b) для каждого неначального узла N существуют узлы $N1$ и $N2$ из T, такие что $\langle N1, N2, N\rangle \in \mathrm{Res}(T)$ и $\langle N2, N1, N\rangle \in \mathrm{Res}(T)$ и никакие тройки из T не имеют N третьей компонентой ($N1$ и $N2$ могут совпадать).

Заметим, что если T — минимальное доказательство из S, то значение $\mathrm{Result}(T)$ определено. Минимальное доказательство не обязано быть минимальным в обычном смысле. Может случиться так, что конечный дизъюнкт T или какие-либо другие дизъюнкты из T встречаются более чем при одном узле из T, т. е. некоторые леммы могут быть доказаны по нескольку раз. Мы говорим, что T есть *минимальное доказательство* C *из* S, если T — минимальное доказательство из S и $\mathrm{Result}(T) = C$.

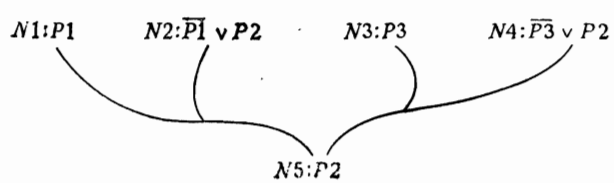
Ниже приводятся два примера доказательств, не являющихся минимальными.

Пример 1.



Это доказательство не является минимальным, поскольку имеет два конечных узла.

Пример 2.



Это доказательство неминимально, поскольку узел $N5$ появляется в результате двух разных резолюций.

Определение. Пусть T и T' — два резолюционных доказательства. Мы говорим тогда, что T и T' имеют одинаковую форму, если между узлами T и T' существует отношение «$\sim$», обладающее следующими свойствами:

(1) Для каждого узла N из T найдется узел N' из T', такой что $N \sim N'$, и для каждого узла N' из T' найдется узел N из T, такой что $N \sim N'$.

(2) Пусть $\langle N1, N2, N3\rangle$ — резолюция из T (т. е. элемент $\mathrm{Res}(T)$), $\langle N1', N2', N3'\rangle$ — резолюция из T' и $N3 \sim N3'$. Тогда либо $N1 \sim N1'$ и $N2 \sim N2'$, либо $N1 \sim N2'$ и $N2 \sim N1'$. И то и другое может быть истинным, если $N1 = N2$ или $N1' = N2'$.

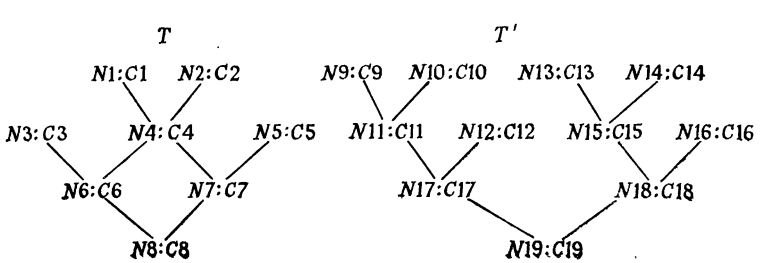
(3) Пусть N — узел T, N' — узел T' и $N \sim N'$. Узел N является начальным в T тогда и только тогда, когда N' — начальный в T'; N — конечный в T тогда и только тогда, когда N' — конечный в T'.

(4) Отношение «$\sim$» является взаимно однозначным соответствием между конечными узлами T и T'.

В этом случае мы называем «$\sim$» *соответствием формы* между T и T'. Свойство (1) соответствия формы является на самом деле логическим следствием свойств (2), (3) и (4). Основной смысл соответствия формы заключается в том, что если T и T' представлены множествами *деревьев* резолюционных доказательств, то эти множества деревьев имеют одинаковую форму (метки узлов в деревьях игнорируются). Мы пишем $T \sim T'$, если «$\sim$» — соответствие формы между T и T'. Заметим, что отношение «иметь одну и ту же форму» является отношением эквивалентности. Кроме того, если $T \sim T'$, то глубина T совпадает с глубиной T'.

Ниже приводится пример соответствия формы между доказательствами, имеющими различные структуры.

Пример 3.



Узлы доказательств T и T' связываются соответствием формы следующим образом:

T	T'
$N1$	$N9$, $N13$
$N2$	$N10$, $N14$
$N3$	$N12$
$N4$	$N11$, $N15$
$N5$	$N16$
$N6$	$N17$
$N7$	$N18$
$N8$	$N19$

Мы расширим понятие соответствия формы «$\sim$» между T и T' до отношения между резолюциями из T и T' следующим образом.

Пусть $\langle N1, N2, N3\rangle \in \mathrm{Res}(T)$ и $\langle N1', N2', N3'\rangle \in \mathrm{Res}(T')$. Тогда мы говорим, что $\langle N1, N2, N3\rangle \sim \langle N1', N2', N3'\rangle$, если $N1 \sim N1'$, $N2 \sim N2'$ и $N3 \sim N3'$ или если $N1 \sim N2'$, $N2 \sim N1'$ и $N3 \sim N3'$.

Если $T1$ и $T2$ — резолюционные доказательства, а «$\sim$» — соответствие формы между ними, то мы говорим, что $C1 \sim C2$ тогда и только тогда, когда существуют узлы $N1$ из $T1$ и $N2$ из $T2$, такие что $C1 = \mathrm{label}(N1)$, $C2 = \mathrm{label}(N2)$ и $N1 \sim N2$.

Пусть R — бинарное отношение между дизъюнктами. Мы обобщим R до бинарного отношения между резолюционными доказательствами следующим образом: $R(U, U')$ истинно тогда и только тогда, когда U и U' имеют одну и ту же форму и существует соответствие формы «$\sim$» между U и U', такое что из $C \sim C'$ следует $R(C, C')$ для всех дизъюнктов C в U и всех дизъюнктов C' в U'.

Пусть $R1$ и $R2$ — бинарные отношения между дизъюнктами, а U и U' — резолюционные доказательства. Мы пишем тогда $(R1; R2)(U, U')$, если существует соответствие формы «$\sim$» между U и U', такое что

(a) если N — начальный узел U, N' — начальный узел U' и $N \sim N'$, то $R1(\mathrm{label}(N), \mathrm{label}(N'))$ истинно,

(b) если N — неначальный узел U, N' — неначальный узел U' и $N \sim N'$, то $R2(\mathrm{label}(N), \mathrm{label}(N'))$ истинно.

Это позволит нам устанавливать отношение между начальными дизъюнктами, отличное от отношения между дизъюнктами, таковыми не являющимися. В приведенном выше примере 3 $R(T, T')$ было бы истинно, если бы $R(C1, C9)$, $R(C1, C13)$, $R(C2, C10)$, $R(C2, C14)$, $R(C3, C12)$ и т. д. были истинными. Кроме того, $(R1; R2)(T, T')$ было бы истинно, если бы

$R1(C1, C9)$, $R1(C1, C13)$, $R1(C2, C10)$, $R1(C2, C14)$, $R1(C3, C12)$, $R1(C5, C16)$ были истинными и если бы истинными были $R2(C4, C11)$, $R2(C4, C15)$, $R2(C6, C17)$ и т. д.

Определение. Пусть R — тождественное отношение, т. е. $R(C1, C2)$ истинно тогда и только тогда, когда $C1$ и $C2$ — один и тот же дизъюнкт. Мы говорим, что доказательства U и U' *изоморфны* тогда и только тогда, когда $R(U, U')$ истинно. Таким образом, изоморфные доказательства имеют одинаковые дизъюнкты при соответствующих узлах, хотя структуры их могут несущественно различаться. Заметим, что если U и U' изоморфны, то они имеют одинаковую глубину.

2.6. Процедура, основанная на абстрактных доказательствах

Мы приведем неполную процедуру доказательства теорем, основанную на абстракциях. Пусть f — функция абстракции, определенная на множестве дизъюнктов S, T — доказательство C' из S и $D' \in f(C')$. Из теоремы 2.5 нам известно, что существует

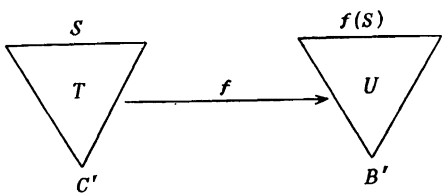


Рис. 2.

доказательство U из $f(S)$ некоторого дизъюнкта B', поглощающего D'. Нередко оказывается, что T и U имеют одну и ту же форму (рис. 2). Поэтому один способ поиска доказательств C' из S заключается в поиске доказательств U такого дизъюнкта B' из $f(S)$ и попытке найти доказательство T дизъюнкта C' из S той же самой формы, что и U. Доказательства из $f(S)$ являются «абстрактными доказательствами», помогающими найти доказательства из S.

Мы точно определим это отношение между T и U и используем его в качестве основы для программы доказательства теоремы «ndfind». К сожалению, данный метод оказывается неполным. Мы проиллюстрируем, почему это происходит, а затем в разд. 3 введем мультидизъюнкты и m-резолюции, для которых имеется простая и полная версия «ndfind».

Определение. Предположим, что f — функция абстракции. Пусть $R1(B, C)$ есть отношение «$B \in f(C)$», и пусть $R2(B, C)$ есть отношение «B поглощает некоторый элемент из $f(C)$». Мы

используем запись $T \to_f U$ в качестве сокращения для $(R1; R2)(U, T)$. Пример 1 из раздела 2.4 дает доказательство T, имеющее такую абстракцию U, что $T \to_f U$, где f — пропозициональная абстракция. Пример 2 из того же раздела дает доказательство T, не имеющее абстракции U, для которой $T \to_f U$.

Процедура «ndfind» по данному абстрактному доказательству U пытается найти доказательство T, такое что $T \to_f U$. Пусть S — множество дизъюнктов, а U — доказательство из $f(S)$. Вместе с каждым узлом N в U, «ndfind» хранит множество clauses(N) дизъюнктов C, обладающих следующим свойством: существуют начальное поддоказательство $U1$ доказательства U и минимальное доказательство $T1$ из S, такие что

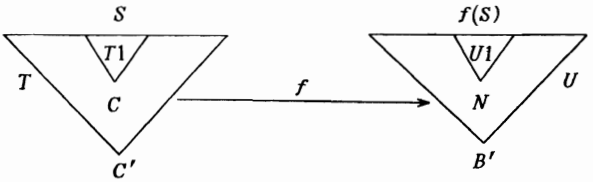


Рис. 3.

$T1 \to_f U1$, $C =$ Result $(T1)$ и N — единственный конечный узел $U1$. Заметим, что C выводим из S с помощью резолюции (рис. 3).

procedure ndfind(U, S, f);
[[допустим, что для всех начальных узлов N из U clauses$(N) = \{c \in S$: label$(N) \in f(C)\}$ и clauses$(N) = \varnothing$ для всех узлов N из U, не являющихся начальными]]
loop while (*there exist узлы* $N1, N2, N$ *из* U и дизъюнкты $C1$, $C2$, C *such that*
(1) $\langle N1, N2, N\rangle \in \mathrm{Res}(U)$
(2) $C1 \in$ clauses$(N1)$ и $C2 \in$ clauses$(N2)$
(3) C является резольвентой $C1$ и $C2$
(4) $C \notin$ clauses(N)
(5) label(N), поглощает некоторый элемент из $f(C)$);
добавить C к clauses(N)
repeat;
end ndfind;

Пусть Z — резолюционное доказательство, порожденное процедурой «ndfind». Нетрудно показать, что Z — наименьшее (с точностью до изоморфизма) доказательство, удовлетворяющее следующему условию.

Если $U1$ — начальное поддоказательство U, а $W1$ — доказательство из S, такое что $W1 \to_f U1$, то $W1$ изоморфно некоторому начальному поддоказательству для Z.

Вполне возможно, что Z имеет начальные поддоказательства, абстракцией которых является U, и, кроме того, несколько начальных поддоказательств с абстракциями, являющимися начальными поддоказательствами для U. Заметим, что мы легко могли бы написать более общий вариант «ndfind» для нахождения всех доказательств T, таких что $(R1; R2)$ (U, T) для произвольных вычислимых отношений между дизъюнктами $R1$ и $R2$.

Предположим, что мы ищем доказательство дизъюнкта C' из множества дизъюнктов S, используя абстракцию f. С этой целью мы выбираем D' в $f(C)$ и выполняем следующие шаги для $d = 1, 2, 3 \ldots$ до тех пор, пока не будет найдено доказательство дизъюнкта C'.

(1) Пусть V — множество всех резолюционных доказательств из $f(S)$ глубины d.

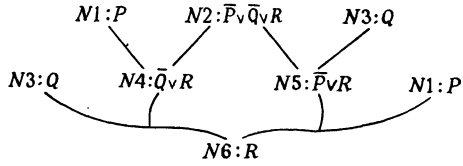
(2) Пусть $V1$ — начальное поддоказательство V, содержащее все резолюции, которые порождают в доказательствах глубины d дизъюнкты, поглощающие D'. Заметим, что $V1$ может иметь более одного конечного узла.

(3) Образуем clauses(N) для узлов N из $V1$, как это требует процедура «ndfind».

(4) ndfind$(V1, S, f)$.

Мы назовем эту процедуру «proofsearch 1». Заметим, что «proofsearch 1» неполна.

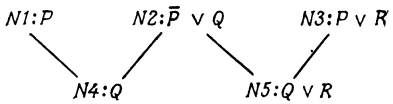
Сейчас мы дадим пример использования «ndfind» в «proofsearch 1». Пусть множество входных дизъюнктов S есть $\{\bar{P}(x) \vee \vee \bar{Q}(x) \vee R(x),\ P(x),\ Q(a)\}$ и f — пропозициональная абстракция. Предположим, что мы ищем доказательство $R(a)$ из S. Мы порождаем доказательства из $f(S)$ некоторого дизъюнкта, поглощающего R, и затем применяем «ndfind». Заметим, что $f(S) = \{\bar{P} \vee \bar{Q} \vee R,\ P,\ Q\}$. Полагая, что абстрактное пространство полностью просмотрено до глубины 2, мы получаем следующее абстрактное доказательство U:



Узлы $N1$ и $N3$ нарисованы дважды для большей ясности. Чтобы использовать «ndfind», мы сначала положим clauses$(N1) = \{P(x)\}$, clauses$(N2) = \{\bar{P}(x) \vee \bar{Q}(x) \vee R(x)\}$, clauses$(N3) = \{Q(a)\}$ и clauses$(N4)$, clauses$(N5)$ и clauses$(N6)$ равными $\varnothing$ (пустому множеству). Когда «ndfind» закончит ра-

боту, она добавит $\bar{Q}(x) \vee R(x)$ к clauses($N4$), $\bar{P}(a) \vee R(a)$ к clauses($N5$) и дважды добавит $R(a)$ к clauses($N6$). Процедура «ndfind» не полностью определяет порядок, в котором эти шаги будут выполнены. Так, $R(a)$ будет получен резолюциями из S двумя различными способами. Позднее мы обсудим, как абстрактный поиск может быть уменьшен применением различных резолюционных стратегий, таких как лок-резолюция [2] или всегда-положительная (all-positive) резолюция [11]. Так как в любом случае процедура «ndfind» неполна, мы могли бы также использовать лок-резолюцию или упорядочение предикатных символов [14], или какую-либо другую подобную стратегию в абстрактном пространстве. Однако идея удалять тавтологии из абстрактного пространства или удалять дизъюнкты, имеющие в абстрактном пространстве поддизъюнкты, по-видимому, не совсем удачна.

Следующий пример показывает, почему «ndfind» неполна, а также мотивирует использование мультидизъюнктов, которые будут обсуждаться в разд. 3. Пусть множество входных дизъюнктов S есть $\{\bar{P}(a) \vee \bar{P}(b) \vee Q(c),\ P(b) \vee R(d),\ P(a)\}$, и мы применяем пропозициональную абстракцию. Предположим, что мы ищем доказательство $Q(c) \vee R(d)$ из S. Мы получаем следующее абстрактное доказательство U глубины 1:



Когда выполняется «ndfind», она порождает $\bar{P}(b) \vee Q(c)$ при $N4$ и $\bar{P}(a) \vee Q(c) \vee R(d)$ при $N5$. Однако дизъюнкт $Q(c) \vee R(d)$ порожден не будет. На глубине 2 «ndfind» также не породит $Q(c) \vee R(d)$, несмотря на то что он может быть получен простым доказательством глубины 2 из S.

Заметим, что причиной неполноты «ndfind» является тот факт, что две различные литеры в дизъюнкте при абстракции могут отображаться на одну и ту же литеру. Следовательно, уменьшение вероятности отображения различных литер на одну и ту же литеру увеличивает вероятность того, что «ndfind» будет полной. Впоследствии мы приведем процедуру «ndfindm» — полную версию «ndfind», основанную на мультидизъюнктах. Однако «ndfindm» является более сложной по сравнению с «ndfind», поскольку мультидизъюнкты могут содержать более одного вхождения произвольной литеры. Поэтому, возможно, имеет смысл использовать «ndfind» вместе с тщательно выбранными абстракциями, несмотря на то что «ndfind» неполна.

2.7. Задача о множестве степени

Сейчас мы приведем более сложный и более реалистический пример и укажем абстракцию, которая представляется подходящей для этого примера. Задача состоит в том, чтобы доказать, что множество степени множества $x \cap y$ есть пересечение множества степени x и множества степени y для любых множеств x и y. Данная задача взята из [1]. Пусть $P(x)$ обозначает множество степени x (т. е. множество всех подмножеств x), $\subset$ — отношение включения, $\cap$ — пересечение множеств, $\in$ — отношение принадлежности и $=$ — равенство множеств. Кроме того, для данных множеств x и y пусть $g(x, y)$ обозначает элемент, принадлежащий x, но не принадлежащий y, в случае когда такой элемент существует. Мы имеем следующее множество S входных дизъюнктов:

$A1$	$x = y \supset (x \subset y)$
$A2$	$x = y \supset (y \subset x)$
$A3$	$(x \subset y) \wedge (y \subset x) \supset x = y$
$A4$	$(x \subset y) \wedge (z \in x) \supset z \in y$
$A5$	$g(x, y) \in x \vee (x \subset y)$
$A6$	$g(x, y) \notin y \vee (x \subset y)$
$A7$	$x \in P(y) \supset (x \subset y)$
$A8$	$(x \subset y) \supset x \in P(y)$
$A9$	$x \in y \cap z \supset x \in y$
$A10$	$x \in y \cap z \supset x \in z$
$A11$	$x \in y \wedge x \in z \supset x \in y \cap z$
$A12$	$P(A \cap B) \neq P(A) \cap P(B)$

Абстракцию для этого примера следует выбирать осторожно. Если бы, к примеру, мы избрали семантическую абстракцию с областью {0, 1, 1, 3, 4}, то дизъюнкт $A4$ имел бы 125 абстракций $i \supset j \wedge k \in i \supset k \in j$ для $i, j, k \in \{0, 1, 2, 3, 4\}$. У всей системы тогда могло бы быть свыше 500 абстрактных входных дизъюнктов. И хотя это число может быть значительно уменьшено с помощью метода, приводимого ниже, данный пример показывает, что следует избегать слишком больших абстрактных пространств.

Пусть f — семантическая абстракция с областью $D = \{P, R, \cap\}$. Значения термов задаются следующим образом: $g(x, y) \to R$ для всех x, y, $A \to R$, $B \to R$, $x \cap y \to \cap$ для всех x, y и $P(x) \to P$ для всех x. Таким образом, данная абстракция выбирает самый важный оператор терма, заменяя A, B и g на R Это объясняется тем, что $\cap$ и P, по-видимому, наиболее важные функциональные символы для данной задачи, а все остальные

мы обозначаем символом R, представляющим произвольное множество. Эта функция абстракции дает одну абстракцию для $A12$, по 9 абстракций для $A1$, $A2$, $A3$, $A5$, $A6$, $A7$, $A8$, $A9$ и $A10$ и по 27 абстракций для $A4$ и $A11$ — всего 136 абстракций в $f(S)$. Например, абстракциями для $x = y \supset (x \subset y)$ являются $R = R \supset (R \subset R)$, $R = \bigcap \supset (R \subset \bigcap)$, $R = P \supset (R \subset P)$ и т. д. Легко видеть, однако, что переменные верхнего уровня можно оставлять в этой абстракции без изменений, т. е. для получения абстракции дизъюнкта C мы оставляем все переменные верхнего уровня из C такими, какие они есть, и заменяем все остальные термы на их внешние функциональные символы, кроме g, A и B, которые на верхнем уровне заменяются на R. В результате каждый дизъюнкт имеет в точности одну абстракцию и все абстрактное пространство содержит 12 входных дизъюнктов. Эти дизъюнкты имеют следующий вид:

$B1$ $\quad x = y \supset (x \subset y)$

$B2$ $\quad x = y \supset (y \subset x)$

$B3$ $\quad (x \subset y) \wedge (y \subset x) \supset x = y$

$B4$ $\quad (x \subset y) \wedge (z \in x) \supset z \in y$

$B5$ $\quad R \in x \vee (x \subset y)$

$B6$ $\quad R \notin y \vee (x \subset y)$

$B7$ $\quad x \in P \supset (x \subset y)$

$B8$ $\quad (x \subset y) \supset x \in P$

$B9$ $\quad (x \in \bigcap) \supset x \in y$

$B10$ $\quad (x \in \bigcap) \supset x \in z$

$B11$ $\quad (x \in y) \wedge (x \in z) \supset (x \in \bigcap)$

$B12$ $\quad P \neq \bigcap$.

Так как дизъюнкты $B9$ и $B10$ — варианты, следует сохранить только один из них и считать его абстракцией $A9$ и $A10$. Поэтому абстрактное пространство будет содержать только 11 входных дизъюнктов. Абстракцию f можно рассматривать как композицию $f_2 f_1$ абстракций f_1 и f_2, где f_1 вычеркивает все аргументы у всех функциональных символов, а f_2 заменяет g, A и B на R. С этой точки зрения естественно, что переменные верхнего уровня должны оставаться неизменными под действием f и для каждого дизъюнкта C множество $f(C)$ должно состоять из одного дизъюнкта.

Теперь мы приведем доказательство $P(x \cap y) \subset P(x) \cap P(y)$ из исходных аксиом и соответствующее абстрактное доказательство. Для доказательства $P(x \cap y) = P(x) \cap P(y)$ необходимо

будет также доказать $P(x) \cap P(y) \subset P(x \cap y)$ и использовать *A*3.
Доказательство из исходных аксиом:

1.	$u \in x \wedge x \in P(w) \supset u \in w$	***A*4, *A*7**
2.	$u \in x \wedge x \in P(y \cap z) \supset u \in y$	1, *A*9
3.	$u \in x \wedge x \in P(y \cap z) \supset u \in z$	1, *A*10
4.	$x \in P(y \cap z) \supset g(x, w) \in y \vee x \subset w$	2, *A*5
5.	$x \in P(y \cap z) \supset x \subset y$	4, *A*6
6.	$x \in P(y \cap z) \supset g(x, w) \in z \vee x \subset w$	3, *A*5
7.	$x \in P(y \cap z) \supset x \subset z$	6, *A*6
8.	$x \in P(y \cap z) \supset x \in P(y)$	5, *A*8
9.	$x \in P(y \cap z) \supset x \in P(z)$	7, *A*8
10.	$x \in P(y \cap z) \supset x \notin w \vee x \in P(y) \cap w$	8, *A*11
11.	$x \in P(y \cap z) \wedge x \in P(u \cap v) \supset x \in P(y) \cap P(v)$	9, 10
12.	$g(P(y \wedge z), w) \in P(y) \cap P(z) \vee P(y \cap z) \subset w$	11, *A*5
13.	$P(y \cap z) \subset P(y) \cap P(z)$	12, *A*6

Абстрактное доказательство:

1.	$u \in x \wedge x \in P \supset u \in w$	*B*4, *B*7
2.	$u \in x \wedge x \in P \supset u \in y$ (Вариант 1)	1, *B*9
3.	$u \in x \wedge x \in P \supset u \in z$ (Вариант 1)	1, *B*10
4.	$x \in P \supset R \in y \vee x \subset w$	2, *B*5
5.	$x \in P \supset x \subset w \vee z \subset y$	4, *B*6
6.	(Вариант 4)	3, *B*5
7.	(Вариант 5)	6, *B*6
8.	$x \in P \supset x \in P$	5, *B*8
9.	(Такой же, как и 8)	7, *B*8
10.	$x \in P \supset x \notin w \vee x \in \bigcap$	8, *B*11
11.	$x \in P \supset x \in \bigcap$	9, 10
12.	$R \in \bigcap \vee P \subset w$	11, *B*5
13.	$P \subset w \vee z \subset \bigcap$	12, *B*6

В данном примере каждое абстрактное доказательство соответствует некоторому плану доказательства. Например, шаг 8 абстрактного доказательства означает: «Доказать, что если x содержится в множестве степени некоторого множества, то x содержится в множестве степени какого-либо другого множества». Шаг 11 означает: Доказать, что если x содержится в множестве степени какого-либо множества, то x содержится в пересечении двух множеств». Поэтому единственными доказа-

тельствами, построенными из $A1, \ldots, A12$, будут доказательства, которые соответствуют некоторому такому плану. Представляется вероятным, что, следуя подобным планам доказательства, мы значительно сократим размеры поискового пространства.

Процедура «ndfind» не указывает, в каком порядке следовать планам доказательства. Нетрудно написать целенаправленную версию «ndfind», которая использует поиск по глубине и предпочитает выполнять резолюции на как можно большей глубине. Здесь, однако, мы не будем обсуждать такие методы.

2.8. Ложные доказательства

Отметим, что может встретиться много абстрактных доказательств, которые не соответствуют ни одному доказательству в исходном пространстве. Мы назовем такие абстрактные доказательства «ложными доказательствами» и ниже обсудим методы их устранения. Ложные доказательства никогда не являются причиной вывода неверного заключения, но они увеличивают размеры поискового пространства. Примером ложного доказательства является следующее:

1.	$x \in Y \wedge x \in Z \supset x \in \bigcap$	Абстракция $A11$
2.	$R \notin Y \vee X \subset Y$	Абстракция $A6$
3.	$R \in Y \wedge R \in Z \supset X \subset \bigcap$	1, 2
4.	$R \in X \vee X \subset Y$	Абстракция $A5$
5.	$Z \subset \bigcap \vee X \subset Y$	3, 4
6.	$Z \in \bigcap \supset Z \in Y$	Абстракция $A9$
7.	$R \in \bigcap \supset X \subset Y$	2, 6
8.	$X \subset Y \vee \bigcap \subset Y$	4, 7
9.	$(X \subset Y) \wedge (Y \subset X) \supset X = Y$	Абстракция $A3$
10.	$X \subset \bigcap \supset X = \bigcap$	8, 9
11.	$X = \bigcap$	5, 10
12.	$P \neq \bigcap$	Абстракция $A12$
13.	NIL	11, 12

Один метод устранения ложных доказательств связан с понятием множества поддержки [15]. Другие методы мы обсудим позднее. Идея состоит в том, чтобы следить, какие входные дизъюнкты могут использоваться в абстрактном доказательстве. Предположим, что U — абстрактное доказательство, и мы ищем доказательство NIL из S. Рассмотрим множество дизъюнктов S_n, абстракции которых используются в качестве входных дизъюнктов в U. Если S_n непротиворечиво, то U обя-

зано быть ложным доказательством. Нам может быть известно, что некоторые входные дизъюнкты обязаны появиться в доказательстве. Поэтому мы можем не рассматривать абстрактные доказательства, которые не содержат все эти существенные входные дизъюнкты. Заметим, что при таком подходе можно использовать несколько множеств поддержки одновременно. Пусть, к примеру, S_i — подмножества множества входных дизъюнктов S, такие что $S - S_i$ непротиворечивы. Если $S_u \cap S_i$ при некотором i пусто, то U — ложное доказательство. Из этого вовсе не следует, что U можно отбросить, поскольку леммы из U могут участвовать в других абстрактных доказательствах. Абстрактные дизъюнкты могут быть отброшены, если они участвуют только в ложных доказательствах.

2.9. Выбор абстракции

Следующие общие принципы помогают решить, какие абстракции наиболее полезны.

(1) Абстрактное поисковое пространство должно быть мало.

(2) Вероятность того, что абстракцией различных литер дизъюнкта является одна и та же литера, должна быть мала.

Довод в пользу (1) — это стремление упростить поиск в абстрактном пространстве. В пользу (2) имеется несколько аргументов. Если различные литеры отображаются, как правило, на различные литеры, то число ложных доказательств будет уменьшаться. Кроме того, число доказательств T, для которых $T \rightarrow_f U$, также будет уменьшаться, что должно сделать процедуру «proofsearch 1» эффективнее, и к тому же более вероятно, что «proofsearch 1» станет полной. Наконец, удаление тавтологий в абстрактном пространстве вряд ли приведет к уничтожению полезных доказательств. Однако требования (1) и (2) в некотором смысле противоречат друг другу. Если абстрактное пространство делать меньше, то вероятность отображения различных литер на одну и ту же литеру будет возрастать. Позднее мы увидим, как одновременное применение нескольких абстракций может до некоторой степени примирить эти требования. Для того чтобы получить полные стратегии, использующие более одной абстракции, нам потребуются m-абстракции и m-дизъюнкты, которые будут введены в разд. 3.

Другой способ примирить требования (1) и (2) — это использование последовательности абстракций, каждая из которых лишь немного ограничивает поиск. Допустим, что мы ищем доказательство NIL из S. Если $f_1, f_2, \ldots, f_k$ — абстракции, мы можем найти доказательства NIL из $f_1 f_2, \ldots, f_k(S)$, затем отобразить их на доказательства NIL из $f_1 f_2, \ldots, f_{k-1}(S)$ и т. д. до тех пор, пока не получим доказательства NIL из S. Если

каждое отображение f_i лишь немного изменяет дизъюнкты, требование (2) будет удовлетворено. Хотя поисковые пространства доказательств из $f_1 \ldots f_j(S)$ могут быть большими, эти доказательства будут найдены с использованием доказательств из $f_1 \ldots f_{j+1}(S)$, и поэтому на каждом шаге поиск будет требовать небольших усилий. Мы применяем нечто подобное «непрерывной деформации» доказательств из $f_1 f_2 \ldots f_k(S)$ для получения доказательств из S. Один из возможных вариантов выбора абстракций заключается в том, чтобы брать отображения f_i, вычеркивающие по одному аргументу у одного функционального или предикатного символа. Когда все аргументы будут вычеркнуты, мы получим пропозициональную абстракцию $f_1 f_2 \ldots f_k$. Подобное использование нескольких уровней абстракции представляется одним из наиболее обещающих методов доказательства теорем, основанных на абстракции. Тот факт, что нужные последовательности абстракций легко подбираются, увеличивает привлекательность данного метода. Приспособленный к m-абстракциям, этот подход дает полную стратегию.

2.10. Логические следствия

Программу «proofsearch 1» можно модифицировать так, чтобы она проверяла, является ли дизъюнкт C' *логическим следствием* множества дизъюнктов S. Этого можно добиться на основании следующего факта [6]. Если C' — логическое следствие S, то существует дизъюнкт C'', выводимый с помощью резолюции из S и поглощающий C'. Кроме того, если D' — абстракция C', то по свойству абстракций (3) имеется абстракция D'' дизъюнкта C'', поглощающая D'. Поэтому существует доказательство из $f(S)$ дизъюнкта B', поглощающего D'', откуда B' поглощает D'. Следовательно, если мы породим все абстрактные доказательства дизъюнктов B', поглощающих D', и применим «ndfind», мы можем породить доказательство некоторого дизъюнкта, поглощающего C', и тогда C' — логическое следствие S. Данный метод, однако, неполный. Позднее мы приведем полные стратегии для решения рассматриваемой задачи.

3. m-РЕЗОЛЮЦИИ И m-АБСТРАКЦИИ

Мы теперь распространим понятие абстракции на «мультидизъюнкты», которые являются мультимножествами литер, т. е. допускается более одного вхождения литеры в мультидизъюнкт. Это дополнительное усложнение позволяет получать более простые стратегии, чем для обычных дизъюнктов. Понятие абстракции обобщается на мультидизъюнкты для получения функций «m-абстракции», которые обладают свойствами, похожими на

свойства функций абстракции. Мы приведем несколько примеров функций m-абстракции и дадим общий метод их образования. Одно из преимуществ функций m-абстракции и мультидизъюнктов состоит в том, что в поиске доказательства довольно естественным образом можно использовать совместно несколько таких функций. Мы введем также «ограниченные мультидизъюнкты» и относящиеся к ним стратегии и абстракции. Преимущество ограниченных мультидизъюнктов заключается в том, что размеры абстрактного поискового пространства для них меньше, чем для обычных мультидизъюнктов. На самом деле в некоторых полезных специальных случаях абстрактные поисковые пространства конечны. Наконец, мы представим дополнительные методы получения функций m-абстракции (и соответствующих функций для ограниченных m-дизъюнктов).

Определение. Мультимножество M — это множество S вместе с функцией g, отображающей S в множество целых положительных чисел. Мы обозначаем S через $\mathrm{Set}(M)$ и для любого $x \in S$, $g(x)$ обозначаем через $\mathrm{mult}(x, M)$. Если $x \notin S$, то по определению $\mathrm{mult}(x, M) = 0$.

Интуитивно мультимножество — это множество, в которое элементы могут входить по нескольку раз. Для каждого $x \in S$ $\mathrm{mult}(x, M)$ говорит, сколько именно раз x входит в M. Для записи M мы используем обозначение $\{n_1 * x_1, \ldots, n_k * x_k\}$, где $\mathrm{mult}(x, M) = \sum_{j,\ x_j = x} n_j$. Вместо $1 * x$ мы пишем просто x.

Мы часто будем рассматривать обыкновенное множество A как мультимножество M, в которое каждый элемент из A входит ровно по одному разу и которое других элементов не содержит.

Размер $|M|$ мультимножества $M = \{n_1 * x_1, \ldots, n_k * x_k\}$ по определению есть $\sum_{i=1}^{k} n_i$.

Определение. Если $M1$ и $M2$ суть мультимножества, то их *объединение* $M1 \cup M2$ определяется равенством

$$\mathrm{mult}(x, M1 \cup M2) = \mathrm{mult}(x, M1) + \mathrm{mult}(x, M2),$$

пересечение $M1 \cap M2$ — равенством

$$\mathrm{mult}(x, M1 \cap M2) = \min(\mathrm{mult}(x, M1),\ \mathrm{mult}(x, M2)).$$

Разность $M1 - M2$ определяется равенством

$$\mathrm{mult}(x, M1 - M2) = \max(0, \mathrm{mult}(x, M1) - \mathrm{mult}(x, M2)).$$

Иногда мы пишем $\cup$ вместо $\cup$. Заметим, что

$$\mathrm{Set}(M1 \cup M2) = \mathrm{Set}(M1) \cup \mathrm{Set}(M2).$$

Определение. Если $M1$ и $M2$ — мультимножества, то мы пишем $M1 \subset M2$ ($M1$ — подмультимножество $M2$), если для всех x $\mathrm{mult}(x, M1) \leqslant \mathrm{mult}(x, M2)$.

Определение. Если M — мультимножество и g — отображение из $\mathrm{Set}(M)$ в множество N, то $g(M) = \dot{\bigcup}_{x \in M} \{g(x)\}$ т. е. $\mathrm{mult}(y, g(M)) = \sum_{x,\, g(x)=y} \mathrm{mult}(x, M)$ и $|g(M)| = |M|$.

Заметим, что для мультимножеств $M1$ и $M2$ $g(M1 - M2) = g(M1) - g(M2)$, если $M2 \subset M1$. Обычные множества этим свойством не обладают.

Определение. *Мультидизъюнкт* (или m-дизънкт) — это мультимножество литер, т. е. вместе с каждой литерой дизъюнкта хранится ее сложность — целое положительное число, говорящее, сколько раз данная литера входит в мультидизъюнкт. Для записи мультидизъюнктов мы дублируем каждый его элемент столько раз, сколько вхождений он имеет в этот мультидизъюнкт. Так, $\{P, P, Q\}$ — мультидизъюнкт, в котором сложность литеры P равна 2, а сложность литеры Q равна 1.

Определение. Если C — мультидизъюнкт, а α — подстановка, то $C\alpha$ есть $\{L\alpha\colon L \in C\}$, где $L\alpha$ повторяется необходимое число раз, т. е. $\mathrm{mult}(L1, C\alpha) = \sum_{L \in C,\, L\alpha = L1} \mathrm{mult}(L, C)$. Поэтому $|C\alpha| = |C|$, и если $C = \{L_1, L_2, \ldots, L_n\}$, то $C\alpha = \{L_1\alpha, L_2\alpha, \ldots, L_n\alpha\}$.

Пример. Пусть $C = \{\bar{P}(x), \bar{P}(c), Q(x)\}$, а α есть $\{x \leftarrow c\}$, т. е. α заменяет x на c. Тогда $C\alpha = \{\bar{P}(C), \bar{P}(c), Q(c)\}$. Заметим, что литера $\bar{P}(c)$ дважды входит в $C\alpha$.

Определение. Предположим, что $C1$ и $C2$ — мультидизъюнкты, $A1 \subset C1$ и $A2 \subset C2$ (т. е. число вхождений каждой литеры в $A1$ не превосходит числа ее вхождений в $C1$; то же самое верно для $A2$ и $C2$). Предположим далее, что существуют подстановки $\alpha1$ и $\alpha2$, такие что для некоторой литеры L $\mathrm{Set}(A1\alpha1) = \{L\}$ и $\mathrm{Set}(A2\alpha2) = \{\bar{L}\}$. Пусть $\alpha1$ и $\alpha2$ — наиболее общие такие подстановки. Тогда $(C1 - A1)\alpha1 \dot{\cup} (C2 - A2)\alpha2$ является *m-резольвентой* $C1$ и $C2$. (Вспомните определение $C1 - A1$ и $C2 - A2$ для мультимножеств $C1$ и $A1$, $C2$ и $A2$.)

Примеры. Пусть $C1 = \{\bar{P}(a), \bar{P}(x)\}$ и $C2 = \{P(a)\}$. Тогда $\{\bar{P}(x)\}$, $\{\bar{P}(a)\}$ и NIL (пустое мультимножество) являются m-резольвентами $C1$ и $C2$.

Пусть $C1 = \{\bar{P}, \bar{P}\}$ и $C2 = \{P, Q, Q\}$. Тогда m-резольвентами $C1$ и $C2$ являются $\{\bar{P}, Q, Q\}$ и $\{Q, Q\}$.

Обычные дизъюнкты можно рассматривать как мультидизъюнкты, в которых сложность каждой литеры из данного дизъ-

юнкта равна 1. Мы имеем следующие результаты, связывающие m-резолюцию с обычной резолюцией.

Теорема 3.1. *Пусть* $C3$ *— обычная резольвента дизъюнктов* $C1$ *и* $C2$, *а* $D1$ *и* $D2$ *суть* m-*дизъюнкты, такие что* $\mathrm{Set}(D1) = C1$, $\mathrm{Set}(D2) = C2$. *Тогда существует такая* m-*резольвента* $D3$ *от* m-*дизъюнктов* $D1$ *и* $D2$, *что* $\mathrm{Set}(D3) = C3$.

Теорема 3.2. *Предположим, что дизъюнкт* C *выводится из множества дизъюнктов* S *обычной резолюцией. Тогда существует* m-*дизъюнкт* D, *выводимый из* S m-*резолюцией и такой, что* $\mathrm{Set}(D) = C$. (*В выводе* D *мы рассматриваем дизъюнкты из* S *как* m-*дизъюнкты.*) *Заметим, что если* $C = \mathrm{NIL}$, *то и* $D = \mathrm{NIL}$.

Теорема 3.3. *Предположим, что* m-*дизъюнкт* $D3$ *является* m-*резольвентой* m-*дизъюнктов* $D1$ *и* $D2$. *Тогда некоторая обычная резольвента* $\mathrm{Set}(D1)$ *и* $\mathrm{Set}(D2)$ *поглощает* $\mathrm{Set}(D3)$.

Теорема 3.4. *Пусть* S *— множество дизъюнктов и* m-*дизъюнкт* D *выводим из* S m-*резолюцией.* (*В этом выводе мы считаем дизъюнкты из* S m-*дизъюнктами.*) *Тогда существует обычный дизъюнкт* C, *выводимый из* S *обычной резолюцией, причем* C *поглощает* $\mathrm{Set}(D)$.

Определение. m-*абстракция* есть отображение f мультидизъюнктов на (обычные) множества мультидизъюнктов, удовлетворяющее следующим свойствам:

(1) Если $C3$ есть m-резольвента $C1$ и $C2$, и $D3 \in f(C3)$, то существуют m-дизъюнкты $D1 \in f(C1)$ и $D2 \in f(C2)$, такие что $D3$ является примером некоторой m-резольвенты $D1$ и $D2$.

(2) $f(\mathrm{NIL}) = \{\mathrm{NIL}\}$.

Обратим внимание, насколько упростились свойства m-абстракций по сравнению со свойствами обычных абстракций. Следующие два результата, аналогичные теоремам 2.1 и 2.2 для обычных абстракций, показывают, что и m-абстракции также можно легко строить.

Теорема 3.5. *Пусть* Φ *— отображение литер в литеры. Расширим* Φ *до отображения мультидизъюнктов в мультидизъюнкты следующим образом*:

$$\Phi(\{n_1 * L_1,\ n_2 * L_2,\ \ldots,\ n_k * L_k\}) = \\ = \{n_1 * \Phi(L_1),\ n_2 * \Phi(L_2),\ \ldots,\ n_k * \Phi(L_k)\}$$

Таким образом, $|\Phi(C)| = |C|$, *хотя* Φ *не обязано быть взаимно однозначным. Пусть, далее,* Φ *удовлетворяет следующим свойствам*:

(1) $\Phi(\bar{L}) = \overline{\Phi(L)}$ *для всех литер* L.

(2) *Если мультидизъюнкт D — пример мультидизъюнкта C, то $\Phi(D)$ — пример $\Phi(C)$.*

Тогда отображение f, определенное на мультидизъюнктах равенством $f(C) = \{\Phi(C)\}$, является функцией m-абстракции.

Доказательство подобно доказательству теоремы 2.1.

Теорема 3.6. *Пусть F — множество отображений литер в литеры. Каждое отображение $\Phi \in F$ расширим до отображения мультидизъюнктов в мультидизъюнкты так же, как и в предыдущей теореме. Допустим, что для всех $\Phi \in F$ выполнено равенство $\Phi(\bar{L}) = \overline{\Phi(L)}$, какую бы литеру L мы ни взяли, и если мультидизъюнкт D является примером мультидизъюнкта C, то для любого $\Phi 2 \in F$ существует отображение $\Phi 1 \in F$, такое что $\Phi 2(D)$ — пример $\Phi 1(C)$. Определим отображение f на мультидизъюнктах посредством $f(C) = \{\Phi(C)\colon\ \Phi \in F\}$. Тогда f — функция m-абстракции. (Заметим, что $f(C)$ — обычное множество мультидизъюнктов.)*

Доказательство подобно доказательству теоремы 2.2.

Если f — m-абстракция, удовлетворяющая условиям теоремы 3.6, то мы говорим, что f *определена посредством отображений литер.*

Теорема 3.7. *Если f — m-абстракция, определенная посредством отображений литер, то f удовлетворяет также следующему свойству: какие бы мультидизъюнкты $C1$ и $C2$ мы ни взяли, если $C1$ поглощает $C2$, то для каждого $D2 \in f(C2)$ существует $D1 \in f(C1)$, поглощающий $D2$.*

(Мы говорим, что мультидизъюнкт $D1$ поглощает мультидизъюнкт $D2$, если существует такая постановка θ, что $D1\theta \subset D2$, т. е. для любой $L \in D1\theta$ $\mathrm{mult}(L, D1\theta) \leqslant \mathrm{mult}(L, D2)$.) Этот результат будет полезен для получения стратегий, проверяющих, является ли данный m-дизъюнкт логическим следствием некоторого множества m-дизъюнктов.

3.1. Примеры m-абстракций

Представленные ниже m-абстракции получаются из абстракций, приведенных в разд. 2.1, путем повторения каждой литеры необходимое число раз.

(1) *Основная m-абстракция. Если C — m-дизъюнкт $\{L_1, L_2, \ldots, L_k\}$*, то $f(C) = \{\{L_1\theta, L_2\theta, \ldots, L_k\theta\}\colon\ L_i\theta$ — основные литеры при $1 \leqslant i \leqslant k\}$.

(2) *Пропозициональная m-абстракция.* Если C — m-дизъюнкт $\{L_1, L_2, \ldots, L_k\}$, то $f(C) = \{D\}$, где D есть m-дизъюнкт $\{L_1', L_2', \ldots, L_k'\}$, а L_i' получается из L_i вычеркиванием всех

аргументов у предикатного символа, т. е. если L_i есть $P(t_1, \ldots, t_n)$, то L_i' есть P, и если L_i есть $\bar{P}(t_1, \ldots, t_n)$, то L_i' есть $\bar{P}$.

Точно так же мы можем определить m-абстракции, основанные на переименовании предикатных или функциональных символов, изменении знаков литер, изменении порядка или удалении аргументов различных функциональных или предикатных символов. Заметим, что, как и прежде, переименование функциональных и предикатных символов не обязано быть взаимно однозначным, т. е. два различных предикатных или функциональных символа могут быть заменены на один и тот же символ.

(3) *Семантическая m-абстракция.* Допустим, $\mathcal{I}$ — интерпретация множества дизъюнктов над некоторым множеством функциональных и предикатных символов, а D — область интерпретации $\mathcal{I}$. Каждой основной литере L мы сопоставим литеру L' так же, как и в определении семантической абстракции из части 1. Каждому основному m-дизъюнкту $C = \{L_1, \ldots, L_n\}$ мы сопоставим m-дизъюнкт $C' = \{L_1', \ldots, L_n'\}$, где L_i' — литера, сопоставленная литере L_i указанным выше способом. Заметим, что $|C'| = |C|$. Если $C1$ — произвольный m-дизъюнкт, то $f(C1) = \{D: D$ сопоставлен некоторому основному примеру C m-дизъюнкта $C1\}$. Так, если $\mathcal{I}$ — обычная интерпретация целых чисел, то $\{3 \leqslant 3,\ 3 \leqslant 3\}$ — одна из m-абстракций $\{x \leqslant y,\ y \leqslant x\}$. Мы называем f m-абстракцией, полученной из $\mathcal{I}$. Как и прежде, семантические m-абстракции представляются особенно полезными, когда область D конечна, поскольку тогда $f(C)$ — конечное множество мультидизъюнктов при любом C.

Мы можем определить композицию $f_1 f_2$ m-абстракций f_1 и f_2 и показать, как и раньше, что $f_1 f_2$ — это m-абстракция, если таковыми являются f_1 и f_2. Объединение двух m-абстракций также является m-абстракцией. Более того, легко показать, что если f_1 и f_2 — m-абстракции, определенные посредством отображений литер, то их объединение и композиция также определены посредством отображений литер. Для каждой m-абстракции можно определить обратную (если, конечно, она существует). Если m-абстракция имеет обратную, она не приводит к потере какой-либо информации; возможно, такую абстракцию следовало бы назвать m-изоморфизмом.

3.2. m-абстракции m-резолюционных доказательств

Мы сейчас покажем, как m-абстракции могут использоваться для управления поиском доказательства m-дизъюнкта C из множества m-дизъюнктов S. Мы опустим некоторые детали, поскольку ход рассуждений такой же, как и в случае обычных абстракций. Понятия m-абстрактного доказательства, глубины

m-абстрактного доказательства и т. д. определяются аналогично тому, как это сделано для обычной резолюции. Узлы m-абстрактного доказательства V, как и прежде, мы обозначаем через $\mathrm{Nodes}(V)$, а m-резолюции V — через $\mathrm{MRes}(V)$. m-резолюция V — это тройка $\langle N1, N2, N3\rangle$ узлов V, такая что $\mathrm{label}(N1)$ есть m-резольвента m-дизъюнктов $\mathrm{label}(N1)$ и $\mathrm{label}(N2)$. Как и прежде, мы требуем, чтобы $\langle N2, N1, N3\rangle \in \mathrm{MRes}(V)$, если $\langle N1, N2, N3\rangle \in \mathrm{MRes}(V)$. Если $M1$ и $M2$ — отношения между мультидизъюнктами, а $V1$ и $V2$ суть m-резолюционные доказательства, то мы определяем $(M1; M2)(V1, V2)$ так же, как и раньше. Вместо $(M; M)(V1, V2)$ будем писать сокращенно $M(V1, V2)$. Если f — функция m-абстракции и S — множество m-дизъюнктов, то мы пишем $f(S)$ для обозначения множества $\bigcup_{C\in S} f(C)$. Здесь $f(S)$ — обычное множество мультидизъюнктов.

Определение. Пусть T и U — m-резолюционные доказательства, f — функция m-абстракции, $M1(B, C)$ — отношение «$B \in f(C)$» и $M2(B, C)$ — отношение «B имеет пример в $f(C)$». Мы пишем тогда $T \to_f U$, если $(M1; M2)(U, T)$ истинно. Доказательство U мы считаем абстракцией доказательства T. Заметим, что в этом случае T и U будут иметь одинаковую форму и глубину. Кроме того, если T — доказательство из S, то U — доказательство из $f(S)$. Обозначение $T \to_f U$ использовалось также в [9], правда, в несколько ином смысле: доказательство U могло быть гораздо «меньше», чем T, и f могло быть «слабой» абстракцией.

Теорема 3.8. *Пусть T — минимальное m-резолюционное доказательство m-дизъюнкта C' из множества m-дизъюнктов S, а f — функция m-абстракции. Тогда для любого m-дизъюнкта $D' \in f(C')$ существует m-резолюционное доказательство U из $f(S)$, такое что $T \to_f U$, $\mathrm{Result}(U)$ определен и имеет D' своим примером* (рис. 4).

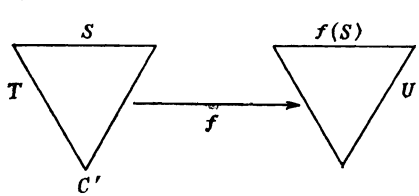
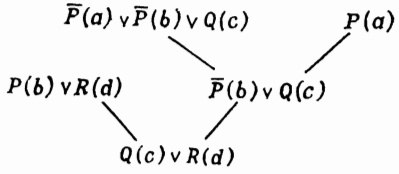


Рис. 4.

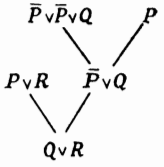
Этот результат гораздо лучше аналогичной теоремы для обычных абстракций. Заметим, что каждой абстракции D' m-дизъюнкта C' соответствует доказательство U — абстракция доказательства T.

Данный результат следующим образом связан с обычной резолюцией. Допустим, что дизъюнкт C' выводим обычной резолюцией из множества дизъюнктов S, а f — функция m-абстракции. Тогда существует такой m-дизъюнкт $C1$, что $\mathrm{Set}(C1) = C'$ и для любого $D' \in f(C1)$ из множества $f(S)$ посредством m-резолюции выводим некоторый дизъюнкт, имеющий D' своим примером. Позднее мы обсудим, как m-абстракции могут оказать помощь в нахождении обычных резолюционных доказательств дизъюнктов, отличных от NIL.

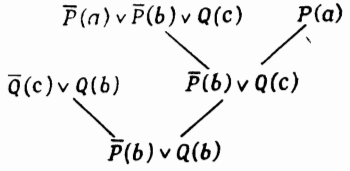
Примеры. (1) Рассмотрим следующее m-резолюционное доказательство (оно является также обычным резолюционным доказательством):



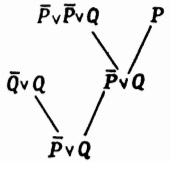
Данный пример совпадает с примером 2 из разд. 2.4. Используя пропозициональную m-абстракцию, мы получим следующее m-абстрактное доказательство:



(2) Рассмотрим следующее m-резолюционное доказательство, взятое из примера 3 разд. 2.4 (оно также является обычным резолюционным доказательством):



Используя пропозициональную m-абстракцию, мы получим следующее m-абстрактное m-резолюционное доказательство:



4. ПОЛНАЯ СТРАТЕГИЯ ДЛЯ ОДИНОЧНОЙ m-АБСТРАКЦИИ

Мы определим процедуру «ndfindm», аналогичную процедуре «ndfind» из разд. 2, но для мультидизъюнктов и m-резолюции. Эта процедура использует m-абстрактные доказательства в качестве ориентира в поисках m-резолюционного доказательства. Пусть f — функция m-абстракции, дано m-резолюционное доказательство U из $f(S)$, и мы хотим найти все доказательства T из S, для которых $T \to_f U$. Вместе с каждым узлом N из U мы храним множество m-clauses(N) m-дизъюнктов, выводимых из S m-резолюцией.

procedure ndfindm(U, S', f);
  [[положим m-clauses$(N) = \{C \in S$: label$(N) \in f(C)\}$ для всех начальных узлов N из U и m-clauses$(N) = \varnothing$ для всех остальных N из U]]
  loop
    while (*there exist* узлы $N1$, $N2$, N из U и m-дизъюнкты $C1$, $C2$, C *such that*
      (1) $\langle N1, N2, N\rangle \in \mathrm{M\,Res}(U)$
      (2) $C1 \in m$-clauses(M) и $C2 \in m$-clauses $(N2)$
      (3) C является m-резольвентой $C1$ и $C2$
      (4) $C \notin m$-clauses(N)
      (5) label(N) имеет примером некоторый m-дизъюнкт из множества $f(C)$;
    add C *to* m-clauses(N);
  repeat
end ndfindm

Нетрудно было бы написать более общую версию «ndfindm», которая при любых вычислимых отношениях $M1$ и $M2$ между m-дизъюнктами находила бы все такие доказательства T, для которых $(M1; M2)(U, T)$ истинно. Сейчас, однако, нам будет достаточно только приведенной выше версии этой процедуры.

Когда «ndfindm» существует, для каждого узла N из U m-clauses(N) будет содержать в точности те m-дизъюнкты C, которые обладают следующим свойством: имеются начальное поддоказательство $U1$ доказательства U и минимальное доказательство $W1$ из S, такие что $W1 \to_f U1$, $C = \mathrm{Result}(W1)$ и N — единственный конечный узел $U1$ (рис. 5).

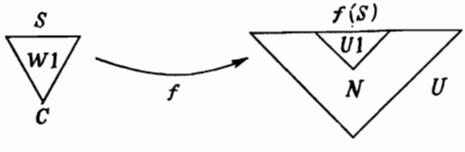


Рис. 5.

Кроме того, если W — какое-либо m-резолюционное доказательство из S, для которого $W \to_f U1$ при некотором начальном поддоказательстве $U1$ доказательства U, то W изоморфно некоторому начальному поддоказательству доказательства, порожденного процедурой «ndfindm».

Пусть T — m-резолюционное доказательство m-дизъюнкта C' из множества m-дизъюнктов S, U — m-резолюционное доказательство из $f(S)$, такое что $T \to_f U$, и пусть U — начальное поддоказательство V. Тогда после выполнения ndfindm(V, S, f) мы получим доказательство C'. (На самом деле эта процедура породит доказательство, изоморфное T.) Заметим, что мы имеем теперь лучшую ситуацию, чем для простых абстракций. Если T имеет глубину d, то и U также будет иметь глубину d. Следовательно, если мы желаем посмотреть, существует ли доказательство m-дизъюнкта C' глубины d, то мы можем выбрать какой-либо m-дизъюнкт D' из $f(C')$ и применить ndfindm(V, S, f), где V содержит все доказательства U из $f(S)$, такие что U имеет глубину d, а D' является примером единственного конечного дизъюнкта U.

На этой идее основана следующая полная стратегия доказательства теорем посредством m-абстракций. Порождение V и $V1$ в данной процедуре аналогично порождению V и $V1$ в «proofsearch 1». Узлы V и $V1$ имеют вид $\langle D1, d_1\rangle$, где $D1$ — m-абстракция некоторого m-дизъюнкта, а d_1 — целое число, обозначающее глубину узла. Если $N = \langle D1, d_1\rangle$, то мы пишем label$(N) = D1$ и depth$(N = d_1$. Доказательство $V1$ строится так, что если U — произвольное минимальное m-резолюционное доказательство из $f(S)$ какого-либо m-дизъюнкта, имеющего D' своим примером, и глубина U равна d, то U изоморфно некоторому начальному поддоказательству $V1$.

procedure proofsearch 2(S, C', f);
[[попытка построить доказательство C' из S, используя функцию m-абстракции f. Полная стратегия доказательства теорем.]]
выбрать D' из $f(C')$;
$S1 \leftarrow \{\langle D, 0:\rangle (\exists C \in S) D \in f(C)\}$;
для всех $\langle D, 0\rangle \in S1$ *do*
  m-clauses$(\langle D, 0\rangle) \leftarrow \{C \in S\colon D \in f(C)\}$ *od*;
for $d = 1$ *to* ∞ *until* C' порожден из S *do* [1])
  [[ищется доказательство C' из S глубины d]] пусть V — наименьшее m-резолюционное доказательство, такое что
  (a) $S1 \subset$ Nodes(V)

---

[1]) Для d от 1 до ∞ до тех пор, пока C' не порожден из S, выполнить ...; *od* (*do* наоборот) обозначает конец оператора *do*. — *Прим. перев.*

(b) Если $\langle B1, d_1\rangle \in \mathrm{Nodes}(V)$, $\langle B2, d_2\rangle \in \mathrm{Nodes}(V)$, $d_1 < d$, $d_2 < d$ и $B3$ является m-резольвентой $B1$ и $B2$, то $\langle B3, d_3\rangle \in \mathrm{Nodes}(V)$ и $\langle\langle B1, d_1\rangle, \langle B2, d_2\rangle, \langle B3, d_3\rangle\rangle \in \mathrm{M\,Res}(V)$, где $d_3 = 1 + \max(d_1, d_2)$;

пусть $V1$ — наименьшее доказательство V, такое что

(a) Если $\langle B', d\rangle \in \mathrm{Nodes}(V)$ и D' — пример B', то $\langle B', d\rangle \in \mathrm{Nodes}(V1)$

(b) Если $\langle N1, N2, N3\rangle \in \mathrm{M\,Res}(V)$ и $N3 \in \mathrm{Nodes}(V1)$, то $N1 \in \mathrm{Nodes}(V1)$, $N2 \in \mathrm{Nodes}(V1)$ и $\langle N1, N2, N3\rangle \in \mathrm{M, Res}(V1)$;

[[Заметим, что V может быть найдено полным перебором, а $V1$ может быть получено удалением узлов и m-резолюцией из V. Возможно, $V1$ получается также применением дополнительных уровней m-абстракции.]]

для всех новых нёначальных узлов N из $V1$ *do* m-clauses$(N) \leftarrow \varnothing$ *od*;

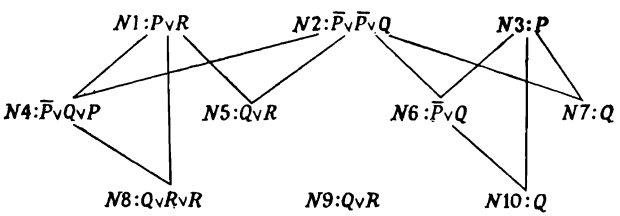
[[все начальные узлы $V1$ будут содержаться в $S1$ и иметь поэтому приписанные им m-дизъюнкты]]

ndfindm(V, S, f)

od:

end proofsearch 2;

Рассмотрим следующий пример. Пусть $S = \{\bar{P}(a) \vee \bar{P}(b) \vee Q(c),\ P(a),\ P(b) \vee R(d)\}$, $C' = Q(c) \vee R(d)$ и f — пропозициональная m-абстракция. Предположим, что мы находимся на глубине $d = 2$. Тогда доказательство V будет иметь следующий вид:



Чтобы не загромождать диаграмму, выводы $N9$ были опущены. Их можно получить двумя способами: из $N3$ и $N4$ и из $N1$ и $N6$. Заметим, что все вхождения $Q \vee R$ на глубине 2 объединены в один узел, но они не объединяются с вхождениями $Q \vee R$ на глубине 1. Более строго, $N9 = \langle Q \vee R,\ 2\rangle$, $N5 = \langle Q \vee R,\ 1\rangle$, $N1 = \langle P \vee R, 0\rangle$ и т. д. Доказательство $V1$ получается из V вычеркиванием узлов $N5$, $N7$, $N8$ и $N10$, поскольку они не дают в доказательстве глубины 2 никакого дизъюнкта, имеющего $Q \vee R$ своим примером. (Дизъюнкт $Q \vee R \vee R$ следовало бы сохранить, если бы мы использовали «proofsearch 2» для проверки выводимости $Q(c) \vee R(d)$ обычной резолюцией, так как

$\mathrm{Set}(\{Q, R, R\}) = \{Q, R\}$ имеет $\{Q, R\}$ своим примером.) Таким образом, в данном примере $V1$ имеет 6 узлов. Перед применением «ndfindm» m-clauses$(N1)$, m-clauses$(N2)$ и m-clauses$(N3)$ будут состоять из $P(b) \vee R(d)$, $\bar{P}(a) \vee \bar{P}(b) \vee Q(c)$ и $P(a)$ соответственно. Для остальных узлов N множества m-clauses(N) будут пустыми. В результате своей работы «ndfindm» породит $\bar{P}(a) \vee \bar{Q}(c) \vee R(d)$ при узле $N4$, $\bar{P}(b) \vee Q(c)$ — при $N6$ и (двумя способами) $Q(c) \vee R(d)$ при $N9$. Приведенный пример показывает одну невыгодную сторону m-дизъюнктов, а именно то, что возможных m-дизъюнктов больше, чем обычных дизъюнктов. До некоторой степени мы компенсируем это неудобство с помощью ограниченных m-дизъюнктов, которые будут обсуждаться в разд. 6. Заметим, что «proofsearch 1» не дает результата в данном примере.

Сейчас мы приведем другой пример использования «proofsearch 2». Этот пример взят из [10], и процедура «proofsearch 1» работала бы в данном случае с таким же успехом. Пример иллюстрирует простую и полезную абстракцию с широкой областью применения — операторную абстракцию. Мы интерпретируем предикатные символы и функции следующим образом. $\mathrm{IN}(x, y)$ означает $x \in y$, $\mathrm{HP}(x, m, y)$ означает, что x имеет m частей типа y, $\mathrm{T}(M, N)$ — это M раз по N и $\mathrm{SK1}(N, M, Z, Y, X)$ — такой объект W, что $\mathrm{IN}(W, Y)$ и $\neg\mathrm{HP}(W, N, Z)$, если он существует. Заметим, что если W не существует, то $(\forall W)[\mathrm{IN}(W, Y) \supset \mathrm{HP}(W, N, Z)]$. Входные дизъюнкты S имеют следующий вид.

A1 IN (ДЖОН, МАЛЬЧИК)

A2 IN $(X,$ МАЛЬЧИК$) \supset \mathrm{IN}$ $(X,$ ЧЕЛОВЕК$)$

A3 $\mathrm{HP}(X, M, Y) \supset \mathrm{IN}(\mathrm{SK1}(N, M, Z, Y, X), Y) \vee \mathrm{HP}(X, \mathrm{T}(M, N), Z)$

A4 $\mathrm{HP}(X, M, Y) \wedge \mathrm{HP}(\mathrm{SK1}(N, M, Z, Y, X), N, Z) \supset \mathrm{HP}(X, \mathrm{T}(M, N), Z)$

A5 IN $(X,$ КИСТЬ$) \supset \mathrm{HP}(X, 5,$ ПАЛЬЦЫ$)$

A6 IN $(X,$ ЧЕЛОВЕК$) \supset \mathrm{HP}(X, 2,$ РУКА$)$

A7 IN $(X,$ РУКА$) \supset \mathrm{HP}(X, 1,$ КИСТЬ$)$

A8 $\neg\mathrm{HP}(\text{ДЖОН}, \mathrm{T}(2, 1),$ КИСТЬ$)$

Пусть f — *операторная* абстракция, отображающая каждый терм на его внешний функциональный символ, а переменные верхнего уровня — на самих себя. Напомним, что абстракцию в примере из разд. 2.7 можно представить композицией операторной абстракции и переименования функциональных символов. Операторная абстракция удобна потому, что $f(C)$ состоит из *единственного* дизъюнкта при любом C, и поэтому абстрактное множество входных дизъюнктов мало. На операторную аб-

стракцию можно смотреть как на вычеркивание всех аргументов у всех функциональных символов.

Множество $f(S)$ выглядит так:

B1 IN(ДЖОН, МАЛЬЧИК)
B2 IN(X, МАЛЬЧИК) $\supset$ IN(X, ЧЕЛОВЕК)
B3 HP(X,M,Y) $\supset$ IN(SK1,Y) $\vee$ HP(X,T,Z)
B4 HP(X,M,Y) $\wedge$ HP(SK1,N, Z) $\supset$ HP (X,T, Z)
B5 IN(X, КИСТЬ) $\supset$ HP(X,5, ПАЛЬЦЫ)
B6 IN(X, ЧЕЛОВЕК) $\supset$ HP(X,2, РУКА)
B7 IN(X, РУКА) $\supset$ HP(X,1, КИСТЬ)
B8 $\neg$HP(ДЖОН, T, КИСТЬ)

Интересно отметить, что если f — операторная абстракция, то проблема непротиворечивости множества $f(S)$ разрешима. Доказательство NIL из $f(S)$ выглядит следующим образом.

1.	IN(ДЖОН, ЧЕЛОВЕК)	B1, B2
2.	HP(ДЖОН, 2, РУКА)	1, B6
3.	IN(SK1, РУКА) $\vee$ HP(ДЖОН, T, Z)	2, B3
4.	IN(SK1, РУКА)	3, B3
5.	HP(SK1,1, КИСТЬ)	4, B7
6.	HP(X,M,Y) $\supset$ HP(X,T, КИСТЬ)	5, B4
7.	HP(ДЖОН, T, КИСТЬ)	2, 6
8.	NIL	7, B8

Соответствующее доказательство NIL из S имеет следующий вид:

1.	IN(ДЖОН, ЧЕЛОВЕК)	A1, A2
2.	HP(ДЖОН, 2, РУКА)	1, A6
3.	IN(SK1(N, 2, Z, РУКА, ДЖОН), РУКА) $\vee$ HP(ДЖОН, T(2,N),7)	2, A3
4.	IN(SK1(1, 2, КИСТЬ, РУКА, ДЖОН), РУКА)	3, A8
5.	HP(SK1(1, 2, КИСТЬ, РУКА, ДЖОН), 1, КИСТЬ)	4, A7
6.	HP(ДЖОН, 2, РУКА) $\supset$ HP(ДЖОН, T(2, 1), КИСТЬ)	5, A4
7.	HP(ДЖОН, T(2, 1), КИСТЬ)	2, 6
8.	NIL	7, A8

Так как m-абстрактные m-дизъюнкты, как правило, проще исходных дизъюнктов, следует ожидать, что построение V в про-

цедуре «proofsearch 2» будет более легким по сравнению с полным перебором для доказательства C из S. Если же непосредственный поиск доказательства из $f(S')$ дизъюнкта, имеющего D' своим примером, все еще слишком сложен, к $f(S)$ можно применить другую функцию m-абстракции и таким образом направить поиск этого доказательства. Развивая эту идею, мы видим, что в поиске доказательства совместно может использоваться любое число «уровней m-абстракции».

Как и прежде, мы не можем проводить удаление тавтологий и наддизъюнктов среди дизъюнктов, порожденных m-резолюцией из $f(S)$. Единственная допустимая стратегия удаления состоит в вычеркивании примеров тех m-дизъюнктов, которые уже были выведены на этой же глубине, запоминая при этом, как именно данные примеры были получены. Другими словами, если $\langle N1, N2, N3\rangle \in \mathrm{Res}(V)$, $\mathrm{label}(N3)$ — пример $\mathrm{label}(N)$ и $N3$ и N расположены на одной глубине в V, то мы можем всюду в $\mathrm{Res}(V)$ заменить $N3$ на N. Таким образом, $\langle N1, N2, N\rangle$, $\langle N2, N1, N\rangle$ и, возможно, некоторые резолюции с N в качестве первой или второй компоненты будут добавлены к $\mathrm{Res}(V)$, а все вхождения $N3$ будут устранены. В исходном пространстве, однако, может использоваться любая полная стратегия доказательства теорем в том случае, когда абстрактное пространство порождается полностью. Например, мы можем так ограничить процедуру «ndfindm» в «proofsearch2», чтобы она порождала резольвенты из S только согласно некоторой полной m-резолюционной стратегии. Поиск из S поэтому будет ограничиваться как m-резолюционной стратегией, так и абстрактным поисковым пространством. Многообещающей стратегией могла бы быть лок-резолюция, должным образом приспособленная к m-дизъюнктам. Если m-абстракция удовлетворяет определенным условиям, то возможно также ограничить и абстрактное поисковое пространство согласно некоторой полной стратегии. Например, если m-абстракция определена посредством отображений литер и сохраняет их знаки, то m-абстракция $P1$-вывода [11] также будет $P1$-выводом, а значит, мы можем использовать $P1$-вывод как в исходном, так и в абстрактном пространствах. В случае, когда должным образом проведена индексация литер для m-абстракций, определенных посредством отображений литер, мы можем также применять лок-резолюцию и в исходном, и в абстрактном пространствах. Наконец, если m-абстракция определена посредством отображений литер Φ, таких что предикатный символ для $\Phi(L)$ совпадает с предикатным символом для L, то в исходном и абстрактном пространствах мы можем использовать m-резолюцию с упорядочением предикатных символов (т. е. все предикатные символы линейно упорядочены, и литеры, устраняемые m-резолюциями, должны со-

держать предикатные символы, максимальные относительно порядка в соответствующих дизъюнктах). Если m-абстракция определена посредством отображений литер, которые сохраняют как знаки, так и предикатные символы литер, то и в абстрактном, и в исходном пространствах могут быть использованы различные комбинации гиперрезолюции и упорядочения [11, 14]. Мы рекомендуем выбирать знаки предикатных символов так, чтобы множество S входных дизъюнктов было «почти» хорновским множеством [4], и затем выполнять лок-резолюцию с таким образом подобранными индексами, чтобы отрицательные литеры устранялись в первую очередь. Улучшение, которое можно получить благодаря использованию таких полных стратегий, по всей видимости, мало́ по сравнению с улучшением, даваемым применением m-абстракций. Однако, даже если стратегии дают улучшение только в 2 или в 3 раза, это уже будет существенно.

Следует заметить, что поисковое пространство будет уменьшаться, когда глубина вывода приближается к максимальной глубине d. Это происходит потому, что множество абстрактных m-дизъюнктов вблизи d будет ограничено только теми m-дизъюнктами, из которых D' или нечто более общее может быть выведено за небольшое число шагов. Поэтому множество m-дизъюнктов, выводимых из S на глубине, близкой к d, также будет ограничено. Подобное ограничение пространства поиска вблизи максимума глубины очень непохоже на поведение большинства обычных процедур доказательства, у которых поисковое пространство растет все больше и больше с увеличением глубины вывода. Наибольшие размеры поискового пространства будут, вероятно, на средних глубинах. Стратегии «proofsearch 3» и «proofsearch 4», которые будут представлены ниже, впрочем как и неполная стратегия «proofsearch 1» из разд. 2, также ограничивают поисковое пространство, когда глубина вывода приближается к максимальному значению.

Еще одно свойство стратегий, основанных на абстракции, состоит в том, что они автоматически выбирают те m-дизъюнкты из S, которые кажутся относящимися к рассматриваемой задаче, т. е. m-дизъюнкт C из S совсем не будет использован, если ни одна его абстракция D не входит в имеющее глубину d доказательство из $f(S)$ m-дизъюнкта D' или какого-либо более общего m-дизъюнкта. Поэтому стратегии, основанные на абстракции, могут принести пользу, когда имеется большое число входных дизъюнктов.

Теперь мы обсудим методы использования в поиске доказательства нескольких m-абстракций одновременно.

5. ОДНОВРЕМЕННОЕ ИСПОЛЬЗОВАНИЕ НЕСКОЛЬКИХ m-АБСТРАКЦИЙ

Определение. Допустим, что M — k-местный предикат, определенный на множестве m-дизъюнктов, т. е. если $C_1, C_2, \ldots, C_k$ — m-дизъюнкты, то $M(C_1, C_2, \ldots, C_k)$ либо истинен, либо ложен. Мы следующим образом расширим M до k-местного предиката на множестве m-резолюционных доказательств. Пусть $U_1, U_2, \ldots, U_k$ — m-резолюционные доказательства. Предикат $M(U_1, U_2, \ldots, U_k)$ будет истинным тогда и только тогда, когда все U_i имеют одну и ту же форму и существуют такие соответствия формы $\sim_i$ между U_i и U_{i+1}, что для всех узлов N_i (N_1 в $U_1, \ldots, N_k$ в U_k) из $N_1 \sim_1 N_2$, $N_2 \sim_2 N_3, \ldots, N_{k-1} \sim_{k-1} N_k$ следует истинность $M(\text{label}(N_1), \text{label}(N_2), \ldots, \text{label}(N_k))$. Заметим, что если M_1 и M_2 — отношения и $M_1(C_1, \ldots, C_k) \supset M_2(C_1, \ldots, C_k)$ для всех m-дизъюнктов $C_1, \ldots, C_k$, то $M_1(U_1, \ldots, U_k) \supset M_2(U_1, \ldots, U_k)$ для всех m-резолюционных доказательств $U_1, \ldots, U_k$.

Пусть f — функция m-абстракции, а $M(B, C)$ — отношение «$\exists\theta B\theta \in f(C)$» на множестве m-дизъюнктов (θ — подстановка). Нам известно, что если $T \to_f U$, то $M(U, T)$ истинно. Допустим, что $f_1, f_2, \ldots, f_k$ — функции m-абстракции, $U_1, U_2, \ldots, U_k$ — m-абстрактные доказательства, для которых $T \to_{f_i} U_i$ при $1 \leqslant i \leqslant k$, и пусть $M'(B_1, B_2, \ldots, B_k)$ — отношение $(\exists C)[\exists\theta_1 B_1\theta_1 \in f_1(C)$ и ... и $\exists\theta_k B_k\theta_k \in f_k(C)]$. В этом случае нетрудно показать, что $M'(U_1, U_2, \ldots, U_k)$ истинно.

Сказанное выше приводит к следующей стратегии поиска. Допустим, что мы ищем доказательство m-дизъюнкта C' из множества m-дизъюнктов S. Выберем $D'_i \in f_i(C)$ для $1 \leqslant i \leqslant k$ и найдем доказательства U_i m-дизъюнктов B'_i из $f_i(S)$, такие что D'_i является примером B'_i и отношение $M'(U_1, U_2, \ldots, U_k)$ истинно. Кажется маловероятным, чтобы такие доказательства U_i нашлись, когда соответствующее доказательство C' из S не существует. Если доказательства U_i найдены, то используем их для управления поиском доказательства T m-дизъюнкта C' из S.

Может оказаться, что отношение M' слишком трудно вычислить, поскольку у нас нет очевидного способа проверить существование требуемого дизъюнкта C. Однако часто оказывается, что существуют другие отношения M'_1, которые легко вычислять, причем из $M'(B_1, B_2, \ldots, B_k)$ следует $M'_1(B_1, B_2, \ldots, B_k)$. Поэтому мы можем искать такие доказательства $U_1, \ldots, U_k$, чтобы $M'_1(U_1, \ldots, U_k)$ было истинным. Например, $M'_1(B_1, \ldots, B_k)$ может устанавливать, что $|B_1| = |B_2| = \ldots = |B_k|$. Это отношение будет работать, когда все f_i определены посредством отображений литер. Если f_i сохраняют число различных предикатных символов, то $M'_1(B_1, B_2, \ldots, B_k)$ может обозначать,

что все B_i имеют одинаковое число литер с различными предикатными символами. Если же f_i сохраняют знаки и предикатные символы литер, то это также может быть отражено в M_i'. Когда $T \to_{f_i} U_i$ при $1 \leqslant i \leqslant k$, $M_1'(U_1, \ldots, U_k)$ будет истинно. Следовательно, мы можем искать такие доказательства U m-дизъюнктов B_i' из $f(S)$, для которых $M_1'(U_1, \ldots, U_k)$ истинно, и использовать эти доказательства для управления поиском доказательства C' из S. Это по-прежнему дает нам полную стратегию доказательства теорем. Заметим, что если $T \to_i U_i$, то глубина доказательства T совпадает с глубиной U_i при любом $1 \leqslant i \leqslant k$. Данный факт может быть использован для проведения поисков доказательств T в порядке возрастания глубины. Как и прежде, если поиск доказательств B_i' из $f_i(S)$ все еще слишком сложен, можно использовать дополнительные уровни абстракции.

5.1. Стратегия без согласования

Следующая процедура использует для поиска доказательства несколько абстракций одновременно. Она, однако, не согласовывает m-абстрактные доказательства друг с другом заблаговременно. Иными словами, мы не проверяем, имеют ли все m-абстрактные доказательства одну и ту же форму и удовлетворяют ли они подходящему отношению между доказательствами. В результате экономятся усилия, необходимые для такого согласования, но по всей вероятности увеличиваются размеры поискового пространства. Но даже и в этом случае предлагаемая процедура должна иметь меньшее поисковое пространство, чем «proofsearch 2», которая использует только одну m-абстракцию.

procedure proofsearch 3(S, C', $\{f_1, f_2, \ldots, f_n\}$);
[[ищется доказательство C' из S с использованием множества $\{f_1, \ldots, f_n\}$ (не обязательно различных) m-абстракций; стратегия полна.]]
for $i = 1$ *to* n *do* выбрать D_i' из $f_i(C')$ *od*;
for $i = 1$ *to* n *do* $S_i \leftarrow \{\langle D, 0\rangle$: $(\exists C \in S) D \in f_i(C)\}$ *od*;
for $d = 1$ *to* ∞ until C' выводим из S *do*
[[ищется доказательство глубины d]]
for $i = 1$ *to* n *do*
$V_1^i \leftarrow$ наименьшее m-резолюционное доказательство, такое что
(a) $S_i \subset \text{Nodes}\,(V_1^i)$ и
(b) если $\langle B1, d_1\rangle \in \text{Nodes}\,(V_1^i), \langle B2, d_2\rangle \in \text{Nodes}\,(V_1^i)$, $d_1 < d$, $d_2 < d$ и $B3$ является m-резольвентой

от $B1$ и $B2$, то $\langle B3, d_3\rangle \in \text{Nodes}(V_1^i)$ и $\langle\langle B1, d_1\rangle, \langle B2, d_2\rangle, \langle B3, d_3\rangle\rangle \in \text{M Res}(V_i)$, где $d_3 = 1 + \max(d_1, d_2)$;

$V_2^i \leftarrow$ наименьшее m-резолюционное доказательство, такое, что

(a) если $\langle B, d\rangle \in \text{Nodes}(V_1^i)$ и D_i' является примером B, то $\langle B, d\rangle \in \text{Nodes}(V_2^i)$, и

(b) если $N3 \in \text{Nodes}(V_2^i)$ и $\langle N1, N2, N3\rangle \in \text{M Res}(V_1^i)$, то $N1 \in \text{Nodes}(V_2^i)$, $N2 \in \text{Nodes}(V_2^i)$ и $\langle N1, N2, N3\rangle \in \text{M Res}(V_2^i)$

od;

[[V_2^i представляет имеющие глубину d доказательства из $f_i(S)$ m-дизъюнктов, примером которых является D_i'. Они могут быть найдены полным перебором, как указано выше, или с использованием дополнительных уровней m-абстракции.]]

$W \leftarrow S$;

[[Если узел N имеет вид $\langle B1, d_1\rangle$, то мы говорим, что $\text{label}(N) = B1$ и $\text{depth}(N) = d_1$.]]

loop

while ($C' \notin W$ *and there exist* m-дизъюнкты $C1$, $C2$ и $C3$ *such that*

(a) $C1 \in W$, $C2 \in W$ и $C3 \notin W$

(b) $C3$ — m-резольвента $C1$ и $C2$

(c) для любого i $(1 \leqslant i \leqslant n)$ существует резолюция $\langle N1, N2, N3\rangle \in \text{M Res}(V_2^i)$, такая что $\text{depth}(C1) = \text{depth}(N1)$, $\text{depth}(C2) = \text{depth}(N2)$, $\text{label}(N1)$ имеет пример в $f_i(C1)$, $\text{label}(N2)$ имеет пример в $f_i(C2)$ и $\text{label}(N3)$ имеет пример в $f_i(C3)$);

добавить $C3$ к W;

[[Может оказаться полезным выбирать $C1$, $C2$ и $C3$ такими, чтобы значение $\max(\text{depth}(C1), \text{depth}(C2))$ было по возможности наибольшим.]]

repeat;

od;

end proofsearch 3;

В данной программе W — m-резолюционное доказательство. Вместе с каждым m-дизъюнктом C из W мы храним целое число, указывающее его глубину. Заметим, что один и тот же m-дизъюнкт может встретиться в W более чем на одной глубине.

Рассматриваемый подход объясняется желанием сократить время и размеры требуемой памяти для построения m-абстрактных доказательств. Время работы «proofsearch 3» с n m-абстракциями будет самое большее в n раз превосходить время работы «proofsearch 2» с одной m-абстракцией. Если бы, однако, каждая m-абстракция удаляла из поискового пространства все излишние m-дизъюнкты, кроме $1/r$ их части, то n абстракций удалили бы все, кроме $(1/r)^n$, части излишних дизъюнктов при условии, что m-абстракции в определенном смысле «независимы». Например, если r-4, а $n = 10$, то при выполнении указанных допущений были бы удалены все, кроме $1/10^6$, ненужных дизъюнктов. Число $n = 20$ или даже $n = 50$ представляется вполне приемлемым для современных компьютеров, особенно при использовании параллельности, так как каждое абстрактное пространство может быть построено независимо от остальных. Можно было бы, по-видимому, удалить почти все m-резолюции, за исключением тех, которые ведут к желаемому доказательству, если бы удалось найти подходящие m-абстракции.

Описанный метод дает мультипликативный эффект в уменьшении поискового пространства при аддитивном увеличении прилагаемых усилий, что достигается за счет ослабления связи между абстрактными доказательствами и исходным поисковым пространством. Запоминаются только глубины, на которых выводятся m-дизъюнкты, а большая часть структуры m-абстрактных доказательств игнорируется. Возможны и другие подходы, использующие больше информации о структуре m-абстрактных доказательств, но полностью их не согласовывающие.

5.2. Стратегия с согласованием

Теперь мы представим другую подобную стратегию, которая использует несколько абстракций одновременно. Эта стратегия пытается «согласовать» m-абстрактные доказательства перед тем, как искать доказательство из исходного множества m-дизъюнктов.

Если $M1$ и $M2$ — k-местные отношения на множестве m-дизъюнктов, а $T_1, T_2, \ldots, T_k$ — m-резолюционные доказательства, то, как и прежде, мы определяем $(M1; M2)(T_1, T_2, \ldots, T_k)$. Здесь $M1$ задает отношение между дизъюнктами, приписанными начальным узлам, а $M2$ — между дизъюнктами, которые приписаны узлам, начальными не являющимися. Допустим, что $S_1, S_2, \ldots, S_k$ — множества m-дизъюнктов, а V_i — множество m-резолюций из S_i. Процедура «match» находит все наборы $\langle T_1, T_2, \ldots, T_k\rangle$ m-резолюционных доказательств из $S_1, S_2, \ldots, S_k$ соответственно, такие что T_i — начальное поддоказа-

тельство V_i $(1 \leqslant i \leqslant k)$ и $(M1; M2)(T_1, T_2, \ldots, T_k)$ истинно. Наборы эти порождаются не явным, а косвенным образом, как это будет описано ниже.

Определение. *Векторной m-резолюцией* называется тройка $\langle u, v, w\rangle$ узлов, в которой метки u, v и w являются при некотором k наборами из k m-дизъюнктов, и, кроме того, для каждого i $(1 \leqslant i \leqslant k)$ i-я компонента $\mathrm{label}(w)$ должна быть m-резольвентой i-х компонент $\mathrm{label}(u)$ и $\mathrm{label}(v)$.

Определение. *Векторным m-дизъюнктом* называется набор из k m-дизъюнктов при некотором k.

Векторные m-резолюционные доказательства мы определим таким же способом, каким были определены m-резолюционные и обычные резолюционные доказательства. Если V — векторное m-резолюционное доказательство, то мы требуем, чтобы все векторные m-дизъюнкты из V имели одинаковое число компонент. Мы пишем $\mathrm{VMRes}(V)$ для обозначения множества векторных m-резолюций векторного m-резолюционного доказательства V.

Пусть $\bar{B}$ — набор из k m-дизъюнктов. Мы пишем B_i для обозначения i-й компоненты $\bar{B}$, а сам набор $\bar{B}$ записываем в виде $\langle B_1, B_2, \ldots, B_k\rangle$. Аналогичные обозначения мы используем и для наборов узлов $\bar{x}$.

Определение. Пусть $\Pi_i(\bar{B}, C)$ — отношение, утверждающее, что B_i есть C. Здесь $\bar{B}$ — вектор m-дизъюнктов, а C — m-дизъюнкт.

Определение. Если V — векторное m-резолюционное доказательство, а T — обычное m-резолюционное доказательство, то мы пишем $\Pi_i(V, T)$, когда существует соответствие формы $\sim$ между V и T, такое что если $N1 \in \mathrm{Nodes}(V)$, $N2 \in \in \mathrm{Nodes}(T)$ и $N1 \sim N2$, то $\Pi_i(\mathrm{label}(N1), \mathrm{label}(N2))$ истинно. Таким образом, если $\Pi_i(V, T)$ истинно, то T — «i-я компонента» доказательства V. Здесь мы используем обычное определение отношения между доказательствами.

Процедура «$\mathrm{match}(V_1, V_2, \ldots, V_k, M1, M2)$» выдает векторное m-резолюционное доказательство V, обладающее следующими свойствами.

Допустим, что $V_1, V_2, \ldots, V_k$ — m-резолюционные доказательства, $T_1, T_2, \ldots, T_k$ — начальные поддоказательства $V_1, V_2, \ldots, V_k$ соответственно, и пусть $(M1; M2)(T_1, T_2, \ldots, T_k)$ истинно. Тогда существует начальное поддоказательство W доказательства V, такое что $\Pi_i(W, T_i)$ истинно при $1 \leqslant i \leqslant k$. Следовательно, W имеет T_i в качестве «i-й компоненты» и W представляет согласование доказательств $T_1, T_2, \ldots, T_k$. Доказательство V поэтому представляет все возможные способы согласования начальных поддоказательств $V_1, V_2, \ldots, V_k$.

procedure match($V_1, V_2, \ldots, V_k, V, M1, M2$);
[[дает в результате такое векторное m-резолюционное доказательство V, что для всех $T_1, T_2, \ldots, T_k$, если T_i — начальное поддоказательство V_i $(1 \leqslant i \leqslant k)$ и $(M1; M2)(T_1, T_2, \ldots, T_k)$ истинно, то существует начальное поддоказательство W доказательства V, для которого $\Pi_i(W, T_i)$ истинно.]] VMRes(V)$\leftarrow\varnothing$;
Nodes(V)$\leftarrow$ {$\langle N1, N2, \ldots, Nk\rangle$: Ni — начальный узел V_i и $M1$(label($N1$), label($N2$), ..., label(Nk)) истинно}; [[для узла N из V вида $\langle N1, N2, \ldots, Nk\rangle$ мы полагаем label(N) = $= \langle$label($N1$), label($N2$), ..., label(Nk)$\rangle$]]
loop
while(*there exist* узлы $\bar{x}$ и $\bar{y}$ из V *such that* $\bar{x}$ и $\bar{y}$ еще не резольвировали);
[[к $\bar{x}$ и $\bar{y}$ применяем правило резолюции]]
для всех $\bar{z}$, таких что $\langle x_i, y_i, z_i\rangle \in$ MRes(V_i) при любом i, $1 \leqslant i \leqslant k$ и $M2$(label(z_1), label(z_2), ..., label(z_k)), *do*
добавить $\bar{z}$ к Nodes(V);
добавить $\langle \bar{x}, \bar{y}, \bar{z}\rangle$ и $\langle \bar{y}, \bar{x}, \bar{z}\rangle$ к VMRes(V);
od;
repeat;
end match

Применение принципа «разделяй и властвуй» может увеличить эффективность процедуры согласования. Мы можем, к примеру, сначала согласовывать $V_1, V_2, \ldots, V_{[k/2]}$, затем $V_{[k/2]+1}, V_{[k/2]+2}, \ldots, V_k$ и, наконец, $V_1, V_2, \ldots, V_k$. Этот прием может оказаться полезным, поскольку на стадии заключительного согласования нам не нужно было бы даже рассматривать те векторные m-дизъюнкты, которые были устранены на более ранних этапах согласования.

На практике нам может потребоваться экономичное представление доказательства V. Например, если $Q_1, Q_2, \ldots, Q_k$ — множества узлов, то мы можем использовать $\langle Q_1, Q_2, \ldots, Q_k\rangle$ для представления множества узлов $\{\langle x_1, x_2, \ldots, x_k\rangle$: $x_i \in Q_i$ при $1 \leqslant i \leqslant k\}$. Мы можем аналогичным образом использовать $\langle\langle Q_1, \ldots, Q_k\rangle, \langle Q'_1, \ldots, Q'_k\rangle, \langle Q''_1, \ldots, Q''_k\rangle\rangle$ для обозначения множества векторных m-резолюций. Но даже такое представление может стать слишком громоздким, если k больше 3 или 4.

Мы теперь покажем, как векторное m-резолюционное доказательство, порожденное процедурой «match», можно использовать для управления поиском доказательства в исходном пространстве. Пусть $f_1, f_2, \ldots, f_k$ — функции m-абстракции, $M1(\bar{x}, C)$ — это отношение $x_1 \in f_1(C) \wedge x_2 \in f_2(C) \wedge \ldots \wedge x_k \in f_k(C)$, а $M2(\bar{x}, C)$ — отношение «x_1 имеет пример в $f_1(C)$ и ... и x_k имеет пример в $f_k(C)$». Расширим $M1$ и $M2$ до от-

ношений между доказательствами обычным способом. В этом случае легко показать, что если T — m-резолюционное доказательство, то существует векторное m-резолюционное доказательство W, для которого $(M1; M2)(W, T)$ истинно. Такое доказательство будет удовлетворять отношению $\Pi_i(W, T_i)$ $(1 \leqslant i \leqslant m)$ при некотором T_i, для которого $T \to_{f_i} T_i$ истинно.

Заметим, что если T — доказательство из S, $(M1; M2)(W, T)$ истинно и $\Pi_i(W, T_i)$ истинно для всех i $(1 \leqslant i \leqslant k)$, то истинным будет также $(R1; R2)(T_1, T_2, \ldots, T_k)$, где отношения $R1$ и $R2$ определяются следующим образом:

$R1(D_1, D_2, \ldots, D_k)$ истинно тогда и только тогда, когда $(\exists C \in S)(D_i \in f_i(C), 1 \leqslant i \leqslant k)$;

$R2(B_1, B_2, \ldots, B_k)$ истинно тогда и только тогда, когда $(\exists C)(\exists \theta_i\ B_i\theta_i \in f_i(C), 1 \leqslant i \leqslant k)$.

Предположим, что мы ищем доказательство m-дизъюнкта C' из S. Пусть D_i'-произвольные m-дизъюнкты, такие что $D_i' \in f_i(C')$ $(1 \leqslant i \leqslant k)$, а V_i — m-резолюционные доказательства из $f_i(S)$, содержащие изоморфные копии всех имеющих глубину d минимальных m-резолюционных доказательств из $f_i(S)$ m-дизъюнктов B_i', примерами которых являются D_i'. Если существует доказательство T m-дизъюнкта C из S, имеющее глубину d, то существуют доказательства T_i m-дизъюнктов B_i' из $f_i(S)$ $(1 \leqslant i \leqslant k)$, такие что $T \to_{f_i} T_i$ и T_i является начальным поддоказательством V_i. Поэтому $(R1; R2)(T_1, \ldots, T_k)$ будет истинно, а значит, «match$(V_1, V_2, \ldots, V_k, V, R1, R2)$» породит некоторое доказательство W, для которого $\Pi_i(W, T_i)$ будет истинно при $1 \leqslant i \leqslant k$. Более точно, W будет начальным поддоказательством доказательства, порожденного процедурой «match». Заметим, что $(M1; M2)(W, T)$ также будет истинно. Следовательно, приемлемая стратегия поиска заключается в порождении посредством «match» такого доказательства W и применения к W процедуры «ndfindm» для получения T. С этой целью необходимо модифицировать «ndfindm» так, чтобы она могла обращаться с векторными m-резолюционными доказательствами. В результате получается стратегия доказательства теорем «proofsearch 4», полностью аналогичная «proofsearch 2», за исключением того лишь, что «proofsearch 4» использует одновременно несколько абстракций. Сейчас мы дадим описание «proofsearch 4». Эта процедура ищет доказательства в порядке увеличения глубины и использует недетерминированный поиск так же, как и «ndfindm», но для *векторов* мультидизъюнктов.

procedure proofsearch 4 $(S, C', f_1, f_2, \ldots, f_k, D_1', D_2', \ldots, D_k')$;

[[ищем доказательство C' из S, используя (не обязательно различные) функции m-абстракции $f_1, f_2, \ldots, f_k$ и m-дизъ-

юнкты D'_i, такие что $D'_i \in f_i(C')$ при $1 \leqslant i \leqslant k$. Стратегия полна.]]
пусть $M1(D_1, D_2, \ldots, D_k)$ — отношение между m-дизъюнктами $(\exists C \in S)(D_i \in f_i(C), 1 \leqslant i \leqslant k)$;
пусть $M2(B_1, \ldots, B_k)$ — некоторое отношение между m-дизъюнктами, такое что из $(\exists C)(\exists \theta_i\ B_i\theta_i \in f_i(C)$ при $1 \leqslant i \leqslant k)$ следует $M2(B_1, B_2, \ldots, B_k)$;
for $d = 1$ *to* ∞ *until* C' выводим из S *do*
[[ищем доказательство глубины d]]
for $i = 1$ *to* k *do*
$V_i \leftarrow m$-резолюционное доказательство, содержащее в качестве начальных поддоказательств изоморфные копии всех минимальных доказательств T глубины d из $f_i(S)$ m-дизъюнктов, имеющих D'_i своими примерами;
[[порождаем V_i так же, как и в «proofsearch 2»]]
od;
match$(V_1, V_2, \ldots, V_k, V, M1, M2)$;
для всех начальных узлов $N \in$ Nodes(V) *do*
m-clauses$(N) \rightarrow \{C \in S:\ D_i \in f_i(C),\ 1 \leqslant i \leqslant k\}$, где label$(N) = \langle D_1, D_2, \ldots, D_k\rangle$*od*;
для всех $N \in$ Nodes(V), не являющихся начальными узлами V, *do*
m-clauses$(N) \leftarrow \varnothing$ *od*;
loop
while (C' еще не был выведен из S *and there exists* векторная m-резолюция $\langle N1, N2, N3\rangle$ из VMRes(V) и m-дизъюнкты $C1$, $C2$ и $C3$ *such that*
(a) $C1 \in m$-clauses$(N1)$ и $C2 \in m$-clauses$(N2)$
(b) $C3$ является m-резольвентой $C1$ и $C2$
(c) $C3 \notin m$-clauses$(N3)$
(d) для всех i $(1 \leqslant i \leqslant k)$ $\exists \theta_i\ B_i\theta_i \in f_i(C3)$, где $\langle B_1, B_2, \ldots, Bk\rangle =$ label$(N3)$));
добавить $C3$ к m-clauses$(N3)$;
repeat;
od
end proofsearch 4;

Так же как и для процедур «proofsearch 1», «proofsearch 2» и «proofsearch 3», если все абстркатные пространства порождаются полностью, мы можем ограничить m-резолюции из S согласно любой полной стратегии доказательства теорем и все равно получить «proofsearch 4» в качестве полной стратегии. Более того, если m-абстракции удовлетворяют подходящим свойствам, подобным тем, о которых говорилось выше, мы можем даже ограничить абстрактный поиск согласно некоторой полной стратегии доказательства теорем. Такие комбинации m-абстрак-

ций и полных стратегий создают ограничения на возможные применения m-резолюций, сохраняя при этом глобальность стратегии доказательства теорем. Иными словами, на выбор очередной m-резолюции нетривиальным образом оказывает влияние структура задачи в целом, а не дизъюнкты, способные резольвировать согласно определенным критериям.

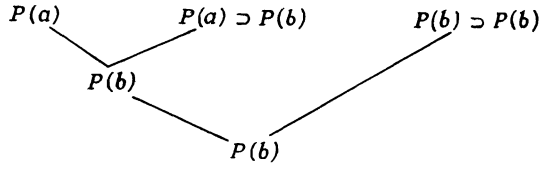
Согласование абстрактных доказательств в «proofsearch 4» за счет дополнительных затрат должно привести к более ограничительной поисковой стратегии, чем «proofsearch 3». Затраты, необходимые для использования k m-абстракций, могут составлять приблизительно k-ю степень тех затрат, которые требуются для применения одной m-абстракции. Это происходит по той причине, что необходимо хранить наборы из k m-дизъюнктов, а не единичные m-дизъюнкты. Для небольших k, однако, данная стратегия представляется осуществимой. При этом некоторую помощь может оказать использование упоминавшегося раньше экономичного представления наборов.

Теперь мы приведем простой и несколько более сложный пример использования процедуры «proofsearch 4».

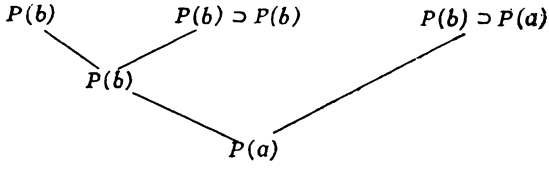
Пусть $S = \{P(a, b), P(a, b) \supset P(b, b), P(b, b) \supset P(b, a)\}$, $C' = P(b, a)$, f_1 отображает $P(x, y)$ на $P(x)$ и f_2 отображает $P(x, y)$ на $P(y)$. Для простоты предположим, что мы ограничиваем доказательства только $P1$-выводами, которые требуют, чтобы один из предков каждого дизъюнкта был положительным дизъюнктом. И пусть $P1$-вывод используется как в исходном, так и в абстрактных пространствах. Тогда

$$f_1(S) = \{P(a),\ P(a) \supset P(b),\ P(b) \supset P(b)\},$$
$$f_2(S) = \{P(b),\ P(b) \supset P(b),\ P(b) \supset P(a)\}.$$

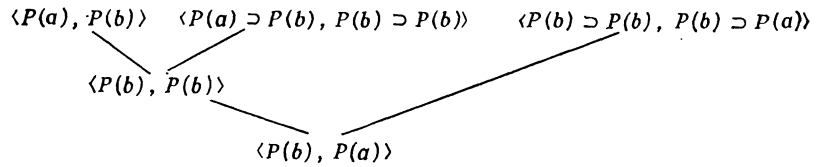
Если взять глубину d равной 2, то доказательство V_1 будет иметь следующий вид:



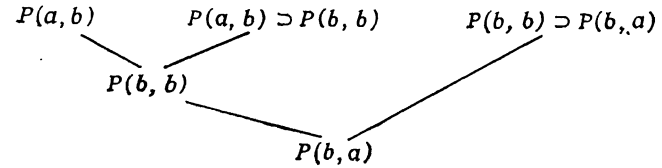
Доказательство V_2 такой же глубины будет выглядеть следующим образом:



Отметим, что мы включаем в V_1 только те m-резолюции, которые участвуют в доказательстве $P(b)$ глубины 2, а в V_2 — только те m-резолюции, которые участвуют в имеющем глубину 2 доказательстве $P(a)$. Согласовывая V_1 и V_2, мы получаем следующее доказательство:



Из этого доказательства легко уже находится требуемое доказательство $P(b, a)$ из S:



Хотя согласование и не устраняет ложные доказательства на глубине 2, оно все же устраняет из V_1 и V_2 ложное доказательство на глубине 1. Отметим необходимость сохранения в V_1 и V_2 тавтологий $P(b) \supset P(b)$.

Сейчас мы приведем более сложный пример использования нескольких m-абстракций с согласованием. Этот пример иллюстрирует также применение семантических абстракций. Задача, взятая из [7], формулируется так: доказать, что если A — простое число и $A * C^2 = B^2$, то B и C не являются взаимно простыми. Мы считаем, что переменные принимают значения из множества целых чисел, бóльших единицы. Предикат $P(x)$ означает «x — простое число», $M(x, y, z)$ означает «$x * y = z$», $S(x)$ — «x в квадрате», $D(x, y)$ — «делит y» и $F(x, y)$ есть y/x. Входные дизъюнкты S имеют следующий вид:

A1 $P(A)$
A2 $M(A,S(C),S(B))$
A3 $M(X,X,S(X))$
A4 $M(X,Y,Z) \supset M(Y,X,Z)$
A5 $M(X,Y,Z) \supset D(X,Z)$
A6 $P(X) \wedge D(X,W) \wedge M(Y,Z,W) \supset D(X,Y) \vee D(X,Z)$
A7 $D(X,Y) \wedge M(X,S(Z),S(Y)) \supset M(X,S(F(X,Y)),S(Z))$
A8 $\neg D(X,C) \vee \neg D(X,B)$

Для иллюстрации ложных доказательств мы добавим к S еще два дизъюнкта:

A9 $\neg D(S(X),S(X+1))$
A10 $\neg P(S(X))$

Они истинны, поскольку рассматриваемая область состоит из целых чисел, больших единицы. Нашей целью является доказательство NIL из S.

Мы рассмотрим две семантические абстракции. Каждая из них имеет своей областью $I=\{2, 3, 4, 5, \ldots\}$ и интерпретирует функции стандартным образом, т. е. $S(3)=9$ и т. д. Однако первая абстракция придает A, B и C значения 2, 4 и 3 соответственно, тогда как вторая придает им соответственно значения 4, 12 и 6. Например, $A3$ имеет абстракции $M(2, 2, 4)$, $M(3, 3, 9)$, $M(4, 4, 16)$, ... как в первой, так и во второй абстракции, что ведет к бесконечному абстрактному пространству. Для работы с ним мы ограничимся некоторым конечным подмножеством $I1$ множества I и отобразим термы на «неопределенный элемент» U, если их значения выходят за пределы $I1$. Чтобы получить слабо изменяющуюся последовательность абстракций, мы можем постепенно увеличивать размеры этого подмножества I. В качестве подходящего множества $I1$ можно взять значения всех неглубоких основных термов из эрбрановского универсума S. В данном примере $I1$ состояло бы тогда из всех значений, которые можно получить из A, B и C небольшим числом операций возведения в квадрат и деления. Так, если $A=2$, $B=4$ и $C=3$, то числа 2^2, 4^2, 3^2, $4/2$, $4^2/2$, $4^2/2^2$ и т. д. можно было бы включить в $I1$. Поэтому для первой абстракции в качестве $I1$ имеет смысл выбрать множество {2, 3, 4, 8, 9, 16}, если абстрактное пространство не слишком велико. Аналогично, для второй абстракции приемлемым выбором $I1$ является множество {2,3,4,6,9,12,16,24,36,144}. Если бы дизъюнкт $A9$ принадлежал S, то в число функций, участвующих в построении $I1$, нам нужно было бы включить функцию $x+1$.

Следующее доказательство является ложным доказательством относительно первой абстракции.

1. $M(2,9,16)$	Абстракция A2
2. $M(2,9,16) \supset M(9,2,16)$	Абстракция A4
3. $M(9,2,16)$	1, 2
4. $M(9,2,16) \subset D(9,16)$	Абстракция A5
5. $D(9,16)$	3, 4
6. $\neg D(9,16)$	Абстракция A9
7. NIL	5, 6

Следующее доказательство является ложным относительно второй абстракции.

1. $P(4)$	Абстракция A1
2. $\neg P(4)$	Абстракция A10
3. NIL	1, 2

Из приведенных примеров должно быть понятно, как возникают ложные доказательства, когда абстрактные литеры унифицируются, а соответствующие им исходные литеры нет. Если обе абстракции используются совместно, эти ложные доказательства будут устранены. Сказанное выше иллюстрирует, как использование нескольких абстракций может устранить доказательства, имеющие место в случае одной абстракции.

Полное опровержение для данного примера можно найти в [7]. Мы приведем лишь часть доказательства вместе с двумя его абстракциями. Это доказательство того, что A делит B:

1. $D(A,S(B))$	A2, A5
2. $\neg P(A) \vee \neg M(Y,Z,S(B)) \vee D(A,Y) \vee D(A,Z)$	1, A6
3. $\neg P(A) \vee D(A,B) \vee D(A,B)$	2, A3
4. $D(A,B) \vee D(A,B)$	3, A1

Заметим, что $D(A,B) \vee D(A,B)$ не сводится автоматически к $D(A,B)$, так как мы используем m-дизъюнкты, являющиеся мультимножествами литер. По отношению к первой абстракции мы получаем следующее доказательство.

A1.1 $P(2)$	абстракция A1
A2.1 $M(2,9,16)$	абстракция A2
A3.1 $M(4,4,16)$	абстракция A3
A5.1 $\neg M(2,9,16) \vee D(2,16)$	абстракция A5
A6.1 $\neg D(2,6) \vee \neg M(4,4,16) \vee \neg P(2) \vee D(2,4) \vee D(2,4)$	абстракция A6
1. $D(2,16)$	A2.1, A5.1
2. $\neg P(2) \vee \neg M(4,4,16) \vee D(2,4) \vee D(2,4)$	1, A6.1
3. $\neg P(2) \vee D(2,4) \vee D(2,4)$	2, A3.1
4. $D(2,4) \vee D(2,4)$	3, A1.1

Заметим, что относительно первой абстракции дизъюнкты A3, A5 и A6 имеют много абстракций, и мы выписываем только *одну* из них. По отношению ко второй абстракции мы получаем следующее доказательство.

A1.2 $P(4)$	
A2.2 $M(4,36,144)$	
A3.2 $M(12,12,144)$	
A5.2 $\neg M(4,36,144) \vee D(4,144)$	
A6.2 $\neg D(4,144) \vee \neg M(12,12,144) \vee \neg P(4) \vee D(4,12) \vee D(4,12)$	
1. $D(4,144)$	A2.2, A5.2
2. $\neg P(4) \vee \neg M(12,12,144) \vee D(4,12) \vee D(4,12)$	1, A6.2
3. $\neg P(4) \vee D(4,12) \vee D(4,12)$	2, A3.2
4. $D(4,12) \vee D(4,12)$	3, A1.2

Хотя $2 * 9 \neq 16$, мы на время воздержимся от критики ради получения абстрактных доказательств, содержащих $M(2,9,16)$. Впоследствии мы станем более придирчивыми и будем требовать строгих доказательств.

Вообще говоря, семантические абстракции, по-видимому, полезны тогда, когда дизъюнкты выражаются в реляционной форме, т. е. когда $f(x_1, \ldots, x_n) = z$ представляется отношением $R_f(x_1, \ldots, x_n, z)$. Причина этого состоит в том, что если мы пытаемся доказать $t_1 = t_2$ для термов t_1 и t_2, то семантической абстракцией, в которой $t_1 = t_2$ истинно, равенство $t_1 = t_2$ преобразуется в дизъюнкты вида $a = a$. А дизъюнкт $a = a$ не сохраняет почти ничего из структуры исходного равенства, тогда как реляционная форма $t_1 = t_2$ сохранит гораздо больше из его структуры даже при абстракции. Например, одной из реляционных форм равенства $x * (y + z) = x * y + x * z$ может быть

$$A(y, z, w) \wedge M(x, y, u) \wedge M(x, z, v) \wedge A(u, v, t) \supset M(x, w, t),$$

где $M(x, y, z)$ обозначает $x * y = z$, а $A(x, y, z)$ обозначает $x + y = z$. Для доказательства этой формулы мы заменим все переменные новыми константами и получим следующий дизъюнкт:

$$A(C6, C7, C4) \wedge M(C5, C6, C2) \wedge M(C5, C7, C3) \wedge A(C2, C3, C1) \supset M(C5, C4, C1).$$

Затем мы добавим $A(C6, C7, C4), \ldots, A(C2, C3, C1)$ к множеству гипотез и попытаемся доказать $M(C5, C4, C1)$ или какой-либо его поддизъюнкт. Интерпретируя $C1$, $C2$, $C3$, $C4$, $C5$, $C6$ и $C7$ элементами некоторой области, в которой $C6 + C7 = C4, \ldots, C2 + C3 = C1$ истинны, можно получить много различных семантических абстракций. Если эта область бесконечна, указанным выше способом можно выделить из нее конечные подмножества. Заметим, что только что рассмотренный пример с $A * C^2 = B^2$ частично представлен в реляционной форме, причем $M(x, y, z)$ представляет $x * y = z$.

Если вместо поиска доказательства конкретного m-дизъюнкта C' из множества m-дизъюнктов S мы ищем доказательство из S произвольного m-дизъюнкта, принадлежащего некоторому множеству m-дизъюнктов $S1$, то с этой целью можно соответствующим образом модифицировать процедуру «proof-search 2». Пусть f — функция m-абстракции, и мы ищем доказательство глубины d. Общая идея состоит в том, чтобы порождать все минимальные доказательства T из $f(S)$, имеющие глубину d и такие, что $(\exists C \in S1)$ (Result(T) имеет пример в $f(C)$). Для выполнения этого нам нужно уметь определять по каждому m-дизъюнкту B, существует ли $C \in S1$, такой что B

имеет пример в $f(C)$. Когда все такие доказательства T уже порождены, их можно использовать для управления поиском доказательства из S.

Когда f и $S1$ удовлетворяют определенным условиям, существуют более эффективные методы и могут использоваться процедуры «proofsearch 3» и «proofsearch 4». Допустим, к примеру, что функция m-абстракции f определена посредством множества F отображений литер, а S — множество обычных дизъюнктов. Имеется обычный дизъюнкт C', и мы хотим определить, существует ли m-резолюционное доказательство m-дизъюнкта $C1$, такого что $\mathrm{Set}(C1) = C'$. (В этом выводе $C1$ мы считаем дизъюнкты из S m-дизъюнктами.) По теореме 3.2 такое m-резолюционное доказательство существует, если C' выводим из S обычной резолюцией.

Пусть $D \in f(C')$, и в данном случае мы рассматриваем C' как m-дизъюнкт. Следовательно, существует отображение литер $\Phi \in F$, такое что $D' = \Phi(C')$. Если теперь $\mathrm{Set}(C1) = C'$, то отсюда вытекает, что $\mathrm{Set}(\Phi(C1)) = \mathrm{Set}(D')$. Поэтому если $C1$ — произвольный m-дизъюнкт, для которого $\mathrm{Set}(C1) = C'$, то существует m-дизъюнкт $D1 \in f(C1)$, такой что $\mathrm{Set}(D1) = \mathrm{Set}(D')$. (Пусть $D1$ есть $\Phi(C1)$.)

Допустим, что T — минимальное m-резолюционное доказательство из S некоторого m-дизъюнкта $C1$, такого что $\mathrm{Set}(C1) = C'$. Тогда существует m-резолюционное доказательство Y из $f(S)$, такое что $T \rightarrow_f Y$ и $\mathrm{Result}(Y)$ имеет $D1$ своим примером (рис. 6). Поэтому для поиска подобных доказа-

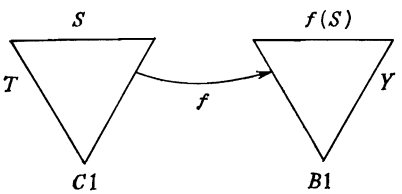


Рис. 6.

тельств T достаточно породить все доказательства Y, такие что $\mathrm{Set}(\mathrm{Result}(Y))$ имеет $\mathrm{Set}(D')$ своим примером, и использовать их для управления поиском T. Таким образом, мы можем приспособить процедуру «proofsearch 2» для поиска m-резолюционных доказательств произвольных m-дизъюнктов $C1$, таких что $\mathrm{Set}(C1) = C'$. Более того, поскольку такое доказательство Y существует для всех m-абстракций D' из $f(C')$, в поисках T мы можем одновременно использовать более одной m-абстракции C'. Таким способом для решения данной задачи могут быть использованы процедуры «proofsearch 3» и «proofsearch 4».

Даже если некоторый m-дизъюнкт $C1$, такой что $\mathrm{Set}(C1) = C'$, выводим m-резолюцией из S, отсюда никак не следует, что C' может быть выведен из S обычной резолюцией. Согласно теореме 3.4, нам известно лишь, что некоторый поддизъюнкт дизъюнкта C' выводим обычной резолюцией из S.

Указанную идею можно развивать и дальше. Пусть нам дано множество обычных дизъюнктов S и обычный дизъюнкт C'. Мы хотим определить, является ли C' следствием S, используя технику поиска, основанную на m-абстракции. C' является логическим следствием S тогда и только тогда, когда существует дизъюнкт C'', выводимый обычной резолюцией из S и поглощающий C' (см. [6]). Кроме того, если f — функция m-абстракции, определенная посредством отображений литер, то из теоремы 3.7 вытекает, что для любого $D' \in f(C')$ существует дизъюнкт $D'' \in f(C'')$, поглощающий D'. Если имеется обычное резолюционное доказательство дизъюнкта C'' из S, то существует и минимальное m-резолюционное доказательство T из S некоторого m-дизъюнкта $C1$, такого что $\mathrm{Set}(C1) = C''$. Рассуждая, как и прежде, получаем, что существует абстракция $D1$ m-дизъюнкта $C1$, для которой $\mathrm{Set}(D1)$ поглощает $\mathrm{Set}(D')$. К тому же имеется m-резолюционное доказательство Y из $f(S)$, такое что $T \rightarrow_f Y$ и $D1$ является примером $\mathrm{Result}(Y)$. Следовательно, для нахождения нужного доказательства T достаточно найти доказательство Y из $f(S)$, такое что $\mathrm{Set}(\mathrm{Result}(Y))$ поглощает $\mathrm{Set}(D')$, и использовать эти доказательства для управления поиском доказательства T. Таким способом процедура «proofsearch 2» может быть приспособлена для проверки, является ли обычный дизъюнкт C' логическим следствием множества обычных дизъюнктов S. Так как доказательства Y существуют для всех $D' \in f(C')$, если C' — логическое следствие S, процедуры «proofsearch 3» и «proofsearch 4» также можно приспособить для выполнения указанной проверки. Заметим, что если найдено какое-либо m-резолюционное доказательство T из S такого m-дизъюнкта $C1$, что $\mathrm{Set}(C1)$ поглощает $C1$, то из теоремы 3.4 нам известно, что некоторый обычный дизъюнкт $C2$, поглощающий $\mathrm{Set}(C1)$ (а значит, и C'), выводим из S обычной резолюцией. Поэтому C' является логическим следствием из S тогда и только тогда, когда такое доказательство T существует.

5.3. Сравнение абстракции с обычными резолюционными стратегиями

Мы сейчас приведем пример, иллюстрирующий сокращение размеров поискового пространства, которое может быть получено, благодаря использованию абстракции. Это простая задача

поиска, которую абстракция легко может разрешить, сводя ее к задаче поиска в пространстве состояний. Задача состоит в том, чтобы проехать из города Шайенн, штат Вайоминг, до города Де-Мойн, штат Айова, и завершить путешествие, имея при себе один батон хлеба. Часть дорожной карты представлена в виде дизъюнктов, причем $N(x, y)$ указывает, что города x и y соседние и связаны дорогой друг с другом. Допустимыми действиями являются: поездка в соседний город, покупка в городе батона хлеба и ожидание в городе в течение одной единицы времени. Мы предполагаем, что покупка хлеба занимает одну единицу времени, а поездка в соседний город — две. Предикат $AT(x, y, t, s)$ означает, что вы находитесь в городе x с y батонами хлеба в момент времени t и в ситуации s. Функция $D(y, s)$ дает ситуацию, получающуюся в результате поездки в город y в ситуации s. Функции $W(s)$ и $B(s)$ дают ситуации, получающиеся в результате ожидания в ситуации s и покупки хлеба в ситуации s соответственно. Значением функции $S1(x)$ является следующее за x целое число. Входные дизъюнкты выглядят следующим образом:

$$\mathrm{AT}(\mathrm{CH},0,0,S0)$$
$$\mathrm{AT}(X1,Y,T,S) \wedge \mathrm{N}(X1,X2) \supset AT(X2,Y,\mathrm{S1}(\mathrm{S1}(\mathrm{T})),\mathrm{D}(X2,S))$$
$$\mathrm{AT}(X1,Y,T,S) \wedge \mathrm{N}(X2,X1) \supset \mathrm{AT}(X2,Y,\mathrm{S1}(\mathrm{S1}(T)),\mathrm{D}(X2,S))$$
$$\mathrm{AT}(X,Y,T,S) \supset \mathrm{AT}(X,Y,\mathrm{S1}(T),\mathrm{W}(S))$$
$$\mathrm{AT}(X,Y,T,S) \supset \mathrm{AT}(X,\mathrm{S1}(Y),\mathrm{S1}(T),B(S))$$
$$\overline{\mathrm{AT}}(\mathrm{DM},\mathrm{S1}(0),T,S)$$

N(WI,EL)	N(LA,CH)	N(LI,OM)
N(EL,SLC)	N(CH,NP)	N(OM,DM)
N(SLC,RS)	N(NP,GI)	
N(RS,LA)	N(GI,LI)	

Мы использовали здесь такие сокращения:

WI:	Виннемака	NP:	Норт-Платт
EL:	Элко	GI:	Грэнд-Айлэнд
SLC:	Солт-Лейк-Сити	LI:	Линкольн
RS:	Рок-Спрингз	OM:	Омаха
LA:	Ларами	DM:	Де-Мойн
CH:	Шайенн		

Кроме того, к множеству входных дизъюнктов мы добавим еще ряд дизъюнктов, не имеющих отношения к задаче:

$$\mathrm{PLUS}(X,Y,Z) \supset \mathrm{PLUS}(S1(X),Y,S1(Z))$$
$$\mathrm{PLUS}(0,X,X)$$

$\mathrm{PLUS}(X,Y,Z) \supset \mathrm{PLUS}(Y,X,Z)$

$\overline{\mathrm{PLUS}}(0,0,1)$

$\mathrm{TIMES}(0,X,0)$

$\mathrm{TIMES}(X,Y,Z) \wedge \mathrm{PLUS}(Z,Y,W) \supset \mathrm{TIMES}(S1(X),Y,W)$

$\mathrm{TIMES}(X,Y,Z) \supset \mathrm{TIMES}(Y,X,Z)$

$\overline{\mathrm{TIMES}}(0,0,1)$

Предположим, что используется лок-резолюция с такой индексацией, чтобы отрицательные литеры отрезались раньше положительных. Кроме того, пусть индексация такова, что литеры $\overline{\mathrm{AT}}(X1,Y,T,S)$ из дизъюнктов 2 и 3 отрезаются первыми, $\overline{\mathrm{N}}(X1,X2)$ и $\overline{\mathrm{N}}(X2,X1)$ — вторыми и литеры $\mathrm{AT}(X2,Y,\mathrm{S1}(\mathrm{S1}(T)))$, $\mathrm{D}(X2,S))$ отрезаются последними. (Если литеры N индексированы так, чтобы отрезаться в первую очередь, то пространство поиска для лок-резолюции с абстракцией или без нее будет меньше.) Оказывается, что искомое доказательство существует на глубине 12 (считая входные дизъюнкты находящимися на глубине 0). Это происходит потому, что для поездки в соседний город необходимы два уровня и такие поездки должны быть совершены пять раз. Кроме того, один уровень требуется для покупки хлеба и один уровень — для получения NIL. Пусть a_n — число положительных единичных дизъюнктов, выводимых с помощью лок-резолюции на уровне n. Тогда $a_n = 2a_{n-1} + 2a_{n-2}$ при $n > 2$, $a_0 = 1$ и $a_1 = 2$. Данная рекурсия объясняется тем, что имеются два действия, которые занимают один уровень (покупка хлеба и ожидание), и два действия, которые занимают два уровня (поездка в соседние города). Поэтому общее число единичных дизъюнктов, выводимых на первых 11 уровнях, составляет 78 741. На 12-ом уровне будет выведено 136 382 новых единичных дизъюнкта, на два меньше, чем можно было ожидать, поскольку две последовательности действий уже достигли пределов карты. Вероятно, около одной седьмой части этих 136 382 единичных дизъюнктов будет выведено на 12-ом уровне, прежде чем появится NIL, так как NIL может быть получен шестью способами. Всего поисковое пространство должно содержать тогда около 100 000 положительных единичных дизъюнктов при условии, что используется метод поиска в ширину. Здесь не учитываются дизъюнкты, выводимые из входных дизъюнктов PLUS и TIMES. Несколько позднее мы приведем часть поискового пространства этой задачи.

Для данного примера мы используем несколько m-абстракций с согласованием. Мы рассматриваем абстракции, которые вычеркивают все аргументы, кроме одного, у AT и все аргументы у PLUS и TIMES. Естественно предположить, что эти

абстракции могут порождаться компьютером полностью автоматически. Так как AT имеет четыре аргумента, мы получаем четыре абстракции. Первая абстракция рассматривает только город, в котором вы находитесь, вторая — число имеющихся у вас батонов хлеба, третья рассматривает только время и четвертая — ситуацию. Предположим, что во всех четырех абстрактных пространствах уже произведена лок-резолюция методом поиска в ширину. Поисковое пространство для четвертой абстракции будет иметь примерно такие же размеры, как и для обычной резолюции. Этот факт, однако, может быть обнаружен автоматически, и рассматриваемая абстракция может быть отброшена, когда пространство поиска начнет безудержно расти. В первой абстракции будет один город на глубине 0, один город на глубине 1, три города на глубине 2, три города на глубине 3 и т. д. — всего 83 положительных единичных дизъюнкта в поисковом пространстве, считая каждый дизъюнкт для каждой глубины отдельно. NIL может быть получен на глубине 11 и 12. Во второй абстракции будет 0 батонов хлеба на глубине 0; 0 или 1 батон на глубине 1; 0, 1 или 2 батона на глубине 2 и т. д. — всего 91 положительный единичный дизъюнкт в поисковом пространстве. NIL во второй абстракции может быть получен на любой глубине от 2 до 12. В третьей абстракции будет $t = 0$ на глубине 0, $t = 1$ на глубине 1, $t = 2$ на глубине 2 и т. д. — всего в поисковом пространстве 13 положительных единичных дизъюнктов. В этой абстракции NIL может быть выведен на любой глубине от 1 до 12. В четвертой абстракции пространство поиска будет содержать около 215 000 положительных единичных дизъюнктов, но эта абстракция использоваться не будет. Мы не рассматриваем дизъюнкты PLUS и TIMES; поисковые пространства в этом случае для всех абстракций будут содержать 38 дизъюнктов (включая NIL и положительные единичные дизъюнкты), поскольку PLUS и TIMES могут быть выведены на произвольной глубине, а NIL — на любой глубине, кроме 0. Однако все эти дизъюнкты могут быть отброшены на основании критерия поддержки, так как ни одна из абстракций AT(CH,0,0,*S*0) или $\overline{\text{AT}}$(DM,S1(0),*t*,*s*) не имеет вхождений в эти доказательства. Сказанное выше иллюстрирует, как абстракции позволяют избавиться от не имеющих отношения к делу выводов.

Посмотрим теперь, сколько положительных единичных дизъюнктов останется в абстрактных пространствах после того, как будут вычеркнуты все дизъюнкты, не участвующие в доказательстве NIL на глубине 11 или 12. (Доказательства NIL на меньших глубинах можно не рассматривать, потому что они отсутствуют в первой абстракции.) В первой абстракции на каждой глубине останется только по одному положительному

единичному дизъюнкту — всего 12 положительных единичных дизъюнктов в поисковом пространстве. Во второй абстракции на глубине 0 останется один положительный единичный дизъюнкт, на глубине 1 — два, на глубине 2 — два, ..., на глубине 10 — два и один дизъюнкт, на глубине 11 — всего 22 дизъюнкта. В третьей абстракции останутся 12 из 13 положительных единичных дизъюнктов; отброшен будет только один дизъюнкт на глубине 12. Тот факт, что в данном случае остается бóльшая часть абстракций, может быть автоматически обнаружен как свидетельство того, что третья абстракция не принесет особой пользы в уменьшении поискового пространства.

Наконец, при поиске в ширину из исходного множества входных дизъюнктов, направляемом двумя первыми абстракциями, будут существенны только 6 доказательств, полученных на 12-ом уровне (и зависящих от того, в каком городе вы покупаете хлеб). Поисковое пространство будет иметь 27 положительных единичных дизъюнктов.

Данный пример показывает возможности абстракций в уменьшении размеров поискового пространства. Остается определить, насколько часто подобные уменьшения будут иметь место. Хотя приведенная задача по сути своей не трудна и может быть легко решена методами поиска в пространстве состояний, представляет интерес способность стратегий доказательства теорем решать такие задачи. Заметим, что в более сложных примерах (содержащих, скажем, 50 городов) уменьшение поискового пространства по сравнению с неограниченной лок-резолюцией было бы еще более впечатляющим. При использовании подходящих эвристик SL-резолюция может довольно успешно справиться с этим примером. Однако более сложные примеры для работы SL-резолюции потребовали бы тщательного выбора эвристик, тогда как абстракция, по-видимому, успешно работала бы независимо от выбираемой карты. Весьма интересной представляется комбинация SL-резолюции и абстракции.

Если бы мы изменили задачу, потребовав прибытия в Де-Мойн в момент $t = 50$, то даже абстракция имела бы большое поисковое пространство, так как существует слишком много решений, получаемых вставкой задержек в различных городах и возвращением в ранее посещенные города. Для таких задач поиск в глубину кажется более предпочтительным, чем поиск в ширину. В этом случае мы все время выполняем резолюции на самой большой глубине. Указанная стратегия довольно скоро привела бы к решению рассматриваемой задачи. Часто может оказаться полезным просматривать абстрактные пространства методом поиска в глубину и пытаться искать доказательства из исходного множества входных дизъюнктов даже прежде, чем закончится полный просмотр абстрактных пространств.

Исходное поисковое пространство

Мы сейчас покажем несколько первых уровней пространства поиска из входных дизъюнктов предыдущей задачи, используя лок-резолюцию указанным выше способом. Входные дизъюнкты индексированы следующим образом:

1. $_1\overline{\mathrm{AT}}(X1,Y,T,S) \vee {}_2\overline{\mathrm{N}}(X1,X2)$
   $\vee {}_3\mathrm{AT}(X2,Y,\mathrm{S1}(\mathrm{S1}(T)),\mathrm{D}(X2,S))$
2. $_4\overline{\mathrm{AT}}(X1,Y,T,S) \vee {}_5\overline{\mathrm{N}}(X2,X1)$
   $\vee {}_6\mathrm{AT}(X2,Y,\mathrm{S1}(\mathrm{S1}(T)),\mathrm{D}(X2,S))$
3. $_7\overline{\mathrm{AT}}(X,Y,T,S) \vee {}_8\mathrm{AT}(X,Y,\mathrm{S1}(T),\mathrm{W}(S))$
4. $_9\overline{\mathrm{AT}}(X,Y,T,S) \vee {}_{10}\mathrm{AT}(X,\mathrm{S1}(Y),\mathrm{S1}(T),\mathrm{B}(S))$
5. $_{11}\mathrm{AT}(\mathrm{CH},0,0,S0)$
6. $_{12}\overline{\mathrm{AT}}(\mathrm{DM},\mathrm{S1}(0),T,S)$

Лок-резольвенты первого уровня:

7. $_2\overline{\mathrm{N}}(\mathrm{CH},X2) \vee {}_3\mathrm{AT}(X2,0,\mathrm{S1}(\mathrm{S1}(0)),\mathrm{D}(X2,S0))$ — 1, 5
8. $_5\overline{\mathrm{N}}(X2,\mathrm{CH}) \vee {}_6\mathrm{AT}(X2,0,\mathrm{S1}(\mathrm{S1}(0)),\mathrm{D}(X2,S0))$ — 2, 5
9. $_8\mathrm{AT}(\mathrm{CH},0,\mathrm{S1}(0),\mathrm{W}(S0))$ — 3, 5
10. $_{10}\mathrm{AT}(\mathrm{CH},\mathrm{S1}(0),\mathrm{S1}(0),\mathrm{B}(S0))$ — 4, 5

Лок-резольвенты второго уровня:

11. $_3\mathrm{AT}(\mathrm{NP},0,\mathrm{S1}(\mathrm{S1}(0)),\mathrm{D}(\mathrm{NP},S0))$ — 7, N

(Отметим, что мы обозначаем все дизъюнкты вида $\mathrm{N}(x,y)$ через N.)

12. $_6\mathrm{AT}(\mathrm{LA},0,\mathrm{S1}(\mathrm{S1}(0)),\mathrm{D}(\mathrm{LA},S0))$ — 8, N
13. $_8\mathrm{AT}(\mathrm{CH},0,\mathrm{S1}(\mathrm{S1}(0)),\mathrm{W}(\mathrm{W}(S0)))$ — 3, 9
14. $_{10}\mathrm{AT}(\mathrm{CH},\mathrm{S1}(0),\mathrm{S1}(\mathrm{S1}(0)),\mathrm{B}(\mathrm{W}(S0)))$ — 4, 9
15. $_8\mathrm{AT}(\mathrm{CH},\mathrm{S1}(0),\mathrm{S1}(\mathrm{S1}(0)),\mathrm{W}(\mathrm{B}(S0)))$ — 3, 10
16. $_{10}\mathrm{AT}(\mathrm{CH},\mathrm{S1}(\mathrm{S1}(0)),\mathrm{S1}(\mathrm{S1}(0)),\mathrm{B}(\mathrm{B}(S0)))$ — 4, 10
17. $_2\overline{\mathrm{N}}(\mathrm{CH},X2) \vee {}_3\mathrm{AT}(X2,0,\mathrm{S1}(\mathrm{S1}(\mathrm{S1}(0))),\mathrm{D}(X2,\mathrm{W}(S0)))$ — 1, 9
18. $_5\overline{\mathrm{N}}(X2,\mathrm{CH}) \vee {}_6\mathrm{AT}(X2,0,\mathrm{S1}(\mathrm{S1}(\mathrm{S1}(0))),\mathrm{D}(X2,\mathrm{W}(S0)))$ — 2, 9
19. $_2\overline{\mathrm{N}}(\mathrm{CH},X2) \vee {}_3\mathrm{AT}(X2,\mathrm{S1}(0),\mathrm{S1}(\mathrm{S1}(\mathrm{S1}(0))),$
    $\mathrm{D}(X2,\mathrm{B}(S0)))$ — 1, 10
20. $_5\overline{\mathrm{N}}(X2,\mathrm{CH}) \vee {}_6\mathrm{AT}(X2,\mathrm{S1}(0),\mathrm{S1}(\mathrm{S1}(\mathrm{S1}(0))),$
    $\mathrm{D}(X2,B(S0)))$ — 2, 10

Пример единичных дизъюнктов на третьем уровне:

21. $_3\mathrm{AT}(\mathrm{NP},0,\mathrm{S1}(\mathrm{S1}(\mathrm{S1}(0))),\mathrm{D}(\mathrm{NP},\mathrm{W}(S0)))$ — 17, N
22. $_6\mathrm{AT}(\mathrm{LA},0,\mathrm{S1}(\mathrm{S1}(\mathrm{S1}(0))),\mathrm{D}(\mathrm{LA},\mathrm{W}(S0)))$ — 18, N

23. $_3$AT(NP,S1(0),S1(S1(0))),D(NP,B($S0$))) 19, N
24. $_6$AT(LA,S1(0),S1(S1(S1(0))),D(LA,B($S0$))) 20, N
25. $_8$AT(NP,0,S1(S1(S1(0))),W(D(NP,$S0$))) 3, 11
26. $_{10}$AT(NP,S1(0),S1(S1(S1(0))),B(D(NP,$S0$))) 4, 11
27. $_8$AT(CH,0,S1(S1(S1(0))),W(W(W($S0$)))) 3, 13
28. $_{10}$AT(CH,S1(0),S1(S1(S1(0))),B(W(W($S0$)))) 4, 13

Первая абстракция

Относительно первой абстракции мы имеем следующее абстрактное поисковое пространство. Абстрактные входные дизъюнкты — это

1. $_1\overline{\text{AT}}(X1) \vee {}_2\overline{\text{N}}(X1,X2) \vee {}_3\text{AT}(X_2)$
2. $_4\overline{\text{AT}}(X1) \vee {}_5\overline{\text{N}}(X2,X1) \vee {}_6\text{AT}(X_2)$
3. $_7\overline{\text{AT}}(X) \vee {}_8\text{AT}(X)$
4. $_9\overline{\text{AT}}(X) \vee {}_{10}\text{AT}(X)$ (не нужен)
5. $_{11}$AT(CH)
6. $_{12}\overline{\text{AT}}$ (DM)

Кроме того, как и выше, имеем много дизъюнктов вида $\text{N}(x,y)$. В доказательствах мы будем обозначать их через N.

Лок-резольвенты первого уровня:

7. $_2\overline{\text{N}}(\text{CH},X2) \vee {}_3\text{AT}(X2)$ 1, 5
8. $_5\overline{\text{N}}(X_2,\text{CH}) \vee {}_6\text{AT}(X2)$ 2, 5
9. $_8$AT(CH) 3, 5

Хотя этот дизъюнкт и совпадает с дизъюнктом 5, его следует сохранить, поскольку он находится на другом уровне.

Лок-резольвенты второго уровня:

10. $_3$AT(NP) 7, N
11. $_6$AT(LA) 8, N
12. $_8$AT(CH) 3, 9
13. $_2\overline{\text{N}}(\text{CH},X2) \vee {}_3\text{AT}(X2)$ 1, 9
14. $_5\overline{\text{N}}(X2,\text{CH}) \vee {}_6\text{AT}(X2)$ 2, 9

Лок-резольвенты третьего уровня:

15. $_2\overline{\text{N}}(\text{NP},\text{X2}) \vee {}_3\text{AT}(\text{X2})$ 1, 10
16. $_5\overline{\text{N}}(\text{X2},\text{NP}) \vee {}_6\text{AT}(\text{X2})$ 2, 10
17. $_2\overline{\text{N}}(\text{LA},\text{X2}) \vee {}_3\text{AT}(\text{X2})$ 1, 11
18. $_5\overline{\text{N}}(\text{X2},\text{LA}) \vee {}_6\text{AT}(\text{X2})$ 2, 11

19. ${}_2\overline{\mathrm{N}}(\mathrm{CH},\mathrm{X2}) \vee {}_3\mathrm{AT}(\mathrm{X2})$ 1, 12
20. ${}_5\overline{\mathrm{N}}(\mathrm{X2},\mathrm{CH}) \vee {}_6\mathrm{AT}(\mathrm{X2})$ 2, 12
21. ${}_8\mathrm{AT}(\mathrm{NP})$ 3, 10
22. ${}_8\mathrm{AT}(\mathrm{LA})$ 3, 11
23. ${}_8\mathrm{AT}(\mathrm{CH})$ 3, 12
24. ${}_3\mathrm{AT}(\mathrm{NP})$ 13, N
25. ${}_6\mathrm{AT}(\mathrm{LA})$ 14, N

Заметим, что дизъюнкты 24 и 25 — повторения дизъюнктов 21 и 22 и могут поэтому быть отброшены (поскольку все они находятся на одном и том же уровне).

Положительными единичными дизъюнктами, выводимыми на четвертом уровне, являются: AT(RS), AT(LA), AT(CH), AT(NP) и AT(GI). Кроме них, выводимы еще шесть неединичных дизъюнктов.

Вторая абстракция

Относительно второй абстракции имеются следующие входные дизъюнкты (для простоты здесь и далее индексы дизъюнктов опущены):

1. $\overline{\mathrm{AT}}(Y) \vee \overline{\mathrm{N}}(X1,X2) \vee \mathrm{AT}(Y)$
2. $\overline{\mathrm{AT}}(\mathrm{Y}) \vee \overline{\mathrm{N}}(\mathrm{X2},\mathrm{X1}) \vee \mathrm{AT}(\mathrm{Y})$ (такой же, как 1)
3. $\overline{\mathrm{AT}}(\mathrm{Y}) \vee \mathrm{AT}(\mathrm{Y})$
4. $\overline{\mathrm{AT}}(\mathrm{Y}) \vee \mathrm{AT}(\mathrm{S1}(\mathrm{Y}))$
5. $\mathrm{AT}(0)$
6. $\overline{\mathrm{AT}}(\mathrm{S1}(0))$

Лок-резольвенты первого уровня:

7. $\overline{\mathrm{N}}(\mathrm{X1},\mathrm{X2}) \vee \mathrm{AT}(0)$ 1, 5
8. $\mathrm{AT}(0)$ 3, 5
9. $\mathrm{AT}(\mathrm{S1}(0))$ 4, 5

Лок-резольвенты второго уровня:

10. $\mathrm{AT}(0)$ 7, N
11.—13. Такие же, как 7—9
14. $\overline{\mathrm{N}}(\mathrm{X1},\mathrm{X2}) \vee \mathrm{AT}(\mathrm{S1}(0))$ 1, 9
15. $\mathrm{AT}(\mathrm{S1}(0))$ 3, 9
16. $\mathrm{AT}(\mathrm{S1}(\mathrm{S1}(0))$ 4, 9

Планы доказательств

Ниже приводится план, образованный первой абстракцией на 11-м уровне:

1. AT(CH)
2. $\overline{\text{N}}$(CH,*X*2) $\vee$ AT(*X*2)
3. AT(NP)
4. $\overline{\text{N}}$(NP,*X*2) $\vee$ AT(*X*2)
5. AT(GI)

   . . .

11. AT(DM)
12. NIL

В результате выполнения этого плана мы прибудем в Де-Мойн без хлеба. На 12-м уровне первая абстракция дает следующие планы:

1. AT(CH)
2. $\overline{\text{N}}$(CH,*X*2) $\vee$ AT(*X*2)
3. AT(NP)
4. AT(NP)
5. $\overline{\text{N}}$(*N*P,*X*2) $\vee$ AT(*X*2)
6. AT(GI)

   . . .

12. AT(DM)
13. NIL

Шаг 4 в этом плане означает задержку на единицу времени в городе Норт-Платт. Такую задержку можно осуществить также и в пяти других местах. Используя вторую абстракцию, мы можем увидеть, что остановку следует посвятить покупке хлеба, а не просто ожиданию в течение единицы времени. Таким образом, вторая абстракция совершенствует планы, образованные первой. Теперь мы представим образец плана, даваемого второй абстракцией на 12-уровне.

1. AT(0)
2. AT(0)
3. $\overline{\text{AT}}$(*X*1,*Y*1) $\vee$ AT(0)
4. AT(0)
5. AT(S1(0))
6. $\overline{\text{AT}}$(*X*1,*Y*1) $\vee$ AT(S1(0))

   . . .

12. AT(S1(0))
13. NIL

Пятый шаг здесь означает покупку хлеба, а второй — ожидание в течение единицы времени.

6. ОГРАНИЧЕННЫЕ m-ДИЗЪЮНКТЫ

Один из недостатков m-дизъюнктов состоит в том, что этих дизъюнктов слишком много. Множество обычных пропозициональных дизъюнктов с k различными предикатными символами конечно, тогда как множество пропозициональных m-дизъюнктов с этими же символами уже бесконечно. Сказанное выше может привести к увеличению размеров поискового пространства для различных стратегий доказательства теорем, основанных на абстракции. Сейчас мы покажем, как можно до некоторой степени преодолеть указанную трудность, сохраняя при этом преимущества m-абстракций. Общая идея заключается в том, чтобы хранить меньше информации о числе вхождений литер в m-дизъюнкт. Например, мы можем указать, что определенная литера входит в m-дизъюнкт по крайней мере дважды.

Определение. *Ограниченное* мультимножество — это такое мультимножество, в котором сложности элементов могут быть равными 0, 1, 2, ..., $b-1$ или ∞ для некоторой границы b. Сложность ∞ означает, что элемент имеет по крайней мере b вхождений. Мы называем b границей мультимножества. Из практических соображений считается, что элемент с границей ∞ имеет бесконечно много вхождений в мультимножество. Мы рассматриваем только границы $b \geqslant 1$.

Определение. Если A — ограниченное мультимножество, то $\mathrm{Set}(A)$ — это множество $\{x\colon \mathrm{mult}(X,A) > 0\}$.

Мы следующим образом определим ограниченные операции сложения $+^b$ и вычитания $-^b$ для ограниченных целых чисел:

$x +^b \infty = \infty +^b x = \infty$ для любого x,

$x +^b y = x + y$, если $x \neq \infty$, $y \neq \infty$ и $x + y < b$,

$x +^b y = \infty$, если $x \neq \infty$, $y \neq \infty$ и $x + y \geqslant b$,

$x -^b y = x - y$, если $x \neq \infty$ и $y \leqslant x$,

$x -^b y = 0$, если $x \neq \infty$ и ($y = \infty$ или $y \geqslant x$),

$\infty -^b 0 = \infty$,

$\infty -^b x = \{\infty, b-1, b-2, \ldots, b-x\}$, если $x \neq \infty$, $x \neq 0$,

$\infty -^b \infty = \{\infty, b-1, b-2, \ldots, 2, 1, 0\}$.

Смысл этих множеств состоит в том, что операция может давать более одного возможного результата. Так, $\infty -^b x$ может принимать любое значение между $b - x$ и ∞, если $x \neq \infty$ и $x \neq 0$. Приведенные определения получаются следующим образом. Пусть $\Phi_b(x)$ есть ∞, если $x \geqslant b$ и $\Phi_b(x) = x$, если $0 \leqslant x < b$. Тогда если $x + y = z$ для обычных целых неотрицательных чисел x, y и z, то мы говорим, что $\Phi_b(x) +^b \Phi_b(y) = \Phi_b(z)$. Если же $x \dot{-} y = z$, то мы говорим, что $\Phi_b(x) -^b -^b \Phi_b(y) = \Phi_b(z)$. Здесь $x \dot{-} y$ по определению есть $\max(0, x - y)$.

Определение. Если A и B — ограниченные мультимножества с границей b, то $A \cup B$ и $A - B$ определяются следующим образом:

$$\operatorname{mult}(x, A \cup B) = \operatorname{mult}(x, A) +^b \operatorname{mult}(x, B)$$
$$\operatorname{mult}(x, A - B) = \operatorname{mult}(x, A) -^b \operatorname{mult}(x, B)$$

Заметим, что $\operatorname{Set}(A \cup B) = \operatorname{Set}(A) \cup \operatorname{Set}(B)$ и $\operatorname{Set}(A) - \operatorname{Set}(B) \subseteq \operatorname{Set}(A - B)$.

Пример. Допустим, что $b = 2$, A есть $\{\infty * P, 1 * Q\}$ и B есть $\{1 * P, 1 * Q, 1 * R\}$. Тогда

$$A \cup B = \{\infty * P, \infty * Q, 1 * R\},$$
$$A - B = \{\infty * P\} \text{ или } \{1 * P\},$$
$$B - A = \{1 * R\}.$$

Если имеется обычное мультимножество C, то пусть $\Phi_b(C)$ определяется посредством $\operatorname{mult}(x, \Phi_b(C)) = \Phi_b(\operatorname{mult}(x, C))$. Таким образом, $\Phi_b(C)$ — ограниченное мультимножество с границей b.

Ограниченные операции на мультимножествах определены так, что если $C1$ и $C2$ — обычные мультимножества, то $\Phi_b(C1) \cup \Phi_b(C2) = \Phi_b(C1 \uplus C2)$ и $\Phi_b(C1) - \Phi_b(C2) = \Phi_b(C1 - C2)$. Заметим, что, как и прежде, разность множеств — «недетерминированная» операция.

Определение. Если A — ограниченное мультимножество, а f — функция на элементах A, то $f(A)$ определяется посредством $\operatorname{mult}(y, f(A)) = \Sigma_{f(x)=y} \operatorname{mult}(x, A)$, где символ суммы обозначает ограниченное сложение. В частности, если A — ограниченное мультимножество литер и α — подстановка, то $A\alpha$ определяется указанным выше способом. Так, для ограниченных мультимножеств с границей 2 $\{1 * P(z), 1 * P(a), 1 * Q(z)\}\{z \leftarrow a\} = \{\infty * P(a), 1 * Q(a)\}$. Здесь $\{z \leftarrow a\}$ — подстановка, заменяющая z на a.

Определение. *Ограниченный* m-дизъюнкт — это ограниченное мультимножество литер.

6.1. Ограниченная m-резолюция

Определение. Пусть $C1$ и $C2$ — ограниченные m-дизъюнкты, $A1 \subseteq C1$ и $A2 \subseteq C2$. Пусть, далее, $\alpha 1$ и $\alpha 2$ — наиболее общие подстановки, для которых существует литера L, такая что $\mathrm{Set}(A1\alpha 1) = \{L\}$ и $\mathrm{Set}(A2\alpha 2) = \{\bar{L}\}$. Тогда $(C1 - A1)\alpha 1 \cup (C2 - A2)\alpha 2$ называется ограниченной m-резольвентой $C1$ и $C2$.

Заметим, что обычные дизъюнкты можно рассматривать как ограниченные m-дизъюнкты, хотя операция резолюции будет уже другой.

Одна из причин применения ограниченной m-резолюции состоит в том, что она является приближением к обычной m-резолюции. Иными словами, $B3$ является ограниченной m-резольвентой $B1$ и $B2$ тогда и только тогда, когда существуют обычные m-дизъюнкты $C1$, $C2$ и $C3$, такие что $C3$ есть m-резольвента от $C1$ и $C2$, $\Phi_b(C1) = B1$, $\Phi_b(C2) = B2$ и $\Phi_b(C3) = B3$. Здесь, как обычно, l — граница.

Примеры. Допустим, что $B1 = \{1 * P, 1 * Q\}$ и $B2 = \{\infty * \bar{P}\}$ — ограниченные m-дизъюнкты с границей 2. Их ограниченными m-резольвентами являются следующие дизъюнкты:

$$\{\infty * \bar{P},\ 1 * Q\},\ \{1 * \bar{P},\ 1 * Q\}.$$

Пусть $B1 = \{1 * P,\ 1 * Q\}$ и $B2 = \{1 * \bar{P},\ 1 * Q\}$ с границей 2, как и прежде. Единственной их ограниченной m-резольвентой будет $\{\infty * Q\}$. Пусть $B1 = \{\infty * P\}$, $B2 = \{\infty * \bar{P},\ 1 * Q\}$, а $b = 2$. Тогда их ограниченными m-резольвентами будут следующие дизъюнкты:

$\{\infty * P,\ \infty * \bar{P},\ 1 * Q\}$,
$\{\infty * P,\ 1 * \bar{P},\ 1 * Q\}$,
$\{\infty * P,\ 1 * Q\}$,
$\{1 * P,\ \infty * \bar{P},\ 1 * Q\}$,
$\{1 * P,\ 1 * \bar{P},\ 1 * Q\}$,
$\{1 * P,\ 1 * Q\}$,
$\{\infty * \bar{P},\ 1 * Q\}$,
$\{1 * \bar{P},\ 1 * Q\}$,
$\{1 * Q\}$.

Теорема 4.1. *Предположим, что S — множество мультидизъюнктов и C' выводим из S m-резолюцией. Пусть* $\Phi_b(S) =$

$= \{\Phi_b(C)\colon C \in S\}$, *т. е.* $\Phi_b(S)$ — *множество ограниченных мультидизъюнктов с границей* b. *Тогда* $\Phi_b(C')$ *выводим из* $\Phi_b(S)$ *ограниченной* m-*резолюцией.*

Следствие. Предположим, что S — множество обычных дизъюнктов и дизъюнкт C выводим из S обычной резолюцией. Тогда существует дизъюнкт C', выводимый из S ограниченной m-резолюцией и такой, что $\mathrm{Set}(C') = C$. (Напомним, что граница больше или равна единице.)

Таким образом, m-резолюционные и обычные резолюционные доказательства могут быть преобразованы в ограниченные m-резолюционные доказательства. Фактически это преобразование можно выполнить так, чтобы ограниченное m-резолюционное доказательство имело ту же форму, что и исходное. Кроме того, можно показать, что множество дизъюнктов S противоречиво тогда и только тогда, когда NIL (пустой дизъюнкт) выводим ограниченной m-резолюцией из S. С этой целью мы рассматриваем обычные дизъюнкты как ограниченные мультидизъюнкты, в которых каждая литера имеет сложность 1 (или ∞, если $b = 1$). Более того, если C' выводим ограниченной m-резолюцией из множества дизъюнктов S, то существует дизъюнкт C, выводимый обычной резолюцией из S, причем C поглощает $\mathrm{Set}(C')$.

6.2. Ограниченные m-абстракции

Определение. *Функция ограниченной* m-абстракции — это функция f, отображающая обычные мультидизъюнкты в множества ограниченных мультидизъюнктов и удовлетворяющая следующим свойствам:

(1) Если $C3$ — m-резольвента $C1$ и $C2$ и $D3 \in f(C3)$, то существуют $D1 \in f(C1)$ и $D2 \in f(C2)$, такие что $D3$ является примером ограниченной m-резольвенты $D1$ и $D2$.

(2) $f(\mathrm{NIL}) = \{\mathrm{NIL}\}$.

Заметим, что $D1$, $D2$ и $D3$ — ограниченные мультидизъюнкты, а $C1$, $C2$ и $C3$ — обычные. Заметим также, что функция Φ_b сама является функцией ограниченной m-абстракции с границей b. Кроме того, если f — функция обычной m-абстракции, то $\Phi_b \circ f$ — функция ограниченной m-абстракции. Таким способом мы можем получать функции ограниченной m-абстракции из всех описанных ранее функций обычной m-абстракции.

Аналогичным образом мы могли бы определить функции абстракции из ограниченных мультидизъюнктов в ограниченные мультидизъюнкты. Мы могли бы также доказать соответствующие теоремы о свойствах замкнутости функций ограниченной m-абстракции относительно объединения и композиции.

Все поисковые стратегии для m-абстракций можно с таким же успехом применять к функциям ограниченной m-абстракции. Например, в поисках доказательства можно одновременно использовать несколько функций ограниченной m-абстракции. Ограниченные m-абстракции могут также использоваться для проверки, является ли некоторый дизъюнкт логическим следствием множества дизъюнктов. Некоторые ограниченные m-абстракции особенно полезны. Если, к примеру, f — пропозициональная или семантическая m-абстракция с конечной областью, то $\Phi_b \circ f$ — функция ограниченной m-абстракции с *конечной областью значений*, т. е. множество $\{D\colon (\exists C)D = (\Phi_b \circ f)(C)\}$ *конечно*. Следовательно, мы можем полностью перечислить это множество так же, как и множество $\{\langle B1, B2, B3\rangle\colon B3$ является ограниченной m-резольвентой от $B1$ и $B2\}$. Различные поисковые стратегии могут использовать эту информацию, не вычисляя ее каждый раз заново для каждой глубины. Кроме того, с помощью подходящего хэш-кодирования или схем индексации эту информацию можно компактно хранить и эффективно восстанавливать. Таким образом, мы получаем многие преимущества мультидизъюнктов вместе с дополнительной выгодой, которую дает *конечное* абстрактное пространство. Разумеется, поисковые стратегии в данном случае менее ограничительные, чем с m-дизъюнктами, но это, по-видимому, с лихвой компенсируется конечностью абстрактного пространства. Увеличение границы будет давать более ограничительную стратегию поиска за счет увеличения размеров абстрактного пространства. Значения границы, близкие к 2, явились бы, видимо, наилучшими для большинства применений. Граница, равная 1, действительно слишком мала, поскольку она не позволяет отличать одно вхождение литеры (обычный случай) от нескольких.

7. СЕМАНТИЧЕСКИЕ АБСТРАКЦИИ С РАЗБИЕНИЕМ

Из одних семантических абстракций возможно строить новые путем разбиения области абстракции. Получающаяся в результате этого техника доказательства напоминает человеческие рассуждения с помощью диаграмм. В частности, семантические абстракции с разбиением соответствуют неполностью определенным диаграммам наподобие тех, которые можно рисовать на доске из точек и каких-либо каракулей, обозначающих неопределенные части диаграммы. По-видимому, это соответствует такому способу рассуждений, который люди (автор по крайней мере) используют для доказательства реальных теорем.

Напомним, что семантическая абстракция отображает дизъюнкты на дизъюнкты вида $\{L_1, \ldots, L_k\}$, где каждая литера L_i суть $P(a_1, \ldots, a_n)$ или $\neg P(a_1, \ldots, a_n)$, а a_i — элементы об-

ласти интерпретации $\mathscr{D}$. Пусть $\mathscr{P}$ — разбиение $\mathscr{D}$. Каждой литере L мы сопоставим литеру $f_p(L)$, определяемую следующим образом. Если L имеет вид $P(a_1, \ldots, a_n)$, то $f_p(L)$ есть $P(A_1, \ldots, A_n)$, где $a_i \in A_i$ и A_i — смежные классы разбиения $\mathscr{P}$. Если же L имеет вид $\neg P(a_1, \ldots, a_n)$, то $f_p(L)$ есть $\neg P(A_1, \ldots, A_n)$, где A_i имеют такой же смысл, как и выше. Наконец, дизъюнкту $C = \{L_1, \ldots, L_k\}$ мы сопоставим дизъюнкт $f_p(C) = \{L'_1, \ldots, L'_k\}$, где L'_i есть $f_p(L_i)$, $1 \leqslant i \leqslant k$.

Используя теорему 2.2, нетрудно показать, что если g — семантическая абстракция с областью $\mathscr{D}$, то $f_p \circ g$ — также абстракция. Однако $f_p \circ g$ может быть конечной (точнее, множество $f_p \circ g(C)$ может быть конечным), даже если g этим свойством не обладает. Таким способом конечные абстракции довольно естественно можно получать из семантических абстракций. Мы назовем $f_p \circ g$ семантической абстракцией с разбиением. Мы запоминаем не отдельные элементы области интерпретации, а только лишь смежные классы из $\mathscr{P}$, которым они принадлежат. Например, мы можем запоминать только класс эквивалентности целых чисел по модулю некоторого простого числа. Или при рассмотрении неравенств мы можем разбить целые числа на «большие» и «маленькие». Для получения последовательности незначительно меняющихся абстракций мы можем выбирать все более и более мелкие разбиения. Таким образом, мы получаем абстракции, которые соответствуют неполностью определенным диаграммам. Значение данного результата состоит не в том, что мы имеем формализм для подобных диаграмм, а в том, что этот формализм достаточно общий и ведет к общей стратегии доказательства теорем. Как и прежде, мы можем определить семантические m-абстракции с разбиением. Возможны также и семантические ограниченные m-абстракции с разбиением.

8. ДРУГИЕ АБСТРАКЦИИ

Пусть f — основная абстракция, а Φ — произвольное отображение литер, такое что $\Phi(\bar{L}) = \overline{\Phi(L)}$. Пусть, далее, g — абстракция, определенная для основных дизъюнктов C посредством $g(C) = \{\Phi(C)\}$. (Здесь, как обычно, $\Phi(C) = \{\Phi(L)\colon L \in C\}$.) Тогда по теореме 2.2 $g \circ f$ — также абстракция. Таким способом мы можем показать, что семантические абстракции и семантические абстракции с разбиением являются абстракциями. Пусть L — основная литера вида $P(t_1, \ldots, t_n)$. Для получения семантических абстракций мы полагаем $\Phi(L)$ равным $P(a_1, \ldots, a_n)$, где a_i — значение t_i в данной интерпретации $\mathscr{I}$. Кроме того, мы определяем Φ так, чтобы $\Phi(\bar{L}) = \overline{\Phi(L)}$. Для семантических абстракций с разбиением $\Phi(L) = P(A_1, \ldots, A_n)$, где $a_i \in A_i$ и

A_i — смежный класс данного разбиения области интерпретации $\mathcal{I}$. Как обычно, выполняется также $\Phi(\bar{L}) = \overline{\Phi(L)}$. Мы можем развить эту идею и дальше.

Предположим, что $\mathcal{I}_1$ и $\mathcal{I}_2$ — две интерпретации с областями $\mathcal{D}_1$ и $\mathcal{D}_2$ соответственно. Пусть, как и выше, L — основная литера вида $P(t_1, \ldots, t_n)$, a_i и b_i — значения t_i в $\mathcal{I}_1$ и $\mathcal{I}_2$ соответственно, и пусть p — новый функциональный символ (представляющий «спаривание»). Определим Φ, положив $\Phi(L) = P(p(a_1, b_1), \ldots, p(a_n, b_n))$ и $\Phi(\bar{L}) = \overline{\Phi(L)}$. Мы получаем другую абстракцию, являющуюся в некотором смысле произведением двух семантических абстракций. Мы можем также положить $\Phi(L) = P(p(A_1, B_1), \ldots, p(A_n, B_n))$, где A_i и B_i — смежные классы некоторых разбиений областей $\mathcal{D}_1$ и $\mathcal{D}_2$ соответственно, $a_i \in A_i$ и $b_i \in B_i$, $1 \leqslant i \leqslant n$. Как обычно, мы полагаем $\Phi(\bar{L})$ равным $\overline{\Phi(L)}$. Подобным способом можно получить и много других абстракций. Например, мы можем взять произведение любого числа семантических абстракций. Аналогичным образом мы можем получить m-абстракции и ограниченные m-абстракции, основанные на таких отображениях литер Φ. Затем можно брать произведение двух семантических m-абстракций и т. д. Понятно, что существует огромное число возможных путей использования абстракций и связанных с ними понятий для получения полных стратегий доказательства теорем.

9. ВЫВОДЫ

Понятия абстракции, m-абстракции и ограниченной m-абстракции приводят к широкому спектру новых единообразных полных процедур доказательства для исчисления предикатов первого порядка. Вероятно, такие же стратегии с небольшими изменениями применимы и к логикам более высокого порядка. Все указанные стратегии используют упрощенное доказательство из упрощенного множества дизъюнктов (m-дизъюнктов, ограниченных m-дизъюнктов) для управления поиском доказательства из исходного множества дизъюнктов (m-дизъюнктов, ограниченных m-дизъюнктов). Эта техника является более общей, чем использование современных единообразных процедур доказательства. Иными словами, каждый вывод контролируется более осмысленным образом структурой задачи в целом, а не локальными свойствами участвующих в выводе дизъюнктов. К тому же при завершении поиска число абстрактных дизъюнктов более ограниченно, чем в середине, поскольку абстракция «целевого дизъюнкта» должна быть выведена за небольшое число шагов. Поэтому пространство поиска имеет тенденцию к уменьшению при увеличении глубины вывода до максималь-

ного значения. Кроме того, эти стратегии совместимы с любой традиционной полной стратегией резолюционного доказательства теорем. Далее, приведенные методы позволяют проводить поиск в глубину и образовывать подцели более естественным образом, чем это делает большинство резолюционных стратегий. На самом деле мы продолжаем разрабатывать другие методы, которые используют абстракции вместе с обратными рассуждениями и которые в большей степени полагаются на семантику, решая, какая из подцелей достижима.

Абстракции, основанные на конкретных интерпретациях, представляются особенно интересными, поскольку они приближаются к формализации идеи доказательства теоремы для конкретного примера — техники, часто используемой человеком. Многообещающими являются и абстракции, соответствующие интерпретациям с конечными областями, так как совместно с использованием ограниченных m-дизъюнктов они приводят к конечным абстрактным пространствам поиска. К тому же эти абстракции могут порождаться полностью автоматически.

Стратегии, основанные на «мультидизъюнктах» и абстракциях, оказываются более простыми и изящными по сравнению со стратегиями, базирующимися на обычных дизъюнктах и абстракциях. (Мультидизъюнкт — это мультимножество литер.) Подобные стратегии с мультидизъюнктами позволяют естественным образом использовать одновременно несколько абстракций. Комбинация мультидизъюнктов и абстракции является, по-видимому, значительным новым достижением. Другой многообещающей возможностью является использование нескольких уровней абстракции.

Для иллюстрации некоторых из предлагаемых стратегий приведены структурные программы. Но чтобы определить практическую ценность представленной здесь техники, необходим опыт реализации этих программ. Однако благодаря простоте и изяществу лежащих в их основе идей реализовывать эти стратегии должно быть относительно просто и легко.

Остается проделать еще большую работу по распространению понятия абстракции на другие системы правил вывода и на логики более высокого порядка. Можно ли, к примеру, применять абстракции для автоматического синтеза программ? Вполне возможно, что абстракции могут дать более быстрые системы доказательства теорем даже в исчислении высказываний. Кроме того, было бы желательно соединить абстракцию с более осмысленным использованием семантики и со стратегией для равенства. Можно исследовать также совместимость абстракции с такими традиционными стратегиями, как лок-резолюция. Наконец, мы планируем изучить также сочетание абстракции с обратными рассуждениями (от цели). Надежда

на появление более удачных систем доказательства теорем в результате применения абстракции основана на том, что абстракция представляется качественно отличной от рассматривающихся в прошлом видов стратегий доказательства, поскольку она ограничивает поиск с помощью использования общей информации о решаемой задаче. Привлекательны также универсальность этого подхода и возможность применять специальные знания о том, какие абстракции полезны для данного класса проблем.

Благодарности

Мне хотелось бы поблагодарить Марсию Коупли за квалифицированную и добросовестную перепечатку этой и других статей. Я высоко ценю помощь, оказанную Department of Computer Science at the University of Illinois, а также финансовую поддержку National Science Foundation. Замечания рецензентов помогли мне в улучшении изложения материала.

ЛИТЕРАТУРА

[1] Bledsoe W. W. Non-resolution theorem proving — Artificial Intelligence 9(1), (1977), 1—35.
[2] Boyer R. S. Locking, a restriction of resolution. — Thesis, Univ of Texas at Austin TX (1971).
[3] Chang C. L., Lee R. C. Symbolic logic and mechanical theorem proving. — Acad. Press, N. Y., 1973. [Имеется перевод: Чень Ч., Ли Р. Математическая логика и автоматическое доказательство теорем. — М.: Наука, 1983.]
[4] Henschen L., Wos L. Unit refutations and Horn sets. — J ACM 21 (1974), 590—605
[5] Kling R. E. A paradigm for reasoning by analogy. — Artificial Intelligence 2 (1971), 147—178.
[6] Kowalski R. The case for using equality axioms in automatic demonstration. — Proc. of the Symp. on Automatic Demonstration. Lecture Notes in Math. 125, Springer-Verlag, N. Y., 1970, p. 112—127.
[7] Luckham D. Refinement theorems in resolution theory.—Proc. of the Symp. on Automatic Demonstration. — Lecture Notes in Math. 125, Springer-Verlag, N. Y., 1970, p. 163—190
[8] Manyer J. C. Towards the use of analogy deductive tasks. — Univ. of Calif. at Santa Cruz (1979).
[9] Plaisted T. A. Abstraction mappings in mechanical theorem proving. — Proc. Fifth Conf. on Automated Deduction. Lecture Notes in Computer Sciences, 87, Springer-Verlag, N Y, 1980, p. 264—280.
[10] Reboh R., Raphael D., Yates R. A., Kling R. E., Nelarde C. Study of automatic theorem proving programs. — Technical Notes, 1975, Artificial Intelligence Center, Stanford Research Institute, Menlo Park, CA, 1972.
[11] Robinson J. A. Automatic deduction with hyper-resolution. — Int. J. Comput. Math. 1 (1965), 227—234.
[12] Robinson J. A. A review of automatic theorem proving. — Proc. Symp Appl. Math. Am. Math. Soc. 19 (1967), 1—18.
[13] Sacedoti E. D. Planning in hierarchy of abstraction spaces. — Artificial Intelligence 5 (1974), 115—135.
[14] Slagle J. R. Automatic theorem proving with renameable and semantic resolution. — J. ACM 14 (1967), 687—697.
[15] Wos L., Robinson G. A., Garson D. F. Efficiency and completeness of the set of support strategy in theorem proving. — J. ACM 12 (1965), 536—541,

Универсальная унификация и классификация эквациональных теорий [1)]

Й. Зикманн, П. Сабо [2)]

Однако, чтобы обобщать, нужен опыт ...

Гретцер У. Универсальная алгебра, 1968.

0. ОБОСНОВАНИЕ

Унификация двух термов относительно теории T равносильна решению уравнения в этой теории. Следует отметить, что математические исследования, посвященные решению уравнений в известных теориях, имеют столь же давнюю историю, как и сама математика. Эти исследования восходят к вавилонской математике (примерно 2000 лет до нашей эры) и продолжают оставаться в центре внимания математиков вплоть до сегодняшнего дня.

Универсальная унификация переносит эту деятельность на более абстрактный уровень: точно так же, как универсальная алгебра абстрагируется от определенных свойств, присущих конкретным алгебрам, и занимается исследованием черт, общих для всех алгебр, универсальная унификация обращается к вопросам, типичным именно для решения уравнений как таковых.

Подобно тому как развитие традиционной теории решения уравнений стимулировалось ее многочисленными приложениями (в каждое время — своими: усложненным делением наследств во времена Вавилона и приложением в физике во времена, более близкие к нам), развитие теории унификации обусловлено ее многочисленными приложениями в информатике, искусственном интеллекте и, наконец, в области вычислительной дедукции — хотя и последней, но не менее важной.

Центральными в теории унификации являются понятия *множества наиболее общих унификаторов* $\mu U\Sigma$ (в традиционных терминах — множество базисных векторов, на которые натягивается пространство решений) и *иерархии проблем унификации*, основанной на $\mu U\Sigma$:

[1)] Siekmann J., Szabo P. Universal Unification and a classification of equational theories, Lecture Notes in Computer Science, vol. 138 (1982), 369—389.

[2)] University of Karlsruhe Institut für Informatik I, West Germany.

(i) теория T является *унитарной*, если $\mu U\Sigma$ всегда существует и имеет не более одного элемента;

(ii) теория T является *финитарной*, если $\mu U\Sigma$ всегда существует и конечно;

(iii) теория T является *инфинитарной*, если $\mu U\Sigma$ всегда существует и существует пара термов, таких что множество $\mu U\Sigma$ бесконечно для этой пары;

(iv) теория T является теорией *типа нуль* во всех других случаях

(точное определение этой иерархии дается ниже).

В этой работе мы характеризуем границу между (i) и (ii) для эквациональных теорий. Главные из нерешенных проблем касаются границы между (ii) и (iii), а также новых доказательств существования для $\mu U\Sigma$.

В первой части мы даем краткий обзор теории унификации в ее сегодняшнем состоянии. Вторая часть предлагает классификацию эквациональных теорий с точки зрения унификации, и третья часть содержит главный результат этой работы: характеристику унитарных теорий.

1. ВВЕДЕНИЕ В ТЕОРИЮ УНИФИКАЦИИ

Теория унификации занимается проблемами следующего сорта. Пусть f и g — функциональные символы, a и b — константы, x и y — переменные. Рассмотрим два *терма первого порядка*, построенные из этих символов, например:

$$t_1 = f(x,\ g(a,\ b)),$$
$$t_2 = f(g(y,\ b),\ x).$$

Первый вопрос, который здесь возникает, заключается в следующем: существуют ли термы, которые можно подставить вместо переменных x и y, чтобы из t_1 и t_2 получились два равных терма. В нашем примере такими двумя термами являются $g(a, b)$ и a. Мы будем писать

$$\sigma_1 = \{x \leftarrow g(a,\ b),\ \ y \leftarrow a\},$$

где σ_1 — искомая унифицирующая подстановка: σ_1 является унификатором t_1 и t_2, так как $\sigma_1 t_1 = \sigma_2 t_2$.

В дополнение к *проблеме разрешимости* существуют также проблемы нахождения *алгоритма унификации*, который порождает унификаторы для заданной пары t_1 и t_2.

Рассмотрим, какие изменения происходят с указанными выше проблемами, если мы предполагаем, что f коммутативна, т. е.

(C) $$f(x,\ y) = f(y,\ x),$$

Тогда σ_1, как и ранее, является унифицирующей подстановкой; более того, и подстановка $\sigma_2 = \{y \leftarrow a\}$ — также унификатор t_1 и t_2, поскольку

$$\sigma_2 t_1 = f(x,\ g(a,\ b)) = {}_C f(g(a,\ b),\ x) = \sigma_2 t_2.$$

Но σ_2 является *более общей* подстановкой, чем σ_1, так как σ_1 — частный случай σ_2, получаемый как *композиция* $\lambda \circ \sigma_2$, где $\lambda = \{x \leftarrow g(a, b)\}$; следовательно, алгоритму унификации достаточно вычислить только σ_2.

Существуют пары термов, которые имеют больше одного наиболее общего унификатора (т. е. никакой из унификаторов не является частным случаем другого унификатора) относительно коммутативности, но эти пары имеют самое большее *конечное число* унификаторов. Этот факт противопоставляет данную ситуацию первой (когда унифицировались свободные термы), где любая пара термов имела самое большее одну наиболее общую унифицирующую подстановку.

Совершенно иная ситуация возникает, когда мы предполагаем, что функция, обозначенная f, является ассоциативной, т. е.

$$\text{(A)} \qquad f(x,\ f(y,\ z)) = f(f(x,\ y),\ z).$$

В этом случае σ_1 — по-прежнему унифицирующая подстановка, однако

$$\sigma_3 = \{x \leftarrow f(g(a,\ b),\ g(a,\ b)),\ y \leftarrow a\}$$

также является унификатором:

$$\sigma_3 t_1 = f(f(g(a,\ b),\ g(a,\ b)),\ g(a,\ b)) = {}_A f(g(a,\ b),\ f(g(a,\ b),\ g(a,\ b))) = \sigma_3 t_2.$$

Но $\sigma_4 = \{x \leftarrow f(g(a,\ b),\ f(g(a,\ b),\ g(a,\ b))),\ y \leftarrow a\}$ — опять унифицирующая подстановка, и нетрудно видеть, что существует *бесконечно много* унификаторов, каждый из которых наиболее общий. Окончательно, если мы предполагаем, что обе аксиомы (A) и (C) справедливы для f, тогда ситуация снова изменяется, и для любой пары термов существует *не более конечного числа* наиболее общих унификаторов относительно (A) и (C).

Много специальных алгоритмов унификации для простых теорий уже было разработано ранее, что вызвано их важными приложениями в теории и практике машинной обработки информации. Сейчас мы дадим краткое изложение формализма, в котором выражаются наши проблемы.

1.1. Унификация с алгебраической точки зрения

Как обычно, пусть $\mathbb{N}$ — множество натуральных чисел. Множество «символов с арностью» — это отображение Ω: $M \to \mathbb{N}$, где M — некоторое множество. Для $f \in M$ Ωf является *арностью* f. *Область определения* Ω используется, чтобы обозначать определенные n-арные операции и иногда называется *сигнатурой*; $(f, n) \in \Omega$ сокращается до $f \in \Omega$.

Универсальная алгебра A — это пара (A, Ω), где A — *носитель* и $f \in \Omega$ обозначает отображение

$$f\colon A^n \to A, \text{ где } \Omega f = n$$

(если $a_1, \ldots, a_n \in A$, то мы пишем $f_A(a_1, \ldots, a_n)$ для *реализации* обозначенного отображения [1])). Отметим, что если $\Omega f = 0$, то f — выделенная константа алгебры A; $\mathrm{COD}(\Omega)$, *образ* Ω, есть *тип* алгебры A.

Если A и B — алгебры, φ: $A \to B$ является *гомоморфизмом*, если $\varphi f_A(a_1, \ldots, a_n) = f_B(\varphi a_1, \ldots, \varphi a_n)$; биективный гомоморфизм называется *изоморфизмом*, в символьном виде $\simeq$ [2]).

Для любого подмножества $A_0 \subseteq A$ $\varphi_0 = \varphi|_{A_0}$ есть *ограничение* φ на A_0. Отношение эквивалентности ρ является отношением *конгруэнтности* (или конгруэнцией. — *Перев.*) тогда и только тогда, когда из $a_1 \rho b_1, \ldots, a_n \rho b_n$ следует $f_A(a_1, \ldots, a_n) \rho f_A(b_1, \ldots, b_n)$. $A/\rho = (A/\rho, \Omega)$ есть *факторалгебра* универсальной алгебры A по конгруэнции ρ; $[a]_\rho$ есть класс конгруэнции ρ, порожденный $a \in A$.

Для класса алгебр $\boldsymbol{K}_0$ фиксированного типа алгебра $A = (A, \Omega)$ является *свободной* в $\boldsymbol{K}_0$ *над множеством* X (записывается в виде $A_{\boldsymbol{K}_0}(X)$) тогда и только тогда, когда выполнены следующие условия:

(i) $(A, \Omega) \in \boldsymbol{K}_0$,

(ii) $X \subseteq A$;

(iii) если $B \in \boldsymbol{K}_0$ и φ_0: $X \to B$ — произвольное отображение, то существует единственный гомоморфизм φ: $A \to B$, для которого $\varphi_0 = \varphi|_X$.

Если $\boldsymbol{K}$ — класс *всех* алгебр фиксированного типа, тогда $A_{\boldsymbol{K}}(X)$ (так как эта алгебра существует и единственна с точностью до изоморфизма) — *абсолютно свободная алгебра над* X. Элементы $A_{\boldsymbol{K}}(X)$ называются *термами*, а их конкретное представление W_Ω^X дается следующими правилами:

---

[1]) Другими словами, элемент множества A, полученный применением к набору из n элементов $a_1, \ldots, a_n$ множества A операции f, записывается в виде $f_A(a_1, \ldots, a_n)$. — *Прим. перев.*

[2]) Другими словами, если существует изоморфизм между A и B, то говорят, что A и B *изоморфны* и пишут $A \simeq B$. — *Прим. перев.*

(i) если $x \in X$, то $x \in W_\Omega^X$;

(ii) если $t_1, t_2, \ldots, t_n$ — термы и $\Omega f = n$, $n \geqslant 0$, тогда $f(t_1, \ldots, t_n) \in W_\Omega^X$.

Мы предполагаем, что Ω состоит из двух непересекающихся множеств Φ и Γ, таких что

$$f \in \Phi \Leftrightarrow \Omega f \geqslant 1 \quad \text{и} \quad f \in \Gamma \Leftrightarrow \Omega f = 0.$$

Φ называется множеством функциональных символов, Γ — множеством констант и X — множеством переменных.

Мы определяем операции

$$\hat{f}\colon (W_\Omega^X)^n \to W_\Omega^X \text{ для } n = \Omega f$$

соотношением $\hat{f}(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$. Пусть $\hat{\Omega}$ будет множеством таких (строящих термы) операций. Пусть $\varnothing$ обозначает пустое множество.

$F_\Omega^X = (W_\Omega^X, \hat{\Omega})$ изоморфна $A_K(X)$ и поэтому называется абсолютно свободной *алгеброй термов* над X. $F_\Omega^\varnothing$ — *инициальная* алгебра термов (или эрбрановский универсум). Для $F_\Omega^\varnothing$ будем использовать запись F_{Ω_0}. Наш интерес к F_{Ω_0} объясняется тем фактом, что для любой алгебры $A = (A, \Omega)$ существует единственный гомоморфизм

$$h_A\colon F_{\Omega_0} \to A.$$

Но тогда вместо исследования A мы можем ограничить наше внимание факторалгеброй алгебры F_{Ω_0} по конгруэнции, индуцированной h_A [1]).

Чтобы в инициальной алгебре иметь в своем распоряжении переменные, мы определяем $\Omega_X = \Omega \cup X$, т. е. рассматриваем переменные как специальные константы. Так как $F_\Omega^X \simeq F_{\Omega_X}^\varnothing$, мы просто пишем F_Ω, если $X \neq \varnothing$ и $X \subset \Omega$, и пишем F_{Ω_0}, если $X = \varnothing$. Поскольку термы являются объектами в F_Ω, мы будем писать $t \in F_\Omega$ вместо $t \in W_\Omega^X$.

Равенство — это пара термов s, $t \in F_\Omega$, в символьном виде $s = t$. Равенство $s = t$ является *истинным* в алгебре A (того же типа), в символьном виде $A \models s = t$, тогда и только тогда, когда для любого гомоморфизма $\varphi\colon F_\Omega \to A$

$$\varphi s = st \quad \text{в} \quad A.$$

Пусть $\sigma\colon X \to F_\Omega$ будет отображением, которое почти всюду равно тождественному отображению. *Подстановка* $\sigma\colon F_\Omega \to F_\Omega$

[1]) Так оно и было бы, если гомоморфизм h_A был бы эпиморфизмом, т. е. был бы гомоморфизмом на всю алгебру A. Этого, конечно, нет, когда мощность минимальной системы образующих алгебры A больше мощности Γ. — *Прим. перев.*

является гомоморфным расширением $\bar{\sigma}$ и представляется конечным множеством пар

$$\sigma = \{x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\}.$$

Σ — это множество подстановок на F_Ω. Тождественное отображение на F_Ω, т. е. *пустая подстановка*, обозначается символом ε. Если t — терм и σ — подстановка, определим отображение V: $F_\Omega \to 2^X$ формулой

$V(t) =$ множество переменных в t и $V(t_1, \ldots, t_n) = \bigcup\limits_{i \leqslant n} V(t_i)$.

Пусть $|t| \in \mathbb{N}$ обозначает длину t (т. е. число символов в t),

$\mathrm{DOM}(\sigma) = \{x \in X : \sigma x \neq x\}$,

$\mathrm{COD}(\sigma) = \{\sigma x : x \in \mathrm{DOM}(\sigma)\}$,

$X\,\mathrm{COD}(\sigma) = V(\mathrm{COD}(\sigma))$.

$\Sigma_0 \subset \Sigma$ обозначает множество *фундаментальных* подстановок, т. е. $\sigma \in \Sigma_0$ тогда и только тогда, когда $\mathrm{COD}(\sigma) \subset F_{\Omega_0}$.

Равенство $s = t$ является *унифицируемым* (*разрешимым*) в A тогда и только тогда, когда существует гомоморфизм ξ: $F_\Omega \to A$, такой что $\xi s = \xi t$ истинно в A. Множество равенств T индуцирует конгруэнцию $=_T$ в F_Ω, и $F_\Omega/=_T$ есть факторалгебра по конгруэнции $=_T$.

Проблема унификации для T, обозначаемая $\langle s = t\rangle_T$, задается равенством $s = t$, $s, t \in F_\Omega$. Проблема состоит в том, чтобы решить, является ли $s = t$ унифицируемым в $F_\Omega/=_T$ или нет.

Мы обозначаем составные части инициальной алгебры $F_\Omega/=_T$ как (F_Ω, T)-*алгебру*.

1.2. Унификация с логической точки зрения

1.2.1. Эквациональная логика

Правильно построенными формулами нашей логики являются равенства, определяемые как пары в $W_\Omega^X \times W_\Omega^X$ и обозначаемые через $s = T$.

Подстановка σ — это конечное множество пар в $X \times W_\Omega^X$ (т. е. классический подход слегка запутывает дело, идентифицируя представление с отображением, которое представляется). Конкретизация σt терма t подстановкой $\sigma = \{x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n\}$ получается путем одновременной замены каждой x_i в t термом t_i.

Пусть T — множество равенств. Равенство $p = q$ *выводимо из* T, $T \vdash p = q$, если $p = q \in T$ или $p = q$ получается из T конечной последовательностью следующих операций:

(i) $t = t$ (аксиома);
(ii) если $s = t$, то $t = s$;
(iii) если $r = s$ и $s = t$, то $r = t$;
(iv) если $s_i = t_i$, $1 \leqslant i \leqslant n$, то $f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)$, $n = \Omega f$.
(v) если $s = t$, то $\sigma s = \sigma t$, где $\sigma \in \Sigma$.

Для множества равенств T, $T \models s = t$ тогда и только тогда, когда $s = t$ истинно во всех *моделях* T.

Теорема (Биркгоф): $T \models s = t$ *тогда и только тогда, когда* $T \vdash s = t$.

Для сокращения $T \models s = t$ (а значит, и $T \vdash s = t$) мы будем использовать запись $s = t$. Равенство $s = t$ *T-унифицируемо* тогда и только тогда, когда существует подстановка σ, такая что $\sigma s = \sigma t$.

Хотя это является традиционной точкой зрения на унификацию, кажущаяся простота такого подхода обманчива: мы не определяем, что мы понимаем под «моделью». Чтобы дать определение модели, нам потребовалось бы понятие интерпретации наших правильно построенных формул как «гомоморфизма» из W_Ω^X в алгебры определенного типа, что вернуло бы нас назад к разд. 1.1. Так как ни $\models$, ни $\vdash$ не являются особо удобными для вычислительной обработки $=_T$, ниже предлагается третий метод.

1.2.2. Вычислительная логика

Для упрощения обозначений мы предполагаем, что в нашем распоряжении имеется бокс символов, GENSYM, из которого мы можем выбирать сколь угодно много «новых» символов. Более формально: для F_Ω пусть $\Omega = \Phi \cup \Gamma \cup X$, где $X = X_0 \cup \text{GENSYM}$, при этом $\Omega x = 0$, $x \in X$.

Мы примем такое вычислительное правило, что всякий раз, когда мы обращаемся к GENSYM через $v \in \text{GENSYM}$ последовательно «обновляются»: $\text{GENSYM}' = \text{GENSYM} - \{v\}$ и $X_0' = X_0 \cup \{v\}$ и $\Omega' = \Phi \cup \Gamma \cup X'$, где $X' = X_0' \cup \text{GENSYM}'$. Так как $F_\Omega \simeq F_{\Omega'}$, мы будем иногда опускать штрих и просто писать F_Ω [1]).

Переименовывающая подстановка $\rho \in \Sigma_X \subset \Sigma$ определяется следующими условиями:

---

[1]) Если X и Y — некоторые множества, то алгебры Ω-слов над X и Y (в обозначениях настоящей работы — абсолютно свободные алгебры термов над X и Y) изоморфны, $F_\Omega^X \simeq F_\Omega^Y$, тогда и только тогда, когда X и Y равномощны. — *Прим. перев.*

(i) $\mathrm{COD}(\rho) \subset X$;
(ii) $\forall x, y \in X$: если $x \neq y$, то $\rho x \neq \rho y$.

Для $s, t \in F_\Omega$: $s \sim_\rho t$, если $\exists \rho \in \Sigma_X$, такое что $\rho s = \rho t$. Если $\rho s = t$, тогда t называется *X-вариантом* s; если в дополнение к этому $\mathrm{COD}(\rho) \subset \mathrm{GENSYM}$, тогда t называется *новым* X-вариантом s.

Чтобы формализовать доступ к подтерму терма, поступаем следующим образом. Пусть $\mathbb{N}^*$ — множество последовательностей положительных целых чисел, Λ — пустая последовательность, $\cdot$ — операция конкатенации на последовательностях. Члены $\mathbb{N}^*$ называются *позициями* и обозначаются через $\pi \in \mathbb{N}^*$. Они используются следующим образом: зададим для любого $t \in F_\Omega$ множество $\Pi(t) \subset \mathbb{N}^*$ *позиций в* t следующим образом:

(i) если $\Omega t = 0$, то $\Pi(t) = \{\Lambda\}$;
(ii) если $t = f(t_1, \ldots, t_n)$, то $\Pi(t) = \{\Lambda\} \cup \{i \cdot \pi\colon\ 1 \leqslant i \leqslant n, \pi \in \Pi(t_i)\}$.

Например: $\Pi(f(g(a, y), b)) = \{\Lambda, 1, 2, 1 \cdot 1, 1 \cdot 2\}$. Подтерм терма $t = f(t_1, \ldots, t_n)$ на π, $t|\pi$, определяется так:

(i) $t|\pi = t$ для $\pi = \Lambda$ или $\pi \notin \Pi(t)$;
(ii) $t|i \cdot \pi' = t_i|\pi'$ для $\pi = i \cdot \pi'$.

Например: $f(g(a, y), b)|1 \cdot 2 = y$.

Результат применения *замены* ρ к терму t, $\hat{\rho}t$, где $\hat{\rho} = [\pi \leftarrow s]$ — *замена подтерма* на позиции π термом s, определяется так:

(i) $\hat{\rho}t = s$, если $\pi = \Lambda$;
(ii) $\hat{\rho}t = f(t_1, \ldots, \hat{\delta}t_i, \ldots, t_n)$, если $t = f(t_1, \ldots, t_n)$, $\pi = i \cdot \pi'$ и $\hat{\delta} = [\pi' \leftarrow s]$.

Мы обозначаем замены через $\hat{\sigma}$, $\hat{\rho}$, $\hat{\delta}$ и т. д., а подстановки через σ, ρ, δ и т. д.

Отношение $\rightarrow \subseteq F_\Omega \times F_\Omega$ является *нетеровым* (оканчивающимся), если не существует бесконечных последовательностей вида $s_1 \rightarrow s_2 \rightarrow s_3 \rightarrow \ldots$. Как обычно, $\xrightarrow{+}$ — транзитивное, а $\xrightarrow{*}$ — рефлексивное и транзитивное замыкание $\rightarrow$. Отношение $\rightarrow$ является *конфлюентным*, если для любых r, s, $t \in F_\Omega$, таких что $r \xrightarrow{*} s$ и $r \xrightarrow{*} t$, существует элемент $u \in F_\Omega$, такой что $s \xrightarrow{*} u$, $t \xrightarrow{*} u$. Конфлюентное нетерово отношение называется *каноническим*.

Сейчас мы определим два важных отношения $\rightarrow_R$ и $\rightarrowtail_R$ на множестве $F_\Omega \times F_\Omega$.

Переписывающая система — это любое множество $R = \{l_1 \Rightarrow r_1, \ldots, l_n \Rightarrow r_n\}$ пар $l_i, r_i \in F_\Omega$, таких что $V(r_i) \subseteq V(l_i)$, $1 \leqslant i \leqslant n$.

Мы говорим о двух термах s и t, что s *переписывается* в t, $s \rightarrow_R t$, если существуют $\pi \in \Pi(s)$, $\sigma \in \Sigma$ и $(l_i \Rightarrow r_i) \in R$, такие

что $s|\pi = \sigma\tilde{l}_i$ и $t = \hat{\sigma}s$, где $\hat{\sigma} = [\pi \leftarrow \sigma\tilde{r}_i]$, а l_i, $\tilde{r}_i$ — новые X-варианты l_i, r_i. Изредка мы сохраняем след этой информации, записывая $s \xrightarrow[[\pi,\ i,\ \sigma]]{} t$, $s \xrightarrow[[\pi,\ i]]{} t$, $s \xrightarrow[\pi]{} t$ и т. д.

Мы говорим о двух термах s и t, что s *парамодулируется* в t, $s \rightarrowtail_R t$, если существуют $\pi \in \Pi(s)$, $(l_i \Rightarrow r_i) \in R$, $\sigma \in \Sigma$, такие что $\sigma(s|\pi) = \sigma\tilde{l}_i$ и σ — наиболее общий унификатор (см. 1.3 ниже), $\tilde{l}_i$ — новый X-вариант l_i и $\sigma s \xrightarrow[[\pi,\ i]]{} t$.

Например, для $R = \{g(x, 0) \rightarrow 0\}$ мы имеем

$$s = f(g(a, y), y) \rightarrowtail_R f(0, 0) = t,$$

где $\pi = 1$ и $\sigma = \{x \leftarrow a, y \leftarrow 0\}$. Однако отметим, что $s \not\rightarrowtail_R t$, так как мы не имеем права делать подстановки в s.

Наши обозначения и определения переписывающих систем для термов согласуются с [HT 80]; важность переписывающих систем (демодуляции) для доказательства теорем [1]) впервые была отмечена в [WR 67].

Предположим, что для эквациональной теории T существует переписывающая система R_T, такая что для любых s, $t \in F_\Omega$ $s = t$ в том и только в том случае, когда $\exists p \in F_\Omega$, так что $s \xrightarrow{*}_{R_T} p$ и $t \xrightarrow{*}_{R_T} p$. В этом случае мы говорим, что T *вкладывается в* R_T и пишем $T \infty R_T$.

Существует методика получения для эквациональной теории T переписывающей системы R_T, такой что $T \infty R_T$; более того, для многих теорий, имеющих практическое значение, можно получить переписывающие системы R_T, для которых отношение $\rightarrow_{R_T}$ является каноническим ([KB 70], [HT 80], [PS 81], [HU 80.1]). Канонические отношения $\rightarrow_R$ образуют важный базис для *вычислений в эквациональной логике*, так как они определяют единственную *нормальную форму* $\|t\|$ для каждого $t \in F_\Omega$, задаваемую следующим условием: $t \xrightarrow{*} \|t\|$ и $\nexists s \in F_\Omega$, такой что $\|t\| \rightarrow s$. Отсюда

(i) $s =_T t$ тогда и только тогда, когда $\|s\| = \|t\|$.

В случае когда система R_T нетерова (т. е. R_T определяет нетерово отношение $\rightarrow_{R_T}$, мы также говорим, что она является *системой редукций*.

В качестве следствия теоремы Биркгофа имеем

(ii) $\exists\sigma \in \Sigma$: $\sigma s = \sigma t$ тогда и только тогда, когда $\exists \bar{p}$, $p \in F_\Omega$ и $\exists\delta \in \Sigma$: $h(s, t) \xrightarrow{*}_{R_T} h(\bar{p}, p)$, где $\delta p =_\varnothing \delta\bar{p}$.

Здесь h — «дополнительный» функциональный символ, не входящий в Ω; таким образом, h, будучи «новым», не «затрагивается» R_T.

[1]) Имеется в виду доказательство теорем на ЭВМ, или автоматическое доказательство теорем.— *Прим. перев.*

В [FA 79], [LB 79], [HU 80], [SS 81.5], [H 82] эта теорема взята за основу алгоритма универсальной унификации: точно так же, как универсальная машина Тьюринга получает на свой вход конкретный аргумент и описание конкретной машины Тьюринга, на вход *универсального алгоритма унификации* поступает пара, состоящая из конкретной проблемы унификции и конкретной эквациональной теории T.

1.3. Универсальная унификация

Эквациональная теория T *разрешима* в том и только в том случае, если равенство $s =_T t$ разрешимо для любых $s, t \in F_\Omega$. Пусть $\mathsf{T}_=$ обозначает семейство разрешимых конечно определенных эквациональных теорий [1]).

Проблема T-унификации $\langle s = t \rangle_T$ состоит из пары термов $s, t \in F_\Omega$ и теории $T \in \mathsf{T}_=$.

Подстановка $\sigma \in \Sigma$ является *T-унификатором* для $\langle s = t \rangle_T$ тогда и только тогда, когда $\sigma s =_T \sigma t$. *Множество унификаторов* (для s и t *относительно* T), $U\Sigma_T(s, t)$ — это подмножество из Σ, которое унифицирует $\langle s = t \rangle_T$. Легко видеть, что $U\Sigma_T(s, t)$ рекурсивно перечислимо (р. п.) для любых s и t. Действительно, так как F_Ω рекурсивно перечислимо, то и Σ р. п.; тогда для каждой подстановки $\delta \in \Sigma$ проверяем, верно ли, что $\delta s =_T \delta t$ (эта проблема разрешима, так как $T = \mathsf{T}_=$); если верно, то $\delta \in U\Sigma_T(s, t)$, иначе $\delta \notin U\Sigma_T(s, t)$.

Мы будем опускать индекс T и (s, t), если они понятны из контекста. Композиция подстановок определяется как обычная композиция отображений: $(\sigma \circ \tau)t = \sigma(\tau t)$. Если $W \subseteq X$, то T-равенство распространяется на подстановки.

Подстановки σ и τ являются *T-равными на* W, $\sigma =_T \tau[W]$, тогда и только тогда, когда $\forall x \in W \quad \sigma x =_T \tau x$.

Мы говорим, что σ — это *пример* τ и τ — *более общая* подстановка, чем σ (обозначается $\sigma \leqslant_T \tau[W]$), тогда и только тогда, когда

$$\exists \lambda \in \Sigma : \sigma = \lambda \circ \tau[W] \text{ для некоторого } W \subset X.$$

Если $\sigma \leqslant_T \tau[W]$ и $\tau \leqslant_T \sigma[W]$, то σ и τ являются *T-эквивалентными на* W, в символьном виде $\sigma \approx_T \tau[W]$.

Для $\Sigma_1, \Sigma_2 \subseteq \Sigma$ мы определяем $\Sigma_1 \circ \Sigma_2 = \{\sigma_1 \circ \sigma_2 : \sigma_1 \in \Sigma_1, \sigma_2 \in \Sigma_2\}$. $\Sigma_1 \subseteq_T \Sigma_2[W]$ тогда и только тогда, когда $\forall \sigma_1 \in \Sigma_1$ $\exists \sigma_2 \in \Sigma_2$, такие, что $\sigma_1 =_T \sigma_2[W]$. $\Sigma_1 =_T \Sigma_2[W]$ тогда и только тогда, когда $\Sigma_1 \subseteq_T \Sigma_2[W]$ и $\Sigma_2 \subseteq_T \Sigma_1[W]$.

---

[1]) В данной работе не сформулировано понятие эквациональной теории, но эквациональная теория понимается как множество равенств. Таким образом, $\mathsf{T}_=$ — это семейство конечных множеств равенств, в котором каждое множество индуцирует разрешимую конгруэнцию на F_Ω. — *Прим. перев.*

Универсальная унификация связана с двумя фундаментальными проблемами.

Первая проблема (*проблема разрешимости*):

Для заданной теории $T \in \mathbf{T}_{=}$, *разрешим ли вопрос об унифицируемости относительно* T *пары* s *и* t, *где* s *и* t *могут быть любыми* [1])?

То есть мы интересуемся такими классами теорий, что проблема «s и t унифицируемы относительно T» разрешима для любой T из этого класса.

Унификатор σ для $\langle s = t\rangle_T$ называется *наиболее общим унификатором* (н. о. у.), если для любого унификатора $\delta \in U\Sigma_I(s, t)$ $\delta \leqslant \delta[W]$, где $W = V(s, t)$. Так как в общем случае не существует единственного н. о. у. для $\langle s = t\rangle_T$, мы определяем *множество наиболее общих унификаторов*, $\mu U\Sigma_T(s, t)$, следующими условиями:

(i) $\mu U\Sigma \subseteq U\Sigma$ (корректность);
(ii) для любого δ, такого что $\delta s = \delta t$ существуют $\sigma \in \mu U\Sigma$ и $\lambda \in \Sigma$, такие что $\delta =_T \lambda \circ \sigma[W]$ (полнота);
(iii) если $\sigma, \tau \in \mu U\Sigma$, то $\sigma \not\leqslant_T \tau[W]$ (минимальность).

Из условия (ii) следует, в частности, что $U\Sigma =_T \Sigma \circ U\Sigma[W]$, т. е. $U\Sigma$ — *левый идеал* в полугруппе $(\Sigma, \circ)$ и $U\Sigma$ *порождается множеством* $\mu U\Sigma$.

Для практических приложений эти условия являются иногда слишком общими и существуют дополнительные технические требования к $\mathrm{DOM}(\sigma)$, $\mathrm{COD}(\sigma)$ и $\mathrm{XCOD}(\sigma)$ для $\sigma \in \mu U\Sigma$. Мы сформулируем эти требования, когда возникнет необходимость.

Множества $\mu U\Sigma_T$ существуют не всегда; но в случае, когда такое множество существует, оно единственно с точностью до эквивалентности $\approx_T$ для унифицируемости относительно T (см. [HT76], лемма 2.16). На этом основании достаточно стенерировать одно множество $\mu U\Sigma_T$. В дальнейшем мы всегда принимаем $\mu U\Sigma_I$ за представителя класса эквивалентности $[\mu U\Sigma_T]_{\approx}$.

Вторая проблема (*проблема существования*):

Если задана эквациональная теория $T \in \mathbf{T}_{=}$, *верно ли, что* $\mu U\Sigma_T(s, t)$ *существует для любых* $s, t \in F_\Omega$?

Третья проблема (*проблема перечисления*):

---

[1]) Другими словами, существует ли эффективная процедура для определения по данным s и t, унифицируемы ли они относительно T или нет. — *Прим. перев.*

Для заданной эквациональной теории $T \in \mathsf{T}_=$ *определить, является ли множество* $\mu U\Sigma_T(s, t)$ *рекурсивно перечислимым для любых* $s, t \in F_\Omega$.

Таким образом, нас интересуют алгоритмы, которые порождают все наиболее общие унификаторы для заданной проблемы $\langle s = t\rangle_T$. В табл. 1 сведены главные результаты, которые были получены для специальных теорий, состоящих из комбинаций следующих равенств:

Таблица 1

Теория T	Тип T	Унификация разрешима?	$\mu U\Sigma_T$ рекурсивно?	$\mathscr{A}_T$	Ссылки
$\varnothing$	1	Да	Да	Да	[HE30], [RO65], [RO71], [KB70], [G67], [PR60], [BA73], [HT76], [MM79], [PW78]
A	∞	Да	Да	Да	[HM67], [PL72], [SI75], [LS75], [MA77]
C	ω	Да	Да	Да	[SI82]
I	ω	Да	Да	Да	[RS78], [S82.2], [SZ82]
A+C	ω	Да	Да	Да	[ST75], [LS76], [HU79]
A+I	?	Да	Да	Нет	[SS82.3], [SZ82], [SS82.1]
C+I	ω	Да	Да	Да	[RS78]
A+C+I	ω	Да	Да	Да	[LS76]
D	ω	?	Да	Да	[SZ82]
D+A	∞	Нет	Да	Да	[S78], [SZ82]
D+C	∞	?	Да	Да	[SZ82]
D+A+C	∞	Нет	Да	Да	[SZ82]
D+A+I	?	Да	Да	Нет	[SZ82]
H,E	1	Да	Да	Да	[VO78]
H+A	∞	Да	Да	Да	[VO78]
H+A+C	ω	Да	Да	Да	[VO78]
E+A+C	∞	?	?	Нет	[VO78]
QG	ω	Да	Да	Да	[HU80]
AG	ω	Да	Да	Да	[LA79]
H10	?	?	Нет	?	[MA70], [DA73]
FPA	ω	Да	Да	Да	[LA80]
Sot T=$\varnothing$	?	Нет	—	—	[GO81]
Hot T=$\varnothing$	0	Нет	—	—	[HT73], [HT75], [HT76], [BA78]

A (ассоциативность) $f(f(x, y), z) = f(x, f(y, z))$
C (коммутативность) $f(x, y) = f(y, x)$
D (дистрибутивность) $f(x, g(y, z)) = g(f(x, y), f(x, z))$
$f(g(x, y), z)) = g(f(x, z), f(y, z))$
H, E (гомоморфизм, эндоморфизм) $\varphi(x \circ y) = \varphi(x) \circ \varphi(y)$
I (идемпотентность) $f(x, x) = x$

В таблице применены следующие сокращения:

FPA: Конечно-представимые алгебры
QG: Квазигруппы
AG: Абелевы группы
H10: 10-я проблема Гильберта
Sot: Термы второго порядка
Hot: Термы более высокого порядка (т. е. $\geqslant$ 3-го порядка).

Колонка под $\mathscr{A}_T$ указывает, был ли представлен в литературе алгоритм конформного типа или нет. Понятия «тип теории» и «алгоритм конформного типа» определяются ниже.

За исключением работ по десятой проблеме Гильберта, мы не включили в таблицу классические труды по решению уравнений в «реальных» структурах, таких, как кольца и поля, так как эти результаты хорошо известны.

Отношение универсальной унификации к этим классическим результатам похоже на отношение универсальной алгебры к классической алгебре.

Центральное понятие $\mu U\Sigma$ индуцирует иерархию классов эквациональных теорий (теорий, *релевантных унификации*):

(i) Теория T является *унитарной*, если $\forall s, t\ \mu U\Sigma_T(s, t)$ существует и содержит самое большее один элемент. Пусть $\mathbf{U}_1$ — класс таких теорий (*типа один*).

(ii) Теория T является *финитарной*, если она не унитарна и $\forall s, t\ \mu U\Sigma_T(s, t)$ существует и конечно. Пусть $\mathbf{U}_\omega$ — класс таких теорий (*типа* ω).

(iii) Теория T является *инфинитарной*, если $\forall s, t\ \mu U\Sigma_T(s, t)$ существует и существует $\langle p = q \rangle_T$, такая что $\mu U\Sigma_T(p, q)$ бесконечно. Пусть $\mathbf{U}_\infty$ — класс таких теорий (*типа* ∞).

(iv) Теория T является теорией *типа нуль*, если она не попадает ни в один из определенных выше классов. Пусть $\mathbf{U}_0$ — класс таких теорий.

В приведенной выше таблице содержится несколько примеров унитарных, финитарных и инфинитарных теорий. Пример теории типа нуль, принадлежащий Ф. Фейгу [FA81]:

$$T = \{f(1, x) = x;\ \ g(f(x, y)) = g(y)\}.$$

Здесь $\mu U\Sigma_T$ не существует для проблемы $\langle g(x) = g(a)\rangle_T$. Определим частичный порядок на термах следующим образом: $s \leqslant_T t$ в том и только в том случае, если $\exists\delta \in \Sigma$, удовлетворяющая условию $s =_T \delta T$. *Проблема выравнивания* $\langle s \geqslant t\rangle_T$ состоит из пары термов и теории $T \in \mathsf{T}_=$. Подстановка $\nu \in \Sigma$ является *T-выравнивателем* (или односторонним унификатором), если $\nu s =_T t$. Если s и t выравниваемы, мы будем писать $s \leqslant_T t$, соответственно $s \geqslant_T t$.

Понятие $\mu U\Sigma_T$ индуцирует иерархию (*теорий, релевантных выравниванию*), подобную иерархии, основанной на $\mu U\Sigma_T$: теория T является *унитарной по выравниванию*, если $\mu M\Sigma_T$ всегда существует и содержит максимум один элемент. Класс таких теорий обозначим через M_1. Аналогичным образом мы определяем классы M_ω, M_∞ и класс M_0. Отметим, что из того же самого примера, который приведен выше, следует $\mathsf{M}_0 \neq \varnothing$.

Алгоритм унификации $\mathscr{A}_T$ (*алгоритм выравнивания* $\mathscr{M}_T$) — для теории T — это такой алгоритм, который получает на вход два терма s и t и порождает множество $\Psi_T \subseteq U\Sigma_T$ $(\subseteq M\Sigma_T)$ для $\langle s = t\rangle_T$ (для $\langle s \geqslant t\rangle_T$). *Минимальный алгоритм* $\mu\mathscr{A}_T$ ($\mu\mathscr{M}_T$) — это алгоритм, который порождает $\mu U\Sigma_T$ ($\mu M\Sigma_T$).

Для многих практических приложений это требование является недостаточно строгим, так как из него не следует, что алгоритм останавливается в случае, когда $T \in \mathsf{U}_1 \cup \mathsf{U}_\omega$. С другой стороны, для $T \in \mathsf{U}_\omega$ оно иногда излишне жесткое, так как некоторый алгоритм, который порождает конечное надмножество множества $\mu U\Sigma_T$ может оказаться намного более эффективным, чем алгоритм $\mu\mathscr{A}_T$, и, следовательно, более предпочтительным. Исходя из этого мы предлагаем следующее определение.

Алгоритм $\mathscr{A}_T$ называется алгоритмом, *конформным* относительно *типа*, тогда и только тогда, когда:

(i) $\mathscr{A}_T$ порождает множество Ψ_T, такое что $U\Sigma_T \supseteq \Psi_T \supseteq \supseteq \mu U\Sigma_T$ для некоторого $\mu U\Sigma_T$;
(ii) $\mathscr{A}_T$ останавливается и Ψ_T конечно, если $T \in \mathsf{U}_1 \cup \mathsf{U}_\omega$ и
(iii) если $T \in \mathsf{U}_\infty$, то $\Psi_T = \mu U\Sigma_T$ для некоторого $\mu U\Sigma_T$.

Аналогично алгоритм $\mathscr{M}_T$ называется алгоритмом, конформным относительно типа, в том и только в том случае, если выполняются требования (i)—(iii) с заменой U на M.

Опыт показывает, что для разных теорий алгоритмы унификации, как правило, основываются на совершенно разных методах. Однако как из теоретических соображений, так и эвристических, было бы интересно иметь универсальный алгоритм унификации для целого класса теорий, пусть даже неэффективный. *Универсальный алгоритм унификации* (*универсальный алгоритм выравнивания*) *для класса теорий* $\mathsf{T} \subset \mathsf{T}_=$ — это та-

кой алгоритм, который в качестве входных данных получает пару термов (s, t) и теорию $T \in \mathsf{T}$ и порождает полное множество унификаторов (выравнивателей) для $\langle s = t\rangle_T$ (для $\langle s \geqslant t\rangle_T$).

Так как для любой теории $T \in \mathsf{T}$ множество $U\Sigma_T$ рекурсивно перечислимо (из тривиальных соображений), то появляется важное требование, чтобы универсальный алгоритм унификации был минимальным или по крайней мере конформным относительно типа см. [FA79], [LB79], [SS81.5], [H82], [SZ82].

2. КЛАССИФИКАЦИЯ ЭКВАЦИОНАЛЬНЫХ ТЕОРИЙ

В этом разделе определяются некоторые важные подклассы эквациональных теорий, которые, как оказалось, представляют практический интерес, а также являются полезными в качестве «стандартных блоков» для построения других эквациональных классов. Вначале мы дадим определения, а затем сформулируем несколько теорем, чтобы продемонстрировать дескриптивное значение этих теорий.

Как и в разд. 1.3, пусть $\mathsf{T}_=$ — класс таких эквациональных теорий, которые являются конечно определенными и имеют разрешимую проблему слов. В настоящее время самый важный подкласс — это $\mathsf{T}_\Rightarrow := \{T \in \mathsf{T}_=$: существует переписывающая система R, такая что $T \sim R\}$.

Теория T является *регулярной* тогда и только тогда, когда для любого равенства $l = r \in T$ $V(l) = V(r)$; мы будем писать T^*, если T — класс регулярных теорий. В качестве непосредственного результата получаем $\mathsf{T}^*_= \subset \mathsf{T}_\Rightarrow$.

Важное требование по отношению к теории унификации состоит в том, чтобы проблема выравнивания была разрешима для T; пусть $\mathsf{T}_\leqslant$ обозначает класс таких теорий. Класс $\mathsf{A} := \mathsf{T}_\Rightarrow \cap \mathsf{T}_\leqslant$ называется классом *допустимых* теорий. Определяя $\mathsf{T}_\downarrow \subset \mathsf{T}_\Rightarrow$ как подкласс с *конфлюентной* переписывающей системой, и $\mathsf{R} \subset \mathsf{T}_\Rightarrow$ — как подкласс с *нетеровой* переписывающей системой, построим класс $\mathsf{R}_\downarrow = \mathsf{RT}_\downarrow$ (на протяжении всего этого раздела мы используем соглашение, что запись рядом служит сокращенным обозначением для пересечения классов), который назовем *классом канонических теорий в обобщенном смысле* (т. е. любая канонизация допустима). Определяя $\mathsf{C} \subset \mathsf{R}_\downarrow$ как класс, имеющий некоторую (стандартную) канонизацию, и полагая $\mathsf{M}_{\omega 1} = \mathsf{M}_\omega \cup \mathsf{M}_1$, построим классы $\mathsf{ACM}_{\omega 1}$ и $\mathsf{AC^*M}_{\omega 1}$, которые оказываются важными классами для универсального алгоритма унификации: в [SS81], [H82] показано, что мно-

жество $\mu U\Sigma_T$ является рекурсивно перечислимым для некоторого подкласса класса $\mathbf{AC^*M}_{\omega 1}$. Обозначая этот подкласс через $\mathbf{AC^*M}_{\downarrow}$, приходим к формулировке первой теоремы.

Теорема 2.1. $\mathbf{U}_0\mathbf{AC^*M}_{\downarrow} = \varnothing$, *т. е.* $\mu U\Sigma_T$ *существует для любой* $T \in \mathbf{AC^*M}_{\downarrow}$. Пример Ф. Фейга [FA81] показывает, что $\mathbf{U}_0\mathbf{AC^*M}_{\omega 1} \neq \varnothing$. Класс *Ω-свободных теорий* $\mathbf{F}_\Omega$ оказался интересным из-за его дескриптивных достоинств:

$$\mathbf{F}_\Omega = \{T \in \mathbf{T}_=\colon \textit{если } f(t_1, \ldots, t_n) = f(s_1, \ldots, s_n), \text{ то } t_i =_T s_i, \quad 1 \leqslant i \leqslant n\}.$$

Следующие четыре леммы характеризуют $\mathbf{F}_\Omega$ по отношению к базисной иерархии.

Лемма 2.1. $\mathbf{U}_1\mathbf{F}_\Omega \neq \varnothing$ *и* $\mathbf{F}_\Omega \neq \mathbf{U}_1$, *т. е. существует Ω-свободная унитарная теория, однако* $\mathbf{F}_\Omega$ *является классом, отличным от* $\mathbf{U}_1$.

Лемма 2.2. $\mathbf{U}_\omega\mathbf{F}_\Omega \neq \varnothing$ *и* $\mathbf{F}_\Omega \neq \mathbf{U}_\omega$, *т. е. существует Ω-свободная финитарная теория, но* $\mathbf{F}_\Omega$ *не совпадает с* $\mathbf{U}_\omega$.

Лемма 2.3. $\mathbf{U}_\infty\mathbf{F}_\Omega \neq \varnothing$ *и* $\mathbf{F}_\Omega \neq \mathbf{U}_\infty$, *т. е. существует Ω-свободная инфинитарная теория, но* $\mathbf{F}_\Omega$ *отличен от* $\mathbf{U}_\infty$.

Однако остается открытой

Проблема. $\mathbf{U}_0\mathbf{F}_\Omega = \varnothing$?

(То есть для всякой ли Ω-свободной теории существует $\mu U\Sigma_T$?)

Другими словами, F_Ω является в некотором смысле «диагональю» в базисной иерархии эквациональных классов.

В то же время мы имеем такой удивительный результат:

Теорема 2.2. $\mathbf{M}_1 = \mathbf{F}_\Omega$, т. е. $\mathbf{F}_\Omega$ *составляет в точности класс теорий, унитарных относительно выравнивания.*

Продолжая описание строительных блоков теории унификации, назовем класс $\mathbf{U} = \mathbf{U}_1 \cup \mathbf{U}_\omega \cup \mathbf{U}_\infty$ классом теорий, *релевантных унификации*, и определим класс нормальных теорий $\mathbf{N} = \mathbf{AU}$. Тогда из теоремы 2.1 получаем $\mathbf{AC^*M}_{\downarrow} = \mathbf{NC^*M}_{\downarrow}$.

Сейчас мы переходим к определению понятия *локального подкласса* $\lambda\mathbf{T}$ класса T, играющего важную роль в нашей теории.

Для теории T пусть $I(T) := \{\sigma l, \sigma r : l = r \in T, \sigma \in \Sigma\}$, т. е. $I(T)$ состоит из всех примеров l и r.

Пусть $G(T) := \{\hat{\sigma} l, \hat{\sigma} r : l = r \in T, \hat{\sigma} = [\pi \leftarrow x], \pi \in \Pi(l)$ или $\pi \in \Pi(r), x \in X\}$, т. е. конечное множество всех обобщений l и r. При этом мы предполагаем, что термы берутся с точностью до переименования переменных, т. е. на самом деле рассматривается фактормножество $GT/_{\approx}$.

Определим характеристическое множество $\chi(T)$ эквациональной теории T:

$$\chi(T) := I(T) \cup G(T).$$

Пусть $\mathscr{E}(T)$ некоторое свойство первого порядка теории T. Если свойство $\mathscr{E}$ рассматривается только относительно подмножества θ всего множества F_Ω, $\theta \in F_\Omega$, мы будем использовать запись $\mathscr{E}(T)|_\theta$.

Определение. Теория T называется *χ-сводимой* тогда и только тогда, когда

$$\text{из } \mathscr{E}(T)|_{\chi(T)} \text{ следует } \mathscr{E}(T).$$

Пусть $\mathbf{T}_\mathscr{E}$ — класс теорий, характеристический для свойства $\mathscr{E}$. Тогда *χ-подклассом* $\chi\mathbf{T}_\mathscr{E} \subseteq \mathbf{T}_\mathscr{E}$ называется множество

$$\chi T_\mathscr{E} := \{T \in T_\mathscr{E} : T \text{ } \chi\text{-сводимо}\}.$$

В [SS81.5] показано, что

$$\mathbf{AC}^*\mathbf{M}_\infty = \chi\mathbf{AC}^*\mathbf{M}_\infty$$

и, следовательно, $\mathbf{AC}^*\mathbf{M}_{\omega 1} = \chi\mathbf{AC}^*\mathbf{M}_{\omega 1}$. Это уже упрощает проверку свойства $\mathbf{T} \in \mathbf{AC}^*\mathbf{M}_\infty$, так как теперь нам достаточно показать, что это свойство выполняется для проблем выравнивания на $\chi(T)$, т. е. для проблем $\langle s \geqslant t \rangle_T$, где $s, t \in \chi(T)$.

Так как в общем случае $\chi(T)$ бесконечно, остается проблема, как показать, что $(T \in \mathbf{AC}^*\mathbf{M}_\infty)|_{\chi(T)}$. Для решения этой проблемы иногда можно воспользоваться теоремой компактности.

Однако для некоторых классов теорий эту проблему можно полностью свести к проверке выполнимости свойства на *конечном тестовом множестве* $\mathrm{TEST}(T) \subseteq \chi(T)$, таком, что

$$\text{из } \mathscr{E}(T)|_{\theta_1} \text{ следует } \mathscr{E}(T)_{\theta_2},$$

где $\theta_1 = \mathrm{TEST}(T)$ и $\theta_2 = \chi(T)$.

Пусть $\mathbf{T}_\mathscr{E}$ — класс теорий, характеристический для свойства $\mathscr{E}$. Тогда *локальный подкласс с конечным тестовым множеством* — это множество

$\lambda\mathbf{T}_\mathscr{E}$: $= \{T \in \chi\mathbf{T}_\mathscr{E}$: существует *конечное* множество $\mathrm{TEST}(T) \subset \chi(T)\}$.

В частности, мы можем говорить, что некоторая теория T *локально унитарна* и писать $T \in \lambda\mathbf{U}_1$, или *локально финитарна* — $\lambda\mathbf{U}_\omega$, или *локально инфинитарна* — $\lambda\mathbf{U}_\infty$.

По определению

$$\lambda\mathbf{U}_i \subseteq \mathbf{U}_i, \text{ где } i \in \{1, \omega, \infty\}.$$

но, разумеется, главный интерес представляет включение в другую противоположную сторону. Основной результат этого раз-

дела, содержащийся ниже, показывает справедливость такого включения для унитарных теорий.

Другой пример — это теорема Кнута — Бендикса: $\lambda \mathbf{C} = \mathbf{C}$ [KB70].

Пусть $\mathbf{M} = \mathbf{M}_1 \cup \mathbf{M}_\omega \cup \mathbf{M}_\infty$ — класс теорий, *релевантных выравниванию* (т. е. $\lambda M \Sigma_T$ всегда существует). Здесь мы имеем следующую теорему существования:

Теорема 2.3. $\mathbf{T}^*_= \subseteq \mathbf{M}$, *т. е. для регулярных теорий всегда существует наиболее общее выравнивающее множество.*

Теорема 2.4. $\mathbf{F}_\Omega \subseteq \mathbf{T}^*_=$, *т. е. любая* Ω*-свободная теория является регулярной.*

В заключение мы определяем класс $\mathbf{P}$ *перестановочных теорий*, как таких теорий, которые имеют конечные классы эквивалентности:

$$\forall T \in P\colon\ \forall t \in F_\Omega [t]_{=_T} \text{ — конечен.}$$

Для этого класса имеют место следующие результаты.

Лемма 2.4. $\mathbf{P} = \mathbf{P}^*$, *т. е. класс регулярных перестановочных теорий в точности совпадает с классом всех перестановочных теорий.*

Теорема 2.5. $\mathbf{P} \subset \mathbf{U}$, *т. е. для любой теории* $T \in \mathbf{P}$ $\mu U \Sigma_T$ *всегда существует.*

Следствие. $\mathbf{P} \subset \mathbf{M}$.

Теорема 2.6. $\mathbf{P} \subset \mathbf{A}$, *т. е. перестановочные теории являются допустимыми теориями; отсюда, так как* $\mathbf{N} = \mathbf{AU}$ *по определению, получаем*

Следствие. $\mathbf{P} = \mathbf{NP}$, т. е. перестановочные теории являются нормальными теориями. [SZ82] содержит намного больше примеров теорем, которые укрепляют нашу позицию: рассмотренные выше теории являются одними из стандартных деталей «конструктора»; выбирая подходящие детали, можно по желанию строить новые проблемы и теории.

Доказательства приведенных здесь результатов, а также результаты, дополняющие их в связи с классификацией эквациональных теорий, можно найти в [SS82.4].

Список обозначений

$\mathbf{T}_=$: эквациональные теории, конечно определенные, разрешимые

$\mathbf{T}^*$: регулярные теории

$\mathbf{T}_{\Rightarrow}$: T имеет переписывающую систему
$\mathbf{T}_{\leqslant}$: проблема выравнивания разрешима
$\mathbf{A}$: допустимые теории
$\mathbf{T}_{\downarrow}$: конфлюентные теории
$\mathbf{R}$: нетеровы теории
$\mathbf{R}_{\downarrow}$: канонические теории
$\mathbf{C}$: $\mathbf{C} \subset \mathbf{R}_{\downarrow}$: со стандартной канонизацией
$\mathbf{U}_1$, $\mathbf{U}_\omega$: теории унитарные, финитарные,
$\mathbf{U}_\infty$, $\mathbf{U}_0$: инфинитарные, типа нуль
$\mathbf{M}_1\mathbf{M}_\omega$: унитарные по выравниванию, и т. д.
$\mathbf{U}$: $\mathbf{U} = \mathbf{U}_1 \cup \mathbf{U}_\omega \cup \mathbf{U}_\infty$, $\mu U \Sigma_T$ существует, релевантные унификации
$\mathbf{M}$: $\mathbf{M} = \mathbf{M}_1 \cup \mathbf{M}_\omega \cup \mathbf{M}_\infty$, $\mu M \Sigma_T$ существует, релевантные выравниванию
$\mathbf{F}_\Omega$: Ω-свободные теории
$\lambda\mathbf{T}$, $\chi\mathbf{T}$: локальные теории, χ-сводимые теории
$\mathbf{P}$: перестановочные теории с конечными классами эквивалентности
$\mathbf{N}$: нормальные теории: $\mathbf{N} = \mathbf{AU}$

3. ХАРАКТЕРИСТИКА УНИТАРНЫХ ТЕОРИЙ

В 1975 г. П. Хейес высказал предположение, что алгоритм унификации Робинсона для свободных термов, очень возможно, демонстрирует единственный случай унификации, когда существует максимум один наиболее общий унификатор. Это неверно; например, пусть $T_a := \{a = a\}$ для любой константы a, тогда $T_a \in \mathbf{U}_1$.

Однако эта проблема оказалась более сложной, чем предполагали в то время: например, пусть $T_{aa} = \{f(a, a) = a\}$ для любой константы a, тогда

$$T_{aaa} \in \mathbf{U}_1.$$

Следующие две теоремы окончательно устанавливают границу между финитарными и унитарными теориями, но формальное изложение их доказательств заняло бы слишком много места и поэтому не включено в настоящую статью.

Теорема 3.1. $\lambda\mathbf{U}_1 \subset \mathbf{U}_1 \subset \mathbf{F}_\Omega$.

Теорема 3.2. $\lambda\mathbf{U}_1 = \mathbf{U}_1$.

Конечное тестовое множество $\mathrm{TEST}(T)$ для $T \in \mathbf{U}_1$ получается так:

$$\mathrm{TEST}(T) := \{t \in I(T): |t| \leqslant \max(T)\} \cup G(T),$$

где $\max(T) = \max\{|l|, |r| : l = r \in T\}$.

Чтобы пояснить использование приведенных выше теорем, рассмотрим пустую теорию T_ε, т. е. проблему унификации Робинсона для свободных термов. В каменном веке теории унификации, чтобы показать, что $T_\varepsilon \in \mathsf{U}_1$, нужно было изобретать специальный алгоритм и доказывать его полноту и корректность [RO65], [KB70].

Более изысканный метод содержится в [HT76]: факторизуя F_Ω по $\approx$, можно показать, что $F_\Omega/_\approx$ образует полную структуру относительно $\leqslant_T$. Следовательно, если два терма унифицируемы, то существует общий пример, а значит, существует и точная верхняя грань, являющаяся наиболее общим примером с этим свойством. Таким образом, получаем, что $T_\varepsilon \in \mathsf{U}_1$.

В то же время этот результат получается сразу, если воспользоваться сформулированными выше теоремами. Так как абсолютно свободная алгебра термов является, в частности, Ω-свободной, то $T_\varepsilon \in \mathsf{F}_\Omega$. В данном случае любое тестовое множество пусто, так как $\chi(T_\varepsilon)$ пусто. Следовательно, в тестовом множестве не существует пары с более чем одним наиболее общим унификатором. Отсюда $T_\varepsilon \in \mathsf{U}_1$.

ЛИТЕРАТУРА

[BA73] Baxter L. D. An Efficient Unification Algorithm. — Univ of Waterlloo, Techn. Report CS-73-23, 1973.

[BA78] Baxter L. D. The Undecidability of the Third Order Dyadic Unification Problem; — Information and Control, vol. 38, no. 2, 1978.

[DA73] Davis M. Hilbert's tenth Problem is unsolvable. — Amer. Math. Monthly, vol. 80, 1973.

[FA79] Fay M. First. Order Unification in an Equational Theory. — Proc. 4-th Workshop on Autom. Deduction, Texas, 1979.

[G67] Guard J. R., Oglesby F. C., Benneth J. H., Settle L. G. Semi-Automated Mathematics, JACM 1969, vol. 18, no. 1.

[GO81] Goldfarb D. The Undecidability of the Second Order Unification Problem. — J. of Theor. Comp. Sci. 13, 1981.

[GO66] Gould W. E. A Matching Procedure for ω-order Logic (thesis). — Air Force Cambridge Research Labs, 1966.

[GR79] Grätzer G. Universal Algebra. — Springer-Verlag, 1979.

[HE30] Herbrand J. Recherches sour a theorie de la demonstration. — Travaux de la Soc. des Sciences et des Lettres de Varsovie, no. 33, 128, 1930.

[HM67] Хмелевский Ю. Решение некоторых систем уравнений в словах. — ДАН СССР 1964, т. 156, № 4, с. 749—751; т. 171, № 5, с. 1047—1049; т. 177, № 5, с. 1023—1025.

[HT73] Huet G. The Undecidability of Unification in Third Order Logic, Information and Control 22, 1973.

[HT76] Huet G., Résolution d- équations dans des langages d'ordere 1, 2, ..., ω. Thésé d'Etat, Univ. de Paris, VII, 1976.

[HT75] Huet G. A Unification Algorithms for Typed Lambda Calculus. — J. Theoretic. Comp. Sci., 1, 1975.

[HT80] Huet G., Oppen D. C., Equations and Rewrite Rules, In: Formal Languages: Perspectives and Open Problems Ed. R. Book, Academic Press, 1980.

[HT80] Huet G. Confluent Reductions: Abstract Properties and Applications to Term Rewriting Systems. — J. ACM, vol. 27, no. 4, 1980.

[HU80] Hullot J. M. Canonical Forms and Unification. — Proc. 5-th Workshop on Automated Deduction, Springer Lecture Notes, 1980.
[HU80.1] Hullot J. M. A. Catalogue of Canonical Term Rewriting Systems. — Reaserch Rep. CSL-113, SRI-International, 1980.
[H82] Herold A. Universal Unification and extended regular ACFM Theories. — Univ. Karlsruhe, 1982.
[KB70] Knuth D., Bendix P. Simple Word Problems in Universal Algebras. — In: Comp. Problems in Abstract Algebra. — J. Leech (ed), Pergamon Press, 1970.
[KO75] Kovalski R. A Proof Procedure based on Connection Graphs. — J. ACM, vol. 22, no. 4, 1975.
[LA79] Lankford D. S. A. Unification Algorithm for Abelian Group Theory. — Rep. MTR-1, Lousiana Techn. Univ., 1979.
[LB79] Lankford D. S., Ballantyne M. The Refutation Completeness of Blocked Permutative Narrowing and Resolution. — 4-th Workshop on Autom. Deduction, Texas, 1979.
[LS76] Livesey M., Siekmann J. Unification of Sets and Multisets. — Univ. Karlsruhe, Techn. Report, 1976.
[LS75] Livesey M., Siekmann J. Termination and Decidability Results for Stringunifikation. — Univ. of Essex, Memo CSM-12, 1975.
[LO80] Loveland D. Automated Theorem Proving. — North Holland, 1980.
[MM79] Martelli A., Montaneri U. An Efficient Unification Algorithm. — University of Pisa, Techn. Report, 1979.
[MA70] Matiyasevich Y. Diophantine Representation of Rec. Enumerable Predicates. — Proc. of the Scand. Logic Symp., North Holland, 1978.
[MA77] Маканин Г. С. Проблема разрешимости уравнений в свободной полугруппе. — Математ. сб., 1977, т. 103, вып. 2, с. 3—147.
[NI80] Nilson N. Principles of Artificial Intelligence. — Tioga Publ. Comp., California, 1980.
[PW78] Paterson M., Wegman M. Linear Unification. — J. of Comp. and Syst. Science, 1978, 16.
[PR60] Prawitz D. An Improved Proof Procedure. — Theoria 26, 1960.
[PS81] Peterson G., Stickel M. Complete Sets of Reductions for Equational Theories with Complete Unification Algorithms. — J. ACM, vol. 28, no 2, 1981.
[PL72] Plotkin G. Building in Equational Theories. — Machine Intelligence, vol 7, 1972.
[RS78] Raulefs P., Siekmann J. Unification of Idempotent Functions. — Universitat Karlsruhe, 1978.
[RSS79] Raulefs P., Siekmann J., Szabo P., Unvericht E. A. Short Survey on the State of the Art in Matching and Unification Problems. — SIGSAM Bulletin, 13, 1979.
[RO65] Robinson J. A. A Machine Oriented Logic based on the Resolution Principle. J. ACM 12, 1965. [Имеется перевод: Робинсон Дж. Машинно-ориентированная логика, основанная на принципе резолюции. — В кн.: Киберн. сб. — М.: Мир, 1970, вып. 7, с. 194—218.]
[RO71] Robinson J. A. Computational Logic: The Unification Computation. — Machine Intelligence, vol 6, 1971.
[SI75] Siekmann J. Stringunification. — University of Essex, Memo CSM-7.
[SL74] Slagle J. ATP for Theories with Simplifiers. Commutativity and Associativity. — J. ACM 21, 1974.
[ST75] Stickel M. A complete Unification Algorithm for assoc. commutative function. — Proc. of 4th IJCAI, Tbilisi, USSR, 1975.
[SI82] Siekmann J. Unification of Commutative Terms (submitted). — Universitat Karlsruhe, 1982.
[SS82.2] Szabo P., Siekmann J. A Minimal Unification Algorithm for Idempotent Functions. — Universitat Karlsruhe, 1982.

[SS82.3] Szabo P., Siekmann J. Unification in Idempotent Semigroups. — Universitat Karlsruhe, 1982.
[SS82.4] Szabo P., Siekmann J. Universal Unification and a Classification of Equational Theories. — Universitat Karlsruhe, 1982.
[SS81.5] Szabo P., Siekmann J. Universal Unification and Regular ACFM Theories. — Proc. of IJCAI-81.
[S78] Szabo P. The Undecidability of the D_A-Unification Problem. — Universitat Karlsruhe, 1978.
[SZ82] Szabo P. Theory of First Order Unification (in German, thesis). — Universitat Karlsruhe, 1982.
[VA75] van Vaalen J. An Extension of Unification to Substitutions with an Application to Automatic Theorem Proving. — IJCAI-4, Proc. of 1975. [Имеется перевод: Ваален Д. Распространение понятия унификации на подстановки в автоматическом доказательстве теорем. — В кн.: Труды 4-й Международной объединенной конференции по искусственному интеллекту — (Тбилиси, сент. 1975 г.). — Москва, 1975, т 3, с. 37—48.]
[VO78] Vogel E. Unifikationsalgorithmen für Morphismen.—Universität Karlsruhe (in German, Diploma thesis), 1978.
[WR73] Wos L., Robinson G. Maximal Models and Refutation Completeness: Semidecision Procedures in Automatic Theorem Proving. — In: Word problems (W. W. Boone, F. B. Cannonito, R. C. Lyndon, eds), North Holland, 1973.
[WR67] Wos L., Robinson G. A., Carson D., Shalla L. The Concept of Demodulation in Theorem Proving. — J. ACM, vol 14, no 4.
[SS82.1] Szabo P., Siekmann J. A Noetherian and Confluent Rewrite System for Idempotent Semigroups. — Smigroup Forum 1982.
[LA80] Lankford D A new complete FPA-Unification algorithm. — Jan. 1980, MTP-8.
[FA81] Fages F. INRIA report, France.

Решение некоторых открытых проблем с помощью программы для автоматического доказательства теорем [1)]

Л. Воз [2)]

Резюме

Основная цель настоящей работы состоит в том, чтобы продемонстрировать возможность использования программы автоматического доказательства теорем в качестве интеллектуального партнера. Причем эта возможность — не из области предположений. Мы предъявим несколько открытых проблем, решенных при непосредственном участии такого рода программы.

Хотя в каждом из наших примеров применялась одна и та же программа AURA [19] (разработанная совместно Аргоннской национальной лабораторией и Университетом Северного Иллинойса), многочисленные исследования показывают, что такого же рода результаты могли бы быть получены с помощью других программ для доказательства теорем. Помня об этом, мы можем с уверенностью заявить, что в области автоматического доказательства теорем достигнута важная цель — программы, доказывающие теоремы, могут с успехом применяться в различных областях.

Открытые проблемы, о которых пойдет речь, касаются тернарной булевой алгебры, конечных полугрупп, исчисления эквивалентности и синтеза вентильных схем. Но нет сомнений, что, несмотря на наш успех в столь различных областях, многие смотрят весьма скептически на возможность автоматизации глубоких рассуждений.

Природа этого скептицизма заставила нас поставить в этой работе и другую цель: преодолеть, хотя бы частично, сопротивление, возникающее на пути такого рода автоматизации. Мы обсудим два основных заблуждения, на которых основана неверная оценка полезности и потенциальных возможностей автоматического доказательства теорем. Опровергнув эти заблуждения, мы устраним существенное препятствие к тому, чтобы программа автоматического доказательства теорем играла свою истинную роль — роль коллеги и помощника.

[1)] Wos L., Solving open questions with an automated theorem proving program, Lecture Notes in Computer Science, v. 138 (1982), 1—31.

[2)] Argonne National Laboratory, Argonne, Illinois, U. S. A.

1. ВВЕДЕНИЕ

По поводу автоматического доказательства теорем часто задают один и тот же вопрос: когда, наконец, программы будут доказывать новые теоремы? Вопрос этот естественен и вполне осмыслен. Само название области вызывает соответствующий образ. К счастью, сейчас мы можем сказать: программы могут получать новые результаты и уже получили их достаточно много.

В настоящей работе мы кратко обсудим некоторые из открытых проблем, которые были решены с помощью программы автоматического доказательства теорем. Эти проблемы взяты из следующих областей: тернарной булевой алгебры [21], конечных полугрупп [22], формальной логики [26] и синтеза вентильных схем [20, 24]. При их решении программа использовалась различными способами. Обобщая, можно сказать, что программа вела себя как коллега. Коллеги иногда представляют всю информацию для окончательного решения задачи, иногда дают идею решения, а иногда и полностью ошибаются. Точно так же и программа иногда действовала успешно, а иногда ошибочно. Тем не менее, представленный здесь материал показывает, что такого рода программа может помочь в проведении рассуждений. Ее можно также рассматривать как некое дополнение к вычислителю.

Мы ограничимся кратким описанием методологии, использованной для решения упомянутых проблем, поскольку соответствующие детали можно найти во многих источниках. Сказанного, однако, будет достаточно, чтобы проиллюстрировать взаимодействие между программой и исследователем. Эта кооперация весьма напоминает работу с консультантом или сотрудником. Мы не будем делать обзор всей области, а сконцентрируем внимание только на нашей работе. В разд. 2 приводятся результаты, полученные к настоящему времени. В разд. 3 описываются методы и техника, с помощью которых эти результаты получены. В разд. 4 мы рассмотрим некоторые заблуждения, которые препятствуют признанию всей этой области в целом. В разд. 5 обсуждается наша нынешняя деятельность и планы на будущее.

Хотя все проблемы, о которых пойдет речь, были решены с помощью единственной программы для доказательства теорем, использованная нами методология применима ко всей области машинного доказательства в целом. Читатель должен постоянно помнить следующий существенный факт:

для решения перечисленных здесь проблем
не потребовалось никакой дополнительной
программистской работы.

Хотя для каждой из затронутых нами областей следовало сформулировать задачу в виде подходящего множества дизъюнктов и выбрать свою руководящую стратегию, все исследования проводились с помощью стандартной программы для доказательства теорем. Эта программа AURA [19] была совместно разработана Аргоннской национальной лабораторией и Университетом Северного Иллинойса.

2. ОТКРЫТЫЕ ПРОБЛЕМЫ, РЕШЕННЫЕ С ПОМОЩЬЮ ПРОГРАММЫ ДЛЯ АВТОМАТИЧЕСКОГО ДОКАЗАТЕЛЬСТВА ТЕОРЕМ

В этом разделе мы сформулируем решенные нами проблемы. В следующем разделе коротко обсудим подход, который позволил получить их решение.

2.1. Тернарная булева алгебра

Тернарная булева алгебра — это непустое множество с тернарной операцией f и унарной g, которые удовлетворяют следующим пяти аксиомам:

1) $f(f(v, w, x), y, f(v, w, z)) = f(v, w, f(x, y, z))$,
2) $f(y, x, x) = x$,
3) $f(x, y, g(y)) = x$,
4) $f(x, x, y) = x$,
5) $f(g(y), y, x) = x$.

Здесь f играет роль «произведения», g — взятия «обратного».

Являются ли какие-нибудь из этих пяти аксиом независимыми от остальных? Было известно, что 4) и 5) зависимы, причем соответствующие доказательства были получены различными программами, например, Алленом и Лакхэмом [1], Дж. Робинсоном и Возом (не опубликовано). Но поводу аксиом 1), 2) и 3) вопрос оставался открытым. В [21] был сформулирован подход, позволивший установить их независимость.

Чтобы установить зависимость, программу для доказательства теорем можно использовать обычным образом как инструмент для поиска доказательства, в то время как для установления независимости эта программа используется совершенно иначе, поскольку для этого программа должна уметь строить модели — модели, на которых выполняются какие-либо четыре аксиомы и не выполняется пятая. Фактически такая модель задается в виде таблицы, по которой можно проверить действительно ли «произведения» и «обратные» таковы, как это требуется теми аксиомами, которые должны выполняться или нарушаться на данной модели. Важно здесь то, что программа для доказательства теорем используется в новом качестве — как инструмент для построения моделей и контрпримеров. Этот инте-

ресный и существенный для нашей работы подход был сформулирован Винкером в [23].

Прежде чем продолжить, коротко ответим на следующий естественный вопрос: почему бы не извлекать требуемую модель из дерева поиска доказательства? Возражение против такого подхода связано с тем фактом, что многие из интересующих нас задач, по сути, второго порядка. Конечно, можно использовать стандартные приемы и погрузить их в язык первого порядка, но остается серьезная проблема построения завершенного дерева поиска (т. е. такого, из которого получается нужное доказательство). Природа же наших моделей такова, что может потребоваться построение дерева крайне большого, практически недоступного объема. Поэтому мы предпочитаем, когда это необходимо, непосредственно строить нужную модель.

2.2. Конечные полугруппы

И. Капланский обратил наше внимание на следующий (тогда открытый) вопрос. Существуют ли конечные полугруппы, которые обладают нетривиальными антиавтоморфизмами и не обладают нетривиальными инволюциями? Антиавтоморфизмом называется такое отображение h полугруппы на себя, что:

1) h — биекция;

2) $h(x \cdot y) = h(y) \cdot h(x)$.

Антиавтоморфизм нетривиален, если он не является тождественным. Нетривиальная инволюция — это нетривиальный антиавтоморфизм, квадрат которого тождествен. Здесь мы сталкиваемся с той же альтернативой, как и в случае с тернарной булевой алгеброй. Если существование нетривиального антиавтоморфизма влечет за собой существование нетривиальной инволюции, то методом решения задачи является стандартный поиск доказательства. Если же это не так и существует полугруппа требуемого типа, то следует искать модель (или контрпример). Мы избрали метод порождения моделей и с успехом его применили.

При этом совершенно очевидно, что программа не должна перебирать все конечные полугруппы, начиная с наименьшей возможной, поскольку уже полугрупп порядка 6 слишком много. Существенно также исключить из рассмотрения коммутативные полугруппы, так как их антиавтоморфизмы тривиальны. С помощью нашего подхода, детали которого описаны в разд. 3.2, была построена требуемая полугруппа порядка 83.

Зная теперь, что такие полугруппы существуют, естественно поставить вопрос о минимальности. Снова желательно отказаться от полного перебора, и это оказалось возможным. Минимальная полугруппа требуемого типа имеет порядок 7.

Наконец, возникает вопрос о количестве таких минимальных полугрупп. Программа нашла их ровно четыре. Естественно, при этом необходимо было проверять изоморфизм.

2.3. Логическое проектирование вентильных схем

2.3.1. Проектирование

Как приспособить программу, доказывающую теоремы, для проектирования схем? Может ли при этом программа найти схему, более эффективную, чем известные?

Рассматривались схемы, основанные на четырехзначной логике. Технология их изготовления использует так называемые T-вентили. T-вентиль является основным компонентом схем, основанных на многозначной логике. Его аналогом в случае двузначной логики является мультиплексор. Существовали схемы, построенные на T-вентилях, которые были вполне эффективны и даже казались минимальными. В. Войцеховский предположил, что можно попытаться найти более эффективные схемы, используя программу для доказательства теорем.

Действительно, подходящее кодирование задачи [24] для нашей программы привело в большинстве рассмотренных случаев к построению вентильных схем, более эффективных, чем известные. Следует отметить два важных момента. Во-первых, была найдена новая область приложения для программ, доказывающих теоремы. Эти задачи требуют кодирования таблиц типа вход — выход и аксиом, описывающих нужное поведение компонент. Во-вторых, специалист, не занимающийся автоматическим доказательством теорем, предпочел использовать нашу программу, а не писать проблемно-ориентированную.

2.3.2. Отказы

При проектировании схем необходимо добиваться отсутствия отказов. Отказом называется ситуация, при которой некоторый сигнал достигает данной точки быстрее, чем ожидалось, и, более того, быстрее, чем это необходимо. Наличие отказа может, например, привести к тому, что все сигналы станут одновременно равны 1, а это особенно нежелательно. А. Войцик предложил задачу построения программы, обнаруживающей отказы. Для этого Винкер сумел воспользоваться [20] весьма изощренной алгеброй отказов [2], после чего могла быть использована наша программа, которая успешно обнаруживала отказы, если ей предъявлялась схема, содержащая таковые.

Затем возникла проблема проверки того, что условия, при которых возможен отказ, не выполняются. И снова Винкер предложил метод [20], используя который программа осуществляла такую проверку.

2.4. Формальная логика

Мы переходим к наиболее интересным и, вероятно, наиболее трудным задачам, которые решила наша программа. Вопрос состоит в том, существуют ли новые наиболее короткие формулы исчисления эквивалентности, которые могли бы служить единственной аксиомой этого исчисления [6, 16]. Формулами этого исчисления являются выражения, построенные очевидным образом из переменных x, y, z, w, ... и 2-местного функционального символа E ($E(x, y)$ можно читать как «x эквивалентно y»). Легко видеть, что теоремами этого исчисления являются в точности те формулы, в которые каждая переменная входит четное число раз [3]. Стандартными правилами вывода являются правило подстановки и правило отрывания. Однако вместо них часто используется единственное правило вывода, которое называется «отрывание с насыщением».

Определение. Правило отрывания — это правило вывода, которое по двум формулам $E(A, B)$ и A дает формулу B.

Определение. Пусть даны две формулы $E(A, B)$ и A', не имеющие общих переменных. Отрывание с насыщением — это правило вывода, которое по данным двум формулам дает формулу B'', получаемую применением правила отрывания к $E(A'', B'')$ и A'', где $E(A'', B'')$ и A'' получены соответственно из $E(A, B)$ и A' применением наиболее общей возможной подстановки, выравнивающей A и A'.

Итак, для того, чтобы применить правило отрывания с насыщением, следует взять две формулы, переименовать в них переменные так, чтобы одна и та же переменная не входила в обе формулы, затем найти наиболее общую подстановку, после которой стало бы возможным применение правила отрывания, и, наконец, применить отрывание к паре формул, полученных из исходных в результате этой подстановки. Например, применение отрывания с насыщением к $E(E(E(x, x), z), z)$ и $E(E(x, y), E(y, x))$ дает либо $E(x, x)$, либо $E(x, E(E(y, y), x))$ в зависимости от того, в каком порядке рассматриваются посылки.

Имеется 630 формул исчисления эквивалентности, каждая из которых содержит пять вхождений символа E и ровно по два вхождения каждой переменной. Длина каждой из этих 630 формул равна 11 (длиной мы называем количество символов, не считая скобок). Некоторые из этих формул длины 11 настолько сильны, что каждая из них может служить единственной аксиомой для всего исчисления (т. е. любая теорема может быть из нее выведена). К тому времени, как мы занялись этим вопросом, лишь по поводу семи из этих 630 формул не было известно, может ли каждая из них служить единственной аксио-

мой. Существовало предположение [4], что ни одна из этих семи таковой не является.

Типичным методом доказательства того, что данная формула некоторого исчисления слишком слаба, чтобы быть единственной аксиомой этого исчисления, является следующий. Ищем модель, на которой рассматриваемая формула верна, а какая-либо из известных аксиом опровергается. Если такую модель удалось найти, то действительно данная формула не может служить единственной аксиомой, так как тем самым доказано, что одна из теорем нашего исчисления, а именно, одна из его аксиом, не следует из данной формулы. Применение этого стандартного подхода может привести к тому, что придется перебрать очень много моделей, ни одна из которых не обладает нужными свойствами. Ввиду большого числа не устраивающих нас моделей и, что существенней, поскольку мы начали исповедовать «новую» мощную методологию, мы решили вообще отказаться от рассмотрения таких моделей.

Мы придумали метод, который можно было полностью реализовать в рамках имевшейся техники машинного доказательства и который позволял проверить, выводима ли некоторая совокупность теорем из данных формул. Для доказательства того, что некоторая формула не может быть взята в качестве единственной аксиомы, этот метод работал следующим образом. Для данной формулы строилось описание всех теорем, выводимых из нее при помощи отрывания с насыщением. Затем рекурсией по этому описанию показывали, что некоторая известная аксиома исчисления эквивалентности не может быть так описана и, следовательно, не содержится среди формул, выводимых из рассматриваемой. Из этого немедленно следует, что данная формула не может служить единственной аксиомой исчисления. Но, видимо, более интересным и, ввиду упомянутого выше предположения, несколько неожиданным оказался тот факт, что были обнаружены две из проверявшихся формул, каждая из которых в действительности является единственной аксиомой, ранее неизвестной.

Если проводить каждое из двух соответствующих доказательств с использованием лишь отрывания с насыщением, то доказательства эти оказываются весьма длинными и сложными. Но удалось найти такое их представление, которое позволило построить оба доказательства с помощью программы. Так как при этом шаг доказательства не состоял в применении отрывания с насыщением, то был предложен способ перевода этих доказательств в стандартные. Это преобразование также выполнялось программой доказательства теорем, чтобы получить доказательство в удобной для логиков форме.

Чтобы продемонстрировать трудности, с которыми пришлось

столкнуться, заметим, что одно из упомянутых доказательств содержало 162 применения правила отрывания с насыщением. Причем на некоторых шагах встречались формулы, длина которых (количество символов за исключением запятых и скобок) превосходила 95, а одна из формул была длины 103. Пожалуй, ни одна из стандартных проблемно-ориентированных программ не смогла бы справиться с таким доказательством.

3. МЕТОДОЛОГИЯ

Изложение в настоящем разделе ведется параллельно предыдущему, но здесь мы акцентируем внимание на методологических вопросах, рассматривая их с точки зрения машинного доказательства теорем. Начнем с аналога разд. 2.1.

3.1. Тернарная булева алгебра

Изучение тернарной булевой алгебры непосредственно привело к расширению сферы применения программ для доказательства теорем. Если до начала исследований эти программы использовались практически всегда с единственной целью поиска доказательства, то после завершения работ стало ясно, что их (программы) можно применять также для построения моделей и контрпримеров. Такая модель или контрпример представляет собой множество дизъюнктов, которое может быть представлено, например, в форме таблицы, задающей отношения между различными элементами. Анализ этой таблицы показывает, истинны или нет соответствующие «произведения» и «обратные».

Метод, который мы использовали, является хорошей иллюстрацией возможного сотрудничества исследователя и программы. Первым нашим шагом была попытка найти доказательство зависимости аксиомы 2 от остальных четырех. Она не удалась, поскольку (как ныне известно) аксиома 2 независима.

Для доказательства независимости мы строили модель, на которой одновременно выполнялись 1), 3), 4) и 5), но опровергалась 2). В качестве входных были взяты неединичные дизъюнкты, гарантирующие выполнимость 1), 3), 4) и 5). Используемый далее механизм продемонстрирован в Приложении, причем видно, что процесс выглядит точно так же, как процесс построения доказательства. Упомянутые неединичные дизъюнкты являются ядерными для процедуры, использующей гиперрезолюцию [12, 18]. (Напомним, что ядерными дизъюнктами в такой процедуре называются неединичные и вдобавок содержащие хотя бы одну отрицательную литеру, а чисто положительные дизъюнкты называются сателлитными.)

Мы начали поиск модели с предположения о количестве

элементов, которых было бы достаточно, и предположили, что хватит трех. Так как ранее мы обнаружили, что при наличии аксиом 1)—3) утверждение $g(g(x)) = x$ выполняется для всех x, то достаточно было найти элемент предполагаемой модели, для которого упомянутое утверждение оказалось бы неверным, из чего и следовала бы невыполнимость аксиомы 2). Пусть теперь мы имеем некоторую частичную модель, т. е. программе заданы предположения о «произведениях» и об «обратных».

Ядерные дизъюнкты «навязывают» выполнимость некоторых соотношений. Так, например, ядро, соответствующее аксиоме 4), $\neg Q(x) \neg (Q(y) \mathrm{EQVAL}(f(x, x, y), x)$ требует выполнения соотношения $f(a, a, a) = a$, где a — один из элементов рассматриваемой частичной модели. Предикат Q означает: «является элементом данной модели». Следовательно, к входным дизъюнктам необходимо добавить: $Q(a)$, $Q(b)$ и $Q(c)$.

Затем применяется правило гиперрезолюции с упомянутым ядром, которое таким образом играет роль теста для проверки того, что все подходящие тройки элементов согласуются с четвертой аксиомой. Аналогичным образом используются остальные ядерные дизъюнкты.

Демодуляторы, которые соответствуют имеющимся в данной частичной модели «произведениям» и «обратным», являются также входными дизъюнктами и/или добавляются по мере развития процесса ее построения. Эти демодуляторы или переписывающие правила автоматически применяются к дизъюнктам, порожденным гиперрезолюцией. Пусть в данный момент времени порожден некоторый дизъюнкт, не являющийся примером $\mathrm{EQUAL}(x, x)$, причем участвовавшее в его порождении ядро соответствует одной из аксиом, которая должна выполняться. Далее имеются две возможности. Либо обнаружено новое соотношение, которое должно выполняться и в дальнейшем, либо полученный дизъюнкт противоречит имеющимся предположениям о том, что уже включенные в данную модель элементы различны. В первом случае мы добавляем новое соотношение к имеющимся и продолжаем процесс построения модели. Во втором случае мы должны заменить некоторые из имеющихся соотношений другими, поскольку тот выбор, который был сделан к данному моменту, не позволяет завершить процесс построения модели. К примеру, мог быть выведен дизъюнкт $\mathrm{EQUAL}(b, a)$, что неприемлемо, поскольку мы предположили, что элементы a и b различны. В таком случае программа возвращается и пытается найти другие значения для одного или нескольких «произведений» и/или для одного или нескольких «обратных».

Итерируя эту процедуру, программа нашла модель требуемого типа, тем самым доказав независимость аксиомы 2.

Наконец, для того чтобы обнаружить ту самую тройку элементов, на которой опровергается аксиома 2, добавим ядро вида $\neg Q(x) \neg Q(y) \mathrm{EQUAL}(f(y,x,x),x)$. Если построение модели успешно завершено, т. е. все «произведения» и «обратные» определены, то должна существовать тройка термов, которую нельзя демодулировать в некоторый пример условия рефлексивности. Эта тройка будет найдена, когда выполнится гиперрезолюция с указанным ядром и будут сделаны различные подстановки, среди которых имеется и та, что нас интересует.

3.2. Конечные полугруппы

Свои исследования мы начинали со следующего вопроса: существует ли конечная полугруппа, одновременно допускающая нетривиальные антиавтоморфизмы и не допускающая нетривиальные инволюции? Вначале необходимо было выбрать одну из альтернатив: можно было использовать программу для попытки доказать теорему о том, что таких полугрупп не существует, и можно было попытаться опровергнуть это утверждение, построив контрпример в виде модели. К счастью, мы занялись исследованием второй возможности.

Как будет видно, и в этом случае программа для доказательства теорем играла роль ассистента. Перечислим основные пункты принятого плана исследований.

1. Рассмотреть достаточно большие полугруппы. Чем больше структура, тем больше возможностей для одновременной выполнимости утверждений существования.

2. Выбрать такое представление полугрупп, которое позволило бы работать с достаточно большими.

3. Провести эксперименты с программой с целью выявления условий, гарантирующих конечность полугруппы.

4. Поскольку наличие антиавтоморфизма и/или инволюции зависит от определенного типа симметрии, вначале сделать исследуемую структуру максимально симметричной, чтобы разрешить антиавтоморфизмы, а затем ограничивать симметрию, чтобы исключить инволюции.

Соблюдение первого пункта позволяет изучать нетривиальные полугруппы. Ведь если полугруппа мала, ее структура является весьма жесткой. Однако возникает вопрос о хорошем представлении больших полугрупп.

Существуют два основных способа представления: задание таблицы умножения и задание образующих и соотношений. Последнее хорошо знакомо математикам и позволяет изучать структуру полугруппы в целом. Использование такого представления дает нам компактные обозначения, что облегчает изучение больших структур. (Те модели, которые можно надеяться извлечь из дерева поиска доказательства, были бы заданы

иначе.) При таком представлении элементами полугруппы являются слова в алфавите из образующих, умножение — это конкатенация, для задания соотношений и «явного» выполнения умножения используются демодуляторы (см. Приложение).

Проведенные «вручную» рассуждения показывают, что демодуляторы действительно работают нужным образом. В силу некоторых математических соображений мы решили рассматривать полугруппы с четырьмя образующими. Так, например, чтобы исключить коммутативность, всякий нетривиальный антиавтоморфизм должен иметь четный порядок. Кроме того, он не может быть порядка 2, так как тогда нельзя исключить инволюции. Далее, имея четыре образующих, мы могли бы рассмотреть антиавтоморфизмы, которые их зацикливают. Наконец, не следовало заставлять программу проверять слишком уж громадное количество больших полугрупп.

Эксперименты с программой показали, что конечность полугруппы гарантируется, если добавить соотношение, которое требует, чтобы для некоторого n все слова длины большей или равной n совпадали. Первые попытки дали полугруппу порядка 1 и известную полугруппу порядка 8, обе оказались неудачными.

В последующих экспериментах было добавлено соотношение, требующее, чтобы все слова длины большей или равной четырем совпадали. Программа нашла соответствующую полугруппу с четырьмя образующими и этим единственным соотношением, которая имела порядок 85, но изобиловала симметрией. В результате нашлись не только антиавтоморфизмы, но и масса инволюций. Далее мы сосредоточили внимание на антиавтоморфизме h, отображающем b в c, c в d, d в e и e в b, где b, c, d, e — образующие. Инволюцией, которую мы хотели исключить, было отображение j, меняющее местами b и c и оставляющее на месте d и e.

С этой целью мы добавили соотношение $bc = de$, которое несколько нарушало бы симметрию этой полугруппы порядка 85. Программа, которой было задано это соотношение, обнаружила, что его недостаточно. Инволюции все еще существовали, т. е. симметрия нарушалась не слишком сильно. Однако соотношения $bbc = dde$ уже хватило.

Затем программа использовалась для нахождения следствий из этого соотношения и обнаружила лишь одно: $dcc = bee$. При этом постоянно использовалось наличие отображения h.

Конечно, для программы оставалось еще много работы, например: проверка ассоциативности, проверка того, остается ли h антиавтоморфизмом, проверка отсутствия инволюций. В результате всех этих взаимодействий между исследователем и программой построена нужная полугруппа порядка 83.

Затем мы занялись вопросом о минимальности. В резуль-

тате интенсивной работы программы были обнаружены четыре неизоморфных полугруппы требуемого типа порядка 7. Проверка изоморфизма также осуществлялась программой для доказательства теорем. Ни один из перечисленных вопросов не решался перебором: уже полугрупп порядка 6 слишком много.

Важнейшим моментом, о котором постоянно следует помнить, является роль нашей программы. Она являлась помощником в рассуждениях и ей могли быть поручены многие задачи логической природы.

3.3. Электронные схемы

3.3.1. Проектирование

Объект этого исследования — построение различных дискретных схем. Мы решили воспроизвести подход, принятый инженерами. Инженеры исходят из совокупности спецификаций, которым должна удовлетворять схема, и множества возможных компонент с известными характеристиками, которые разрешено использовать для построения. Затем они часто разбивают исходную задачу на более простые и легче реализуемые подзадачи. Итак, необходимо вначале получить спецификации для этих подсхем, а затем, используя имеющиеся компоненты, собрать из подсхем исходную схему.

Спецификации задавались программе непосредственно — в виде таблицы вход/выход. Затем мы указывали, как разумно разбить эту таблицу на подтаблицы, соответствующие подсхемам. Далее указывалось, как распознать достаточно малые схемы, которые могут быть непосредственно сконструированы. Наконец, были заданы инструкции, как использовать базовые компоненты для сборки большой схемы из подсхем. В качестве базовых компонент были взяты T-вентили.

Затем программе была поставлена задача поиска интересных и эффективных схем. Задача была сформулирована как задача поиска доказательства существования схемы требуемого типа. Классы подсхем, которые не представляли интереса, преобразовывались по мере их обнаружения в «пустое соединение» с помощью демодуляторов. Эти преобразования препятствовали взаимодействию интересных схем с неинтересными.

Поскольку цель состояла в нахождении различных схем требуемого типа, т. е. в проверке большого числа возможных решений, программа была снабжена инструкцией, не позволявшей ей останавливаться после того, как некоторое доказательство найдено. Таким образом, один процесс поиска давал большое количество интересующих нас схем.

Затем из так полученного «доказательства» извлекались конкретные схемы. Большинство из них оказались лучше схем, описанных в имевшейся к тому времени литературе.

Почему же оказалось возможным извлекать схемы из дерева поиска доказательства, а не искать их с помощью непосредственного моделирования? Коротко говоря, нам помогла как специфика заданной информации, так и природа решаемой задачи. Например, мы могли быть полностью уверены в правильности различных декомпозиций на подсхемы.

3.3.2. Отказы

Отказом называется ситуация, в которой некоторый сигнал достигает данной точки быстрее, чем ожидалось, и, более того, быстрее, чем это желательно. Для обнаружения отказов необходимо уметь строить все возможные разворачивающиеся во времени последовательности сигналов, отвечающие различным входам. Существует так называемая алгебра отказов, которая позволяет изучать эти последовательности в дискретные моменты времени в зависимости от изменений входа [2].

Итак, нас интересовал вопрос о наличии и/или отсутствии отказов в данной схеме. Поскольку в тех случаях, которые мы изучали, количество подлежащих рассмотрению ситуаций было невелико, мы применили экстенсивный подход. Наличие или отсутствие отказов устанавливалось просто в результате изучения выхода программы после завершения всего процесса поиска (см. Приложение). Поиск оканчивался, когда были сделаны все возможные выводы. Затем применялась техника разбора случаев, чтобы рассмотреть все представляющие интерес изменения на входе схемы.

В результате были проанализированы и корректно расклассифицированы как схемы с отказами, так и без них.

3.4. Формальная логика

Проблема из области формальной логики, которая привлекла наше внимание, состояла в том, чтобы найти более короткое доказательство одной известной теоремы. Эта теорема утверждает, что некоторая формула исчисления эквивалентности, обозначаемая в [14] XGK, является единственной аксиомой этого исчисления. Как было сказано в разд. 2.4, наиболее удобным правилом вывода является правило отрывания с насыщением. Внимательное изучение этого правила показало, что оно весьма похоже на правило *UR*-резолюции [9] и на правило гиперрезолюции. (В *UR*-резолюции ядро должно быть неединичным дизъюнктом, сателлиты и резольвента — единичными.) Мы нашли доказательство, которое почти вдвое короче, чем опубликованное. Мы достигли этого, назначая программные веса [10, 26] определенным формулам таким образом, что некоторые из них оказывались более предпочтительными, другие же вообще игнорировались [26].

В процессе этого исследования мы обнаружили, что можно задать демодуляторы, которые бы классифицировали формулы в соответствии с различными синтаксическими свойствами. Были найдены также другие демодуляторы, которые подсчитывали число вхождений символа E в посылки и в заключение каждого применения правила вывода. Поскольку попытка упрощения доказательства оказалась успешной и открылись новые нетривиальные возможности применения демодуляции, мы обратились к некоторым открытым проблемам.

Вначале мы рассмотрели формулу, обозначаемую XBB, и попытались изучить формулы, которые получаются из нее применением отрывания с насыщением. Хотя было уже известно, что упомянутая формула слишком слаба и не может служить единственной аксиомой, это исследование оказалось весьма полезным. Программа строила дизъюнкты, которые были все длиннее и длиннее. Немедленно возникло предположение, что это свойство, постоянный рост длины, выполняется для всех формул, которые получаются из XBB повторяющимся применением отрывания с насыщением. Если бы это было так, то мы смогли бы охарактеризовать все теоремы, которые следуют из изучаемой формулы. Оказалось, что это действительно так. Доказательство было получено вручную, но гипотеза непосредственно возникла в результате анализа работы программы автоматического доказательства теорем.

Затем мы применили эту гипотезу к другой формуле, о которой не было известно, может ли она быть единственной аксиомой. Программа обнаружила, что для получающихся из нее формул гипотеза о постоянном увеличении длины неверна. Здесь частично использовались для подсчета вхождений символов упомянутые выше демодуляторы. Разбираться в получающихся дизъюнктах было крайне неудобно из-за их сложности и похожести друг на друга. Поэтому мы придумали демодулятор, с помощью которого дизъюнкты преобразуются в более удобный для чтения вид. Мы решили сосредоточить усилия на формуле, обозначаемой XAK.

При последующей работе программы все нужные дизъюнкты переписывались. Переписывающим правилом служил тот самый демодулятор, с помощью которого достигался удобный для чтения вид. Мы обнаружили удивительный факт: все получившиеся дизъюнкты содержали некоторое фиксированное (с точностью до переименования переменных) подвыражение. Возникло предположение, что все формулы, которые следуют из XAK, обладают этим свойством. Если бы это удалось доказать, то все теоремы, выводимые из XAK при помощи отрывания с насыщением, содержали бы это замечательное подвыражение. Тогда мы были бы уверены, что XAK не подходит на

роль единственной аксиомы исчисления эквивалентности, поскольку имеются теоремы этого исчисления, не содержащие упомянутого подвыражения.

Поскольку неограниченное число применений отрывания с насыщением порождает бесконечное множество формул, мы решили описывать его в терминах схем, а не конкретных формул. Итак, мы использовали программу для исследования всевозможных теоремных схем, выводимых из ХАК. Немедленно стало ясно, что возможные схемы можно описать индукцией вместе с некоторым разбором случаев. Индукция проводилась по длине наиболее общего примера двух унифицируемых выражений. Хотя на самом деле индукция не была программно поддержана, но все же анализ разбиения на случаи был сделан с помощью программы. Итак, наша программа вновь проявила себя как коллега.

Мы приняли некоторый естественный, но, как выяснилось, недостаточно обоснованный критерий разбиения на случаи. Оказалось, что дело обстоит сложнее, чем мы ожидали. Работа программы показала, что постоянно возникают новые случаи, которые необходимо рассмотреть. Анализ результата работы позволил найти разумный критерий разбиения на случаи.

Поскольку с деталями можно легко ознакомиться [26], мы ограничимся кратким резюме. Использование программы позволило: сформировать гипотезы, отвергнуть некоторые из них, выработать систему обозначений, обнаружить правильный критерий разбиения на случаи. Имея такую поддержку, мы вряд ли могли потерпеть неудачу. Оказалось возможным правильно рассортировать оставшиеся семь формул. Пять из них, как выяснилось, не подходят на роль единственной аксиомы, каждая из двух других может служить единственной аксиомой исчисления эквивалентности.

Еще одно удачное применение нашей программы состояло в следующем. Для каждой из двух вышеупомянутых формул доказательство того, что она является единственной аксиомой, было представлено в нестандартной нотации, ориентированной на работу со схемами теорем. Логикам такая форма записи доказательства могла бы показаться неинтересной, поэтому мы «заставили» программу перевести каждое из доказательств в стандартную форму, использующую отрывание с насыщением.

Как упоминалось в разд. 2.4, полученные доказательства являются крайне длинными и сложными. Самое длинное доказательство насчитывает 162 применения отрывания с насыщением. Наиболее сложная формула в этом доказательстве содержит 103 символа, исключая запятые и скобки. Поэтому прямое использование программы для автоматического доказательства, по-видимому, не дало бы результата. Но и попытка

логика атаковать эту проблему «в лоб» тоже, вероятно, окончилась бы неудачей. В целом это исследование является яркой демонстрацией нашей точки зрения, состоящей в том, что программу автоматического доказательства теорем следует рассматривать и использовать как помощника.

4. МИФЫ И ОТКЛИКИ

Существуют два типичных заблуждения по поводу автоматического доказательства; мы будем называть эти заблуждения мифами. Первый из них, так называемый 0/1-миф, более важен. Суть его в следующем: считается, что доказывающая программа должна либо решить все проблемы, либо быть признана бесполезной (этим и объясняется название «0/1-миф»). То есть требуется, например, чтобы работа над данной проблемой оканчивалась либо построением доказательства, либо выдачей определенного отрицательного ответа. Если следовать этому заблуждению, то достаточно было бы взглянуть лишь на последнюю строку и узнать, выведен ли пустой дизъюнкт. Если же он не выведен, то считать весь процесс поиска бесполезным, а его результат выбросить за ненадобностью. И, следовательно, не стоило бы анализировать промежуточные дизъюнкты и выискивать алмазы среди этого пепла.

Согласно второму мифу, специализированные программы предпочтительнее универсальных. Займемся вначале первым.

Такая 0/1-точка зрения, которая часто скрыта, может, к примеру, принять следующий вид. Если программе для автоматического доказательства предложена некоторая известная теорема, и если программа не нашла ее доказательство, то заключают, что эта программа по сути бесполезна. Вот аналог этого рассуждения: если вашего сотрудника попросили помочь в каком-либо исследовании, и если в течение некоторого времени ему ничего не удалось сделать, то следует признать сотрудника негодным. Так, конечно, никто не поступает.

С целью некоторого обобщения и чтобы понять, почему 0/1-точка зрения не приложима к автоматическому доказательству, рассмотрим вначале следующий вопрос: что ожидают от сотрудника и что можно ожидать от программы для автоматического доказательства теорем. Некоторый сотрудник мог бы обладать следующими качествами:

1) Он готов помогать в исследовании практически любой проблемы. К сожалению, во многих случаях ему необходимо сообщить слишком много общеобразовательного материала, чтобы заложить основы знаний, и в этих ситуациях он часто не может сделать ничего существенного.

2) Он продемонстрировал недюжинные исследовательские способности в некоторых областях. Ему удалось, например, ре-

шить несколько открытых проблем. Кроме того, его результаты позволили другим получить решение открытых проблем.

Такое сотрудничество, очевидно, весьма полезно. Но этими же двумя свойствами обладает и программа для автоматического доказательства теорем и поэтому ее следует рассматривать с тех же позиций. Наличие первого качества неудивительно и проявляется в том, что обычно приходится затратить много труда на подготовительную работу. Наличие второго качества значительно интересней. Как видно из предыдущих разделов, программа использовалась для решения большого числа открытых проблем, причем использовалась в качестве помощника, а не просто как логический вычислитель. В своих попытках решить эти проблемы мы видели полный спектр возможных исходов. В формальной логике, например, была показана бессмысленность прямого использования отрывания с насыщением. Но, как и при работе с консультантом, мы нашли другой подход, где также были и успехи, и неудачи. А если бы мы следовали 0/1-заблуждению, то первые же неудачи вынудили бы нас вовсе оставить эту проблему. Итак, полученные нами нетривиальные результаты в области формальной логики свидетельствуют о том, что 0/1-миф не следует применять к программам для доказательства теорем так же, как не применяют его к коллегам.

Консультация с коллегой может иметь три исхода:

1) Он может решить проблему полностью самостоятельно.

2) Он может дать нам ценные указания или обратить внимание на неучтенную нами информацию, что позволит упростить проблему.

3) Он вовсе не сможет помочь.

Мы сейчас обсуждаем третий случай. Причиной неудачи может быть либо неправильное понимание проблемы, либо отсутствие интуиции по этому поводу, либо просто глупость вашего коллеги. В любом случае мы не сбрасываем со счетов такого сотрудника. Более того, мы надеемся, что следующая попытка окажется плодотворней. Другими словами, 0/1-точка зрения не применима. Аналогичным образом мы рассматриваем работу с программой для доказательства теорем. Основной момент состоит в том, чего мы ждем от нее. Если же рассматривать такую программу как помощника в исследовательской работе, то можно ожидать любых результатов: от абсолютного успеха до полной неудачи.

Если представить себе критические замечания в адрес автоматического доказательства, то окажется, что они часто основаны, по крайней мере неявно, на 0/1-заблуждении. Называют различные препятствия, которые делают получение существенных результатов труднодостижимым. Вот далеко не полный их

перечень: сложность доказательства, трудность представления данных в языке первого порядка, невозможность использовать интуитивные соображения и т. д. Убеждения в том, что эти препятствия непреодолимы, основаны на свидетельствах несостоятельности программ для доказательства теорем при решении некоторых конкретных задач. К примеру, невозможность завершить доказательство — типичная черта доказывающих программ. Но такого же рода неудачи часто терпят даже опытные сотрудники. Итак, оценивая полезность, можно придерживаться одной из двух точек зрения:

1) Сотрудник или доказывающая программа считаются полезными, если могут внести существенный вклад в решение проблем из ограниченного, но интересного класса.

2) Некто (или нечто) считается полезным при наличии способности разобраться почти в любой проблеме, если и не с блеском, то, по крайней мере, компетентно.

Вторая из этих точек зрения напоминает все тот же 0/1-миф. Мы, конечно, придерживаемся первой и надеемся, что продемонстрировали ее приложимость к программам для доказательства теорем в целом. Подходящим примером служит успешное опровержение гипотез, касавшихся исчисления эквивалентности. Ясно, что гипотезы эти были весьма разумны, поскольку опровергающие их доказательства нетривиальны и сложны. И если принять во внимание длину и сложность найденных программой доказательств, то трудно ожидать, чтобы кто-либо мог предположить, что такие доказательства вообще существуют.

Итак, все более очевидно, что к программе для автоматического доказательства теорем можно относиться как к коллеге. Работая с такой программой можно, например, выдвигать гипотезы, проверять их, находить пробелы в имеющихся доказательствах — совсем как при сотрудничестве с коллегой.

Обратимся теперь ко второму заблуждению: специализированные программы предпочтительней универсальных. Конечно, это заблуждение опровергается, в основном, самим существованием хороших универсальных программ. Мы укажем на легкость эксплуатации, разнообразие приложений и удобство.

Наше мнение по поводу первого из этих трех качеств было бы неоправданным, если бы не тот факт, что для работы в новых областях не понадобилось включать в программу новые модули. Все проблемы, о которых шла речь выше, были решены без какого-либо дополнительного программирования. Мы обнаружили, что имеющиеся в наличии стандартные процедуры могут быть адаптированы к возникающим задачам. Это не означает, что не понадобятся новые правила вывода. Введение такого нового правила могло бы потребовать дополнительного программирования и, возможно, в большом объеме. Такая возмож-

ность — одна из многих причин для разработки новой программы для автоматического доказательства теорем, о которой кратко будет сказано в разд. 5.

Что касается второго качества, разнообразия приложений, то решенные с помощью доказывающей программы проблемы варьируют от проектирования схем до формальной логики, возникающие при этом задачи — от нахождения моделей, более интересных, чем существующие, до построения очень длинных и сложных доказательств. В области проектирования схем речь шла об эффективности, обнаружении отказов и проверке их отсутствия. В области абстрактной математики мы обсуждали проблемы нахождения полугрупп, обладающих требуемыми свойствами. В этой связи рассматривалась проблема минимальности, но не с помощью перебора массы неподходящих полугрупп. Наконец, в области формальной логики исследовались вопросы существования новых аксиом. Это исследование потребовало гибкого подхода, разбора случаев, индукции и такого представления вывода, которое позволило не рассматривать миллионы не имеющих отношения к делу формул.

Что же касается удобства, то мы утверждаем, что здесь еще многое предстоит сделать. Одна из многих трудностей — необходимость хорошо знать входной язык программы. Научиться ясно выражаться на языке дизъюнктов — обременительное занятие. Работа с нашей системой еще более усложняется наличием огромного множества альтернатив и возможных значений параметров. Правда, эта вторая трудность несколько сглажена появлением транслятора [7], написанного Э. Ласком, который позволяет управлять доказывающей программой при помощи легко воспринимаемых человеком инструкций.

Анализ описанной выше деятельности заставляет сомневаться в правильности выбора в пользу специализированных программ. По крайней мере, написать специализированную программу для решения рассмотренных нами задач из области формальной логики, видимо, очень трудно; пришлось бы затратить громадные усилия для реализации описанной выше техники. Да и прямая атака этих открытых проблем, вероятно, оказалась бы невозможной: слишком много формул пришлось бы изучать, если бы вывод строился только при помощи отрывания с насыщением.

5. НЫНЕШНЯЯ ДЕЯТЕЛЬНОСТЬ

Естественно спросить: чем мы занимаемся сейчас? Мы ищем новые нерешенные проблемы. Хотелось бы найти и такие, которые требуют построения моделей и контрпримеров, и такие, которые требуют поиска доказательств. Мы изучаем возможность продолжения исследований в области проектирования

схем с тем, чтобы найти промышленное применение результатов. Мы намерены в конечном итоге иметь большую группу пользователей — специалистов в различных областях с широким спектром интересов.

Последнее приложение программы связано с проектированием ядерных реакторов. Здесь существенно использовался аппарат разбора случаев. У нас есть почти законченная небольшая интерактивная доказывающя программа. Мы продемонстрировали ее использование для анализа описаний реактора.

Для того, чтобы облегчить широкому кругу лиц использование доказывающих программ, мы сейчас сосредоточили усилия на разработке и реализации новой программы для доказательства теорем. Эта программа пишется на языке ПАСКАЛЬ и будет весьма гибкой и портативной. Например, не надо будет заботиться о представлении данных в виде дизъюнктов; можно будет быстро перейти к новым экспериментам как типа «чистого» поиска доказательства, так и прикладного характера. Т. е. пользователи новой системы смогут приспосабливать эту программу и ее возможности к своим нуждам.

Цель состоит в том, чтобы новая доказывающая программа стала доступной многим. Язык ПАСКАЛЬ позволит анализировать и видоизменять ее структуру и свойства. Короче говоря, эта программа сможет служить хорошим инструментом исследования во многих областях.

6. ЗАКЛЮЧЕНИЕ

Автоматическое доказательство теорем способно сейчас внести реальный вклад в исследования в различных областях. С использованием доказывающей программы были решены открытые проблемы в математике и в формальной логике. Но этим не исчерпываются возможные приложения. Есть реальные свидетельства практического использования таких программ, и успех в области проектирования схем и обнаружения отказов — один из примеров тому.

Следует иметь в виду два крайне важных момента. Во-первых, результаты, о которых здесь шла речь, могут быть получены многими доказывающими программами. Техника и методология, которые привели к успеху, присущи области машинного доказательства в целом. Во-вторых, все представленные нами результаты были получены с помощью уже существовавшей программы для доказательства теорем без каких-либо ее модификаций. Помня об этих двух обстоятельствах, разумно предположить, что автоматическое доказательство теорем вступило в новую фазу развития, для которой характерен синтез осмысления возможных целей с реализацией некоторых из них.

Конечно, предстоит решить еще много проблем. Например, очень мало известно о связи между представлением исходной информации, выбором правил вывода и использованием стратегий. Тем не менее, сейчас к доказывающей программе часто можно относиться как к коллеге. Коллега может помочь выработать, проверить, доказать или опровергнуть гипотезу. В этом же может помочь и программа. Хотя программа часто терпит полную неудачу, иногда она получает впечатляющие результаты. Даже многие из тех попыток, которые окончились неудачно из-за нехватки времени и/или памяти, содержат весьма ценную информацию. Взаимодействие программы и пользователя в процессе исследования оказывается выгодным.

Нужны новые открытые проблемы. Попытки их решения ведут к выявлению недостатков существующих систем. Решение каждой такой проблемы иногда ведет к постановке новых открытых проблем. Этот углубляющий диалог укажет (и уже частично указал) истинную роль программы для автоматического доказательства теорем — роль рассуждающего помощника.

ПРИЛОЖЕНИЕ

Мы продемонстрируем применение программы для автоматического доказательства теорем в трех различных областях: в тернарной булевой алгебре [21, 23], теории конечных полугрупп [22], проектировании логических схем [20]. Вначале обсудим некоторые общие для этих областей аспекты. Во-первых, ни в одном из приложений мы не искали доказательство методом от противного, что обычно ассоциируется с использованием доказывающих программ. Вместо этого мы искали модели различных типов: математические модели и контрпримеры или, в случае проектирования схем, ситуации, в которых физическая система может функционировать не так, как требуется. Во-вторых, дизъюнкты, которые порождались программой при построении моделей, совершенно не отличаются по виду от тех, которые порождаются при поиске доказательства. Подчеркнем, что для построения нужных моделей не потребовалось никакого перепрограммирования, эта деятельность программы не противопоставлялась поиску доказательства. Для обеих целей использовались одни и те же правила вывода и значения программных параметров. В частности, протоколы работы программы в обоих случаях настолько похожи, что, глядя на значения программных параметров, их невозможно различить.

Займемся сейчас математическими исследованиями в области тернарной булевой алгебры и конечных полугрупп. В каждом из этих исследований при помощи построения соответствующей модели была решена открытая математическая

проблема [21, 22]. Естественно, до начала работы эти модели были нам неизвестны. Поэтому мы осуществляли итеративный процесс поиска, активно используя доказывающую программу. Каждый представленный ниже протокол работы соответствует завершающему этапу этого процесса.

Мы проиллюстрируем способы задания доказывающей программе той или иной модели. Первый способ состоит в том, что действие каждой функции на элементах модели задается явно, с помощью таблицы. Этот метод применялся для изучения тернарной булевой алгебры. Второй метод описывает действия фукций неявно, используя образующие и соотношения. Он применялся для исследования конечных полугрупп.

ТЕРНАРНАЯ БУЛЕВА АЛГЕБРА

Мы приведем результаты двух процессов поиска вывода. Первый из них устанавливает непротиворечивость одной из моделей. Второй показывает, что на некоторой другой модели одна из аксиом не выполняется. Входные данные для обоих процессов следующие.

* Каждая строка, начинающаяся символом *, является комментарием.
* Значения программных параметров опущены.
* Следующие входные дизъюнкты перечисляют элементы
* строящейся модели;
1) $Q(a)$;
2) $Q(g(a))$; * Вместо $g(a)$ в разд. 3.1 написано b;
3) $Q(g(g(a)))$; * Вместо $g(g(a))$ в разд. 3.1 написано c;
* Следующие дизъюнкты утверждают, что эти три элемента
* различны;
4) $\neg$ EQUAL$(g(x),x)$;
5) $\neg$ EQUAL$(g(g(x)),a)$;
* Следующий демодулятор определяет действие функции g на
* этих трех элементах. Поскольку значение g на a есть $g(a)$
* и значение g на $g(a)$ есть $g(g(a))$, остается определить
* лишь действие g на $g(g(a))$;
6) EQUAL$(g(g(g(a))),g(a))$;
* Следующие демодуляторы определяют действие трехмест-
* ной функции f на этих трех элементах;
7) EQUAL$(f(x, y, g(y)),x)$;
8) EQUAL$(f(x,x,y),x)$;
9) EQUAL$(f(g(x),x,y),y)$;
10) EQUAL$(f(x,y,x),x)$;
11) EQUAL$(f(x,g(g(y)),g(y)),x)$;
12) EQUAL$(f(g(y),g(g(y)),x),x)$;

13) EQUAL($f(a,g(g(a)),g(g(a)))$,a);
14) EQUAL($f(g(g(a)),a,a)$,$g(g(a))$);
15) EQUAL($f(a,g(a),g(a))$,$g(a)$);
* Следующий дизъюнкт поглощает и «вычеркивает» все выводимые тривиальные равенства;
16) EQUAL(x,x);
* Все остальные дизъюнкты неединичные. Каждый из них играет роль ядерного в применении правила гиперрезолюции;
* Следующее ядро служит для проверки того, что данное множество элементов замкнуто относительно функции g;
17) $\neg Q(x)Q(g(x))$;
* Следующее ядро служит для проверки того, что данное множество элементов замкнуто относительно функции f;
18) $\neg Q(x)\neg Q(y)\neg Q(z)Q(f(x,y,z))$;
* Следующее ядро служит для проверки того, что аксиома 1 тернарной булевой алгебры (см. разд. 2.1) выполняется для всех упорядоченных наборов, которые можно образовать из этих трех элементов;
19) $\neg Q(v)\neg G(w)\neg Q(x)\neg Q(y)\neg Q(z)$
EQUAL($f(f(v,w,x),y,f(v,w,z))$,$f(v,w,f(x,y,z))$);
* Следующие три ядра служат для проверки того, что аксиомы 3, 4 и 5 тернарной булевой алгебры выполняются для всех упорядоченных пар, образованных из данных трех элементов;
20) $\neg Q(x)\neg Q(y)$EQUAL($f(x,y,g(y))$,x);
21) $\neg Q(x)\neg Q(y)$EQUAL($f(x,x,y)$,x);
22) $\neg Q(x)\neg Q(y)$EQUAL($f(g(x),x,y)$,y);

В процессе работы программы к множеству перечисленных дизъюнктов не было добавлено ни одного нового. Каждый выведенный дизъюнкт после демодуляции поглощался и вычеркивался. Тот факт, что не возник ни один новый дизъюнкт, содержащий литеру Q, означает, что исходное множество демодуляторов фактически дает полную таблицу для функций f и g. Тот факт, что не добавилось ни одного нового равенства, означает, что в построенной модели выполняются аксиомы 1, 3, 4 и 5. Аксиома 2 опровергается наличием входного дизъюнкта

13) EQUAL($f(a,g(g(a)),g(g(a)))$,a);

Следовательно, аксиома 2 независима от остальных четырех.

Второй процесс поиска вывода для тернарной булевой алгебры является небольшой модификацией первого. Входной дизъюнкт 15) был заменен новым входным дизъюнктом

15) EQUAL($f(a,g(a),g(a))$,$g(g(a))$);

То есть значение функции f изменилось в единственной точке. В этом случае результат работы программы был следующим.

23) 19, 1, 2, 2, 1, 1 EQUAL($g(g(a))$,a);

Демодуляторы: 15, 10, 14, 9, 10
24) 23, 5 противоречие;
Это означает, что наши аксиомы выполняются не при всех значениях переменных. Точнее, при некоторых значениях переменных из аксиомы 1 следует, что $g(g(a)) = a$, вопреки входному условию, требующему, чтобы a и $g(g(a))$ были различны. Читатель легко заметит, что этот второй процесс можно рассматривать либо как неудавшийся процесс построения модели, либо как удавшееся доказательство от противного того факта, что $g(g(a)) = a$ следует из остальных условий.

КОНЕЧНЫЕ ПОЛУГРУППЫ

Обратимся теперь к другому математическому исследованию, связанному с конечными полугруппами. Мы приведем результаты двух процессов работы доказывающей программы. Один процесс, используя представление полугруппы образующими и соотношениями, пополняет множество соотношений, причем в эту полугруппу «встраивается» определенный антиавтоморфизм. Другой процесс показывает, что некоторое заданное на образующих отображение не может быть продолжено до инволюции всей полугруппы.

Первый процесс поиска завершает серию экспериментов по построению полугруппы с требуемой комбинацией свойств. Исходя из данного минимального множества соотношений, он порождает совокупность демодуляторов, достаточную для построения (если это необходимо) списка элементов и таблицы умножения. Используются два правила вывода: парамодуляция — для проверки соответствия между аксиомами и имеющимися соотношениями и гиперрезолюция с определенным ядром — для того, чтобы «наложить» требуемый антиавтоморфизм на эти соотношения. Входными дизъюнктами были следующие.

* Следующий дизъюнкт поглощает все выводимые тривиальные равенства;
1) EQUAL(x,x);
* Следующий дизъюнкт гарантирует ассоциативность операции произведения в данной полугруппе;
2) EQUAL$(f(f(x,y),z),f(x,f(y,z)))$;
* Следующие три дизъюнкта приравнивают все произведения четырех и более элементов данной полугруппы единице;
3) EQUAL$(f(x,f(y,f(z,w))),1)$;
4) EQUAL$(f(1,x),1)$;
5) EQUAL$(f(x,1),1)$;
* Следующее соотношение введено для того, чтобы устранить определенную симметрию в данной полугруппе. Устранение симметрии может исключить возможные инволюции;

6) EQUAL($f(d,f(d,e)),f(b,f(b,c))$);
* Следующее ядро «применяет» антиавтоморфизм h ко всем
* соотношениям;
7)]EQUAL($f(x,y),z$) EQUAL($f(h(y),h(x)),h(z)$);
* Следующие демодуляторы вычисляют образ произвольного
* терма при антиавтоморфизме h;
8) EQUAL($h(f(x,y)),f(h(y),h(x))$);
9) EQUAL($h(b),c$);
10) EQUAL($h(c),d$);
11) EQUAL($h(d),e$);
12) EQUAL($h(e),b$);
13) EQUAL($h(1),1$);

В результате работы доказывающей программы было добавлено единственное новое равенство:

14) 7,6EQUAL($f(d,f(c,c)),f(b,f(e,e))$);

демодуляторы: 8, 12, 11, 11, 2, 8, 8, 10, 9, 9, 2;

Кроме того, было напечатано сообщение, что никаких других следствий с помощью имеющихся правил вывода получить нельзя. Это означает, что требуемое множество демодуляторов порождено полностью. С помощью этих демодуляторов можно построить список всех различных элементов и найти все произведения этих элементов (более детально см. [23]).

Второй процесс показывает, что данная полугруппа не допускает инволюцию, которая переставляет b с d и c с e. Мы имеем следующие входные дизъюнкты.

* Поглощение тривиального равенства;
1) EQUAL(x, x);
* Ассоциативность;
2) EQUAL($f(f(x,y),z),f(x,f(y,z))$);
* Демодуляторы, определяющие произведение в данной полу-
* группе. Последний из них был получен в результате приве-
* денного выше процесса;
3) EQUAL($f(x,f(y,f(z,w))),1$);
4) EQUAL($f(1,x),1$);
5) EQUAL($f(x,1),1$);
6) EQUAL($f(d,f(d,e)),f(b,f(b,c))$);
7) EQUAL($f(d,f(c,c)),f(b,f(e,e))$);
* Следующее ядро «применяет» возможную инволюцию j ко
* всем приведенным выше равенствам;
8)]EQUAL($f(x,y),z$) EQUAL($f(j(y),j(x)),j(z)$);
* Следующие демодуляторы определяют конкретную инволю-
* цию j, задавая это отображение на образующих и правило
* продолжения его на остальные элементы;
9) EQUAL($j(f(x,y)),f(j(y),j(x))$);
10) EQUAL($j(b),d$);
11) EQUAL($j(d),b$);

12) EQUAL($j(c)$,e);
13) EQUAL($j(e)$,c);
14) EQUAL($j(1)$,1);

В результате работы программы имеем два новых равенства:

15) 8, 6 EQUAL($f(e,f(d,d)),f(c,f(b,b))$);
демодуляторы: 9, 13, 11, 11, 2, 9, 9, 12, 10, 10, 2;
16) 8, 7 EQUAL($f(e,f(e,b)),f(c,f(c,d))$);
демодуляторы: 9, 12, 12, 11, 2, 9, 9, 13, 13, 10, 2;

Наличие этих равенств показывает, что заданное на образующих отображение j нельзя продолжить до инволюции всей полугруппы, поскольку тогда должны были быть верны упомянутые равенства, что невозможно в рассматриваемой полугруппе[1]). Итак, эта полугруппа определенную выше инволюцию не допускает. Аналогично можно убедиться в отсутствии других возможных инволюций, задавая их действие на образующих.

ПРОЕКТИРОВАНИЕ ЛОГИЧЕСКИХ СХЕМ

Остановимся теперь на вопросах проектирования логических схем. Наш пример [20] касается обнаружения отказов в бинарной логической схеме. Термин «отказ» означает некоторое неудовлетворительное поведение схемы в процессе изменения входных сигналов. Для порождения зависящих от времени входных сигналов и моделирования поведения схемы используются соответствующие ядра и демодуляторы.

Мы используем классификацию возможных видов сигналов, данную Фантоцци в [2]:

t и f обозначают установившиеся неизменные сигналы;
tf обозначает плавный переход из состояния t в состояние f;
$t*t$ обозначает сигнал, который почти установился, но иногда случайно переходит в f, а затем возвращается в t;
$t*f$ обозначает переход из состояния t в состояние f, который не является плавным, т. е. состояние сигнала колеблется между t и f, прежде чем установится в f;
ft, $f*f$, $f*t$ определяются аналогично.

Мы считаем, что в качестве входных допустимы лишь сигналы t, f, tf или ft. Однако на выходе могут появиться и остальные четыре сигнала, т. е. допускается возможность отказа. Доказывающей программе были предложены следующие данные.

* Правила комбинирования сигналов посредством стандарт-
* ных логических элементов задаются в [2] таблицами. Мы
* кодируем эти таблицы.

---

[1]) Рассматриваемая полугруппа имеет порядок 83. Наличие двух новых соотношений заставляет «склеить» две пары элементов и получить полугруппу порядка 81. — *Прим. перев.*

and	f	ft	f^*f	f^*t	t	tf	t^*t	t^*f
f	f	f	f	f	f	f	f	f
ft	f	ft	f^*f	f^*t	ft	f^*f	f^*t	f^*f
f^*f	f	f^*f	f^*f	f^*f	f^*f	f^*f	f^*f	f^*f
f^*t	f	f^*t	f^*f	f^*t	f^*t	f^*f	f^*t	f^*f
t	f	ft	f^*f	f^*t	t	tf	t^*t	t^*f
tf	f	f^*f	f^*f	f^*f	tf	tf	t^*f	t^*f
t^*t	f	f^*t	f^*f	f^*t	t^*t	t^*f	t^*t	f^*f
t^*f	f	f^*f	f^*f	f^*f	t^*f	t^*f	t^*f	t^*f

;

1) EQUAL(and(f,x),f);
2) EQUAL(and(x,f),f);
3) EQUAL(and(t,x),x);
4) EQUAL(and(x,t),x);
5) EQUAL(and(ft,ft),ft);
6) EQUAL(and(ft,f^*f),f^*f);
7) EQUAL(and(ft, f^*t),f^*t);
8) EQUAL(and(ft, tf), f^*f);

* (остальные полученные из той же таблицы дизъюнкты опущены ввиду недостатка места);

not	f	ft	f^*f	f^*t	t	tf	t^*t	t^*f
	t	tf	t^*t	t^*f	f	ft	f^*f	f^*t

41) EQUAL(not(f),t);
42) EQUAL(not(ft),tf);

* (и так далее);

* Определим логическую функцию «nand» одного, двух и трех * переменных;

49) EQUAL(nand1(x),not(x));
50) EQUAL(nand2(x,y),not(and(x,y)));
51) EQUAL(nand3(x,y,z),not(and(x,and(y,z))));

* Введем демодуляторы, которые вычисляют выходной сигнал * данной схемы по заданному множеству зависящих от времени входных сигналов. Предположим, что имеются три * входных канала, обозначенные a, b, c. Пусть используются * nand-вентили, действие которых описывается функциями * nand$g$1, nand$g$2, nand$g$3.

* Функция «ipt» кодирует входные сигналы для каналов * a, b, c.

* Например, ipt(t,f,ft) означает, что по каналу a поступает * сигнал t, по каналу b—f, а по каналу c—ft (переход от состояния f к состоянию t при отсутствии отказа).

* Функция «eval» вычисляет выходной сигнал данной схемы

* (первый аргумент) по данному входу (второй аргумент);
52) EQUAL(eval(nand*g*1(*x*),vipt),nand1(eval(*x*,vipt)));
53) EQUAL(eval(nand*g*2(*x*,*y*),vipt),nand2(eval(*x*,vipt),
eval(*y*,vipt)));
54) EQUAL(eval(nand*g*3(*x*,*y*,*z*),vipt),nand3(eval(*x*,vipt),
eval(*y*,vipt),eval(*z*,vipt)));
55) EQUAL(eval(*a*,ipt(*xa*,*xb*,*xc*)),*xa*);
56) EQUAL(eval(*b*,ipt(*xa*,*xb*,*xc*)),*xb*);
57) EQUAL(eval(*c*,ipt(*xa*,*xb*,*xc*)),*xc*);
* Проверяем все входные сигналы, не вызывающие отказов.
* Вначале задаем таблицу, описывающую схему при фиксированных входах, а затем таблицу, описывающую одношаговые переходы.
* Результаты одношаговых переходов проверяются на отсутствие отказов;
* Список возможных постоянных входных сигналов;
58) INPUT(*f*,fixed);
59) INPUT(*t*,fixed);
* Список всех возможных входных сигналов, которые соответствуют переходам, не вызывающим отказов;
60) INPUT(*ft*,transition);
61) INPUT(*tf*,transition);
* Ядро, с помощью которого порождается список выходов
* при фиксированных входных сигналах;
62)] CIRCUIT(*w*)] INPUT(*x*,fixed)] INPUT(*y*,fixed)
] INPUT(*z*,fixed)
IO(input,*a*,*x*,*b*,*y*,*c*,*z*,output,eval(*w*,ipt(*x*,*y*,*z*)));
* Ядра, порождающие список выходов, когда два входных
* сигнала фиксированные, а один — переходный;
63)] CIRCUT(*w*)] INPUT(*x*,transition)] INPUT(*y*,fixed)
] INPUT(*z*,fixed)
IO(input,*a*,*x*,*b*,*y*,*c*,*z*,output,eval(*w*,ipt(*x*,*y*,*z*)));
64)] CIRCUIT(*w*)] INPUT(*x*,fixed)] INPUT(*y*,transition)
] INPUT(*z*,fixed)
IO(input,*a*,*x*,*b*,*y*,*c*,*z*,output,eval(*w*,ipt(*x*,*y*,*z*)));
65)] CIRCUIT(*w*)] INPUT(*x*,fixed)] INPUT(*y*,fixed)
] INPUT(*z*,transition)
IO(input,*a*,*x*,*b*,*y*,*c*,*z*,output,eval(*w*,ipt(*x*,*y*,*z*)));
* Проверяемая схема;
66) CIRCUIT(nand*g*2(nand*g*2(*a*,*b*),nand*g*2(nand*g*1(*a*),*c*)));

Среди прочих, доказывающей программой был выведен следующий дизъюнкт:

78) 63, 66, 60, 59, 59, IO (input,*a*,*ft*,*b*,*t*,*c*,*t*,output,*t***t*);

Это утверждение говорит о том, что если по каналам *a*, *b*, *c* поступают входные сигналы *ft*, *t* и *t* соответственно, то на выходе появляется сигнал *t***t*, т. е. возникает нежелательная си-

туация, в которой возможен отказ. Так же можно проверять другие схемы и искать среди них не содержащие отказов.

ЛИТЕРАТУРА

[1] Allen, J. and Luckham, D., «An interactive theorem-proving program», Machine Intelligence, Vol. 5 (1970), Meltzer and Michie (eds), N. Y., pp. 321—336.

[2] Fantauzzi, G., «An algebraic mode for the analysis of logic circuits», IEEE Transactions on Computers, Vol. C-23, № 6, June 1974, pp. 576—581.

[3] Kalman, J., Notre Dame J. of Formal Logic, Vol. 19, № 1, 1978, pp. 141—144.

[4] Kalman, J., private communication.

[5] Lukasiewicz, J., «Der Aquivalenzenkalkul», Collectanea Logica, Vol. 1 (1939), pp. 145—169. English translation in (McCall), pp. 88—115.

[6] Lukasiewicz, J., Jan Lukasiewicz: Selected Works, ed. by L. Borkowski, North-Holland Publishing Co., Amsterdam (1970).

[7] Lusk, E., «Input translator for the environmental theorem prover — user's guide», to be publisched as an Argonne National Laboratory techn. rep.

[8] McCall, S., Polish Logic, 1920—1939, Clarendon Press, Oxford (1967).

[9] McCharen, J., Overbeek, R. and Wos, L., «Problems and experiments for and with automated theorem proving programs», IEEE Transactions on Computers, Vol. C-25 (1976), pp. 773—782.

[10] McCharen, J., Overbeek, R. and Wos, L., «Complexity and related enhancements for automated theorem-proving», Computers and Mathematics with Applications, Vol. 2 (1976), pp. 1—16.

[11] Meridith, C., «Single axioms the systems (C, N), (C, O) and (A, N) of the twovalued propositional calculus», The Journal of Computing Systems, *i*, № 3 (July 1953), pp. 155—164.

[12] Overbeek, R., «An implementation of hyper-resolution», Computers and Mathematics with Applications, Vol. 1 (1975), pp. 201—214.

[13] Peterson, J., Notre Dame J. of Formal Logic. Vol. 17 (1976), pp. 267—271.

[14] Peterson, J., Auckland Univ. Dep. of Math. Rep. Series № 105, 1977.

[15] Peterson, J., Notre Dame J. of Formal Logic, Vol. XIX, 1978, pp. 119—122.

[16] Peterson, J., Auckland Univ. Dep. of Math. Rep. Series № 78.

[17] Prior, A. N., Formal Logic, Second Edition, Ofxord, 1962, Clarendon Press.

[18] Robinson, J., «Automatic deduction with hyper-resolution», International Journal of Computer Mathematics, Vol. 1 (1965), pp. 227—234.

[19] Smith, B., «Reference manual for the environmental theorem prover», to be published as an Argonne National Laboratory technical report.

[20] Winker, S., Private communicatin.

[21] Winker, S. and Wos, L., Proc. of the Eighth Int. Symp. on Multiple-valued Logic. Rosemont, Illinois, 1978, IEEE and ACM Publ., pp. 251—256.

[22] Winker, S., Wos, L. and Lusk, E., Mathematics of Computation, Vol. 37 (1981), pp. 533—545.

[23] Winker, S., «Generation and verification of finite models and cointerexamples using an automated theorem prover answering two open questions», to appear in J. ACM.

[24] Wojciechowski, W. and Wojcik, A., Proc. of the Ninth Int. Symp. on Multiplevalued Logic, Bath, England, 1979.

[25] Wos, L., Robinson, G., Carson, D. and Shalla, L., «The concept of demodulation in theorem proving», J. ACM, Vol. 14 (1967), pp. 698—704.

[26] Wos, L., Winker, S., Veroff, R., Smith, B. and Henschen, L., «Questions concerning possible shortest single axioms in equivalential calculus: an application of automated theorem proving to infinite domains», in preparation.

СОДЕРЖАНИЕ

КИБЕРНЕТИЧЕСКИЙ СБОРНИК

Новая серия

Выпуск
21

Научный редактор С. В. Чудов
Мл. научный ред. Н. С. Полякова
Художник Н. К. Сапожников
Художественный редактор В. И. Шаповалов
Технический редактор Н. И. Борисова
Корректор Н. А. Гиря

ИБ № 5119

Сдано в набор 13.03.84 Подписано к печати 05.10.84. Формат 60×90¹/₁₆. Бумага типографская № 2. Гарнитура литературная. Печать высокая. Объем 8,25 бум. л. Усл. печ. л. 16,50. Усл. кр.-отт. 16,50. Уч.-изд. л. 16,26 Изд. № 1/3606. Тираж 3400 экз. Зак. 1811. Цена 2 р. 50 к.

Издательство «Мир» Москва, 1-й Рижский пер., 2. Отпечатано с матриц Ленинградской типографии № 2 головного предприятия ордена Трудового Красного Знамени Ленинградского объединения «Техническая книга» им. Евгении Соколовой Союзполиграфпрома при Государственном комитете СССР по делам издательств, полиграфии и книжной торговли. 198052, г. Ленинград, Л-52, Измайловский проспект, 29, в Ленинградской типографии № 4 ордена Трудового Красного Знамени Ленинградского объединения «Техническая книга» им. Евгении Соколовой Союзполиграфпрома при Государственном комитете СССР по делам издательств, полиграфии и книжной торговли. 191126, Ленинград, Социалистическая ул., 14.