Московский Государственный Университет
имени М.В. Ломоносова
Российская Академия Наук
Международная Академия Технологических Наук
Российская Академия Естественных Наук

Интеллектуальные Системы.

Теория и приложения

ТОМ **23** ВЫПУСК **4 * 2019**

МОСКВА

ISSN 2411–4448

УДК 519.95; 007:159.955
ББК 32.81

Издается с 1996 г.*

В журнале «Интеллектуальные системы. Теория и приложения» публикуются научные достижения в области теории и приложений интеллектуальных систем, новых информационных технологий и компьютерных наук.

Издание журнала осуществляется под эгидой МГУ имени М. В. Ломоносова, Научного Совета по комплексной проблеме «Кибернетика» РАН, Отделения «Математическое моделирование технологических процессов» МАТН, Секции «Информатики и кибернетики» РАЕН.

Учредитель журнала: ООО «Интеллектуальные системы».

Журнал входит в список изданий, включенных ВАК РФ в реестр публикаций материалов по кандидатским и докторским диссертациям по математике и механике.

Спонсором издания является:

ООО «Два Облака»
Разработка корпоративных информационных систем
http://www.dvaoblaka.ru

Индекс подписки на журнал: 64559 в каталоге НТИ «Роспечать».

Адрес редакции: 119991, Москва, ГСП-1, Ленинские Горы, д. 1, механико-математический факультет, комн. 12-01.
Адрес издателя: 115230, Россия, Москва, Хлебозаводский проезд, д. 7, стр. 9, офис 9. Тел. +7 (495) 939-46-37, e-mail: mail@intsysjournal.org

*) Прежнее название журнала: «Интеллектуальные системы».

ОГЛАВЛЕНИЕ

Часть 1.
Общие проблемы теории интеллектуальных систем

Построение полного набора тем вероятностных тематических моделей

Сухарева А.В., Воронцов К.В.

Интерпретируемость, линейное увеличение сложности с ростом данных, масштабируемость сделали тематическое моделирование одним из наиболее популярных инструментов статистического анализа текстов. Однако есть и ряд недостатков, вызванных зависимостью решения от инициализации. Известно, что построение тематической модели сводится к решению некорректно поставленной задачи неотрицательного матричного разложения. Множество её решений в общем случае бесконечно. Всякий раз модель находит локальный экстремум. Многократное обучение модели по одной и той же коллекции может приводить к обнаружению всё новых и новых тем. На практике часто требуется определить все темы корпуса. Для решения этой задачи в статье предложен и исследован новый алгоритм нахождения полного набора тем, который основан на построении выпуклой оболочки. Экспериментально показано, что за конечное число моделей можно построить базис тем. Правдоподобие базиса тем выше, чем одной модели с аналогичным числом тем. Сравнение базисов моделей LDA (latent Dirichlet allocation) и ARTM (additive regularization for topic modeling) позволяет сделать вывод, что темы наборов совпадают с высокой точностью.

Ключевые слова: вероятностное тематическое моделирование, устойчивость тематических моделей, полный набор тем тематических моделей, латентное размещение Дирихле, LDA, регуляризация, ARTM, BigARTM.

1. Введение

Вероятностные тематические модели являются статистическими алгоритмами, предназначенными для обнаружения абстрактных тем, которые содержатся в большом и неструктурированном наборе документов. Тематические модели наиболее известны как инструмент для интеллек-

туального анализа текста. Они также имеют приложения в биоинформатике, анализе изображений и социальных сетей [1].

Построение тематической модели сводится к решению некорректно поставленной задачи неотрицательного матричного разложения. Множество её решений в общем случае бесконечно. Это приводит к неустойчивости вычислительных методов и зависимости решения от случайного начального приближения. Многократное построение модели по одной и той же коллекции может приводить к нахождению всё новых и новых тем. Несмотря на важность требования устойчивости в задачах компьютерной лингвистики и информационного поиска, проблема до сих пор относительно мало изучена. В литературе определено понятие устойчивости [2], предлагаются меры устойчивости [3, 4, 5]. Также в работах рассматриваются модели, повышающие устойчивость [6, 7].

В данной статье ставится проблема полноты: можно ли найти все темы корпуса, сколько запусков моделирования необходимо для нахождения полного набора тем, возможно ли сократить число запусков с помощью регуляризации. Описанная постановка в литературе не рассматривалась. Актуальность проблемы обусловлена большим числом прикладных задач анализа текстов, в которых требуется как можно полнее определить тематический состав коллекции документов.

Статья организована следующим образом. В разделе 2 мы вводим основные обозначения и терминологию, кратко описываем тематические модели pLSA, LDA и модели подхода ARTM. В разделе 3 определяется понятие базиса тем множества матриц тематических моделей. Далее, предлагается алгоритм для построения полного набора тем на основе выпуклой оболочки. Эмпирические результаты построения полного набора тем обсуждаются в разделе 4. Наконец, в заключении представлены наши выводы.

2. Вероятностные тематические модели

Введем следующие обозначения: $D = \{d_1, \ldots, d_{|D|}\}$ — коллекция текстовых документов, $W = \{w_1, \ldots, w_{|W|}\}$ — словарь термов, встретившихся в них, $T = \{t_1, \ldots, t_{|T|}\}$ — конечное множество тем. В качестве термов могут использоваться слова, n-граммы, коллокации, именованные сущности и т.д. Число тем является гиперпараметром и задается заранее ($|T| \ll |D|$). Каждое вхождение терма $w \in W$ в документ $d \in D$ связано с некоторой темой $t \in T$. Термы и документы являются наблюдаемыми переменными, темы — латентными (скрытыми). Требуется

определить, к каким темам относится каждый документ и какие термы образуют каждую тему.

2.1. Модель pLSA

Пусть коллекция документов представляет собой последовательность пар «документ-слово» (d, w). Подход pLSA (probabilistic latent semantic analysis) [8] моделирует вероятность $p(w|d)$ появления термов w в документах d как смесь условно независимых мультиномиальных распределений. Компоненты смеси можно рассматривать как представления «тем» t. Наблюдаемые пары (d, w) встречаются независимо, что соответствует представлению документов в виде «мешка слов». Появление терма w зависит только от темы t и не зависит от документа d. Каждое слово генерируется из одной темы. В этом случае тематическая модель появления слов в документах выглядит следующим образом:

$$p(d, w) = p(d)p(w|d) = p(d)\sum_{t\in T} p(w|t)p(t|d),$$

где w — терм, t — тема, d — документ коллекции.

При построении тематической модели требуется оценить по известной коллекции D параметры модели $\varphi_{wt} = p(w|t)$ и $\theta_{td} = p(t|d)$. Ставится задача максимизации логарифма правдоподобия:

$$L(\Phi, \Theta) = \sum_{d\in D}\sum_{w\in W} n_{dw} \log p(d, w) = \sum_{d\in D}\sum_{w\in W} n_{dw} \log \sum_{t\in T} \phi_{wt}\theta_{td} \to \max_{\Phi,\Theta} \tag{1}$$

$$\sum_{w\in W} \phi_{wt} = 1, \phi_{wt} \geq 0, \quad \sum_{t\in T} \theta_{td} = 1, \theta_{td} \geq 0,$$

где $\Phi = (\phi_{wt})_{W\times T}$, $\Theta = (\theta_{td})_{T\times D}$, n_{dw} — число вхождений терма w в документ d.

Задача решается с помощью EM-алгоритма. Вероятностные тематические модели находят локальный максимум логарифма правдоподобия (1). Известно, что pLSA не моделирует процесс выбора документов, распределение $p(d)$, это частично генеративная модель. Чтобы полностью реализовать генеративный процесс на уровне документа и усовершенствовать его для тем, было предложено скрытое распределение Дирихле (latent Dirichlet allocation, LDA) [9].

Следует отметить, что модель pLSA легко переобучается из-за необходимости настраивать большое число параметров. Решением данной проблемы может стать небольшая модификация E-шага

EM-алгоритма (Tempered EM) [8], которая также позволяет ускорить обучение модели.

Другая трудность заключается в том, что алгоритм не разделяет тематические и фоновые компоненты смеси тем, не учитывает модальности данных (авторы, теги, картинки, ссылки и т.д.). Кроме того, модель pLSA не позволяет управлять разреженностью. Действительно, если в начале работы алгоритма $\phi_{wt} = 0$ или $\theta_{td} = 0$, то и после завершения работы алгоритма значения этих параметров останется равным 0. Эти проблемы могут быть решены с помощью регуляризации модели pLSA, представленной в общем подходе ARTM, который будет описан ниже.

2.2. Модель LDA

Латентное размещение Дирихле (latent Dirichlet allocation, LDA) [9] — широко известная генеративная вероятностная тематическая модель для дискретных данных, в частности текстов. Подобный подход схож с pLSA с той разницей, что в LDA накладываются дополнительные ограничения на вид распределений $\phi_t \sim Dir(\beta)$ и $\theta_d \sim Dir(\alpha)$. Это приводит к различным модификациям M-шага. Самая простая и популярная из них следующая:

$$\phi_{wt} \propto n_{wt} + \beta_w, \quad \theta_{td} \propto n_{td} + \alpha_t,$$

что совпадает с регуляризатором сглаживания (4) в подходе ARTM. В результате получается более эффективная модель, что подтверждается успешным применением в задачах классификации и коллаборативной фильтрации.

2.3. Подход ARTM

Распределения ϕ_{wt} и θ_{td}, полученные моделью pLSA, слабо разрежены, что нежелательно на практике. Хотелось бы, чтобы в модели каждый документ d представлялся небольшим числом тем t с большими вероятностями $p(t|d)$ и каждая тема t состояла из небольшого числа слов с высокими вероятностями $p(w|t)$. Такая стратегия повышает интерпретируемость модели, обеспечивает более компактное представление данных. Однако ничто в моделировании pLSA не поощряет такой механизм обучения. В последнее время предпринималось немало попыток построить более разреженные распределения. Рассмотрим один из предложенных подходов — аддитивную регуляризацию тематических моделей (additive regularization for topic modeling, ARTM) [10].

Основная идея ARTM заключается в том, чтобы обеспечить гибкий способ добавления некоторой дополнительной информации о задаче к оптимизируемой функции правдоподобия (1). Делается это с помощью взвешенной суммы критериев регуляризации R_i:

$$R(\Phi,\Theta)=\sum_i \tau_i R_i(\Phi,\Theta), \quad L(\Phi,\Theta)+R(\Phi,\Theta)\to\max_{\Phi,\Theta}, \tag{2}$$

где $\tau_i \geq 0$ — коэффициенты регуляризации, регулируют силу действия регуляризатора и настраиваются экспериментально.

Для обучения тематической модели применяется EM-алгоритм. На E-шаге, как и в модели pLSA, оценивается по формуле Байеса вероятность $p(t|d,w)$ и выражается n_{tdw} — число троек, в которых терм w документа d связан с темой t:

$$n_{tdw}=n_{dw}p(t|d,w), \quad p(t|d,w)=\frac{\phi_{wt}\theta_{td}}{\sum_s \phi_{ws}\theta_{sd}}.$$

Введем оператор norm, который преобразует произвольный вектор $(x_i)_{i\in I}$ в вектор вероятностей дискретного распределения:

$$\underset{i\in I}{\operatorname{norm}}(x_i)=\frac{\max(0,x_i)}{\sum\limits_{j\in I}\max(0,x_j)}, \text{ для всех } i\in I,$$

если $x_i \leq 0$ для всех $i \in I$, то $\operatorname{norm}(x)=\mathbf{0}$.

Для решения сформулированной задачи многокритериальной оптимизации (2) достаточно модифицировать M-шаг:

$$\phi_{wt}=\underset{w\in W}{\operatorname{norm}}\left(n_{wt}+\phi_{wt}\frac{\partial R}{\partial \phi_{wt}}\right), \quad \theta_{td}=\underset{t\in T}{\operatorname{norm}}\left(n_{td}+\theta_{td}\frac{\partial R}{\partial \theta_{td}}\right) \tag{3}$$

$$n_{wt}=\sum_{d\in D} n_{tdw}, \quad n_{td}=\sum_{w\in d} n_{tdw},$$

где $R(\Phi,\Theta)$ — непрерывно дифференцируемая функция.

Различные регуляризаторы позволяют не только разреживать распределения, но и повышать согласованность, различность тем, отбирать темы, документы и термы, сглаживать распределения, если это необходимо, выполнять тематическую сегментацию, строить иерархические модели и т.д. Подход ARTM дает возможность комбинировать самостоятельные модели в одну общую модель путем преобразования M-шага.

В данной статье рассматривается минимальный набор регуляризаторов, который покрывает требования большинства задач тематического моделирования: регуляризаторы сглаживания, разреживания и декоррелирования.

Разреживающий и сглаживающий регуляризаторы предполагают, что модельные распределения ϕ_t и θ_d близки по дивергенции Кульбака-Лейблера к некоторым заданным распределениям $\beta = (\beta_w)_{w \in W}$ и $\alpha = (\alpha_t)_{t \in T}$.

- Разреживающий регуляризатор:

$$R(\Phi, \Theta) = -\beta_0 \sum_{t \in T} \sum_{w \in W} \beta_w \ln \phi_{wt} - \alpha_0 \sum_{d \in D} \sum_{t \in T} \alpha_t \ln \theta_{td} \to \max_{\Phi, \Theta}.$$

  По формулам (3) выражение для M-шага записывается следующим образом:

$$\phi_{wt} = \underset{w \in W}{\operatorname{norm}}\big(n_{wt} - \beta_0 \beta_w\big), \quad \theta_{td} = \underset{t \in T}{\operatorname{norm}}\big(n_{td} - \alpha_0 \alpha_t\big).$$

- Сглаживающий регуляризатор:

$$R(\Phi, \Theta) = \beta_0 \sum_{t \in T} \sum_{w \in W} \beta_w \ln \phi_{wt} + \alpha_0 \sum_{d \in D} \sum_{t \in T} \alpha_t \ln \theta_{td} \to \max_{\Phi, \Theta}.$$

  Применение формул (3) дает тоже выражение для M-шага, что и самая простая модификация модели LDA:

$$\phi_{wt} = \underset{w \in W}{\operatorname{norm}}\big(n_{wt} + \beta_0 \beta_w\big), \quad \theta_{td} = \underset{t \in T}{\operatorname{norm}}\big(n_{td} + \alpha_0 \alpha_t\big), \tag{4}$$

  если в качестве векторов гиперпараметров взять дискретные распределения α и β, умноженные на коэффициенты регуляризации: $(\beta_0 \beta_w)_{w \in W}$, $(\alpha_0 \alpha_t)_{t \in T}$.

- Декорреляция тем как минимизация ковариаций между столбцами ϕ_t и ϕ_s матрицы Φ:

$$R(\Phi) = -\tau \sum_{t \in T} \sum_{s \in T \setminus t} \sum_{w \in W} \phi_{wt} \phi_{ws} \to \max_{\Phi},$$

  где $\tau \geq 0$ — коэффициент регуляризации.

Формулы (3) М-шага в данном случае принимают вид:

$$\phi_{wt} = \underset{w \in W}{\text{norm}}\Big(n_{wt} - \tau\phi_{wt} \sum_{s \in T \setminus t} \phi_{ws}\Big).$$

Декоррелирующий регуляризатор стремится уменьшить пересечение между распределениями слов по темам ϕ_t, это повышает различность тем, что положительно влияет на интерпретируемость модели [11].

3. Полный набор тем

Построение вероятностной тематической модели является некорректно поставленной задачей стохастического матричного разложения. Множество ее решений в общем случае бесконечно:

$$F \approx \Phi\Theta = (\Phi S)(S^{-1}\Theta),$$

где S — произвольная невырожденная матрица, при условии, что матрицы $(\Phi S), (S^{-1}\Theta)$ стохастические.

Каждый раз модель находит локальный экстремум, поэтому нельзя гарантировать, что были найдены все темы корпуса. Однако в этом пространстве можно построить базис векторов тем тематических моделей ϕ_t, состоящий из линейно независимых, хорошо интерпретируемых тем.

3.1. Алгоритм построения полного набора тем

Из неустойчивости тем следует проблема полноты набора тем, найденного тематической моделью. Возникают следующие вопросы:

- действительно ли темы новые или это комбинации предыдущих;
- можно ли найти все темы корпуса, сколько моделей для этого нужно построить.

Для ответов на эти вопросы рассмотрим алгоритм построения базиса тем моделей, отличающихся инициализацией. Все описанные выше модели — pLSA, ARTM и LDA — находят темы в пространстве распределений над словами. Каждое такое распределение можно рассматривать как точку в единичном $(|W| - 1)$-симплексе слов $\triangle$.

Определение. *Множество* $V = \{v_1, \ldots, v_m\} \subset \triangle$ *— базис тем множества матриц тематических моделей* $\Phi_1, \ldots, \Phi_n \subset \triangle$, *если* $\forall \phi \in \Phi_j$ *выполняется:*

$$\min_{v \in convV} \rho(\phi, v) \leq \varepsilon, \tag{5}$$

$$convV = \Big\{ v = \sum_{i=1}^{m} \alpha_i v_i \big| v_i \in V, \quad \sum_i \alpha_i = 1, \quad \alpha_i \geq 0 \Big\},$$

где $\rho(\phi, v)$ *— функция расстояния.*

Из определения следует, что базис V состоит из векторов тем $\phi \in \Phi_j, \quad j = \overline{1, n}$, для которых $\min\limits_{v \in convV} \rho(\phi, v) > \varepsilon$, именно они линейно независимы. Кроме того, решение каждой отдельной тематической модели локально оптимально, а значит его можно улучшить с помощью замены тем на близкие им в случае повышения правдоподобия (log-likelihood). На этом и основан предлагаемый жадный алгоритм для нахождения базиса тем.

Алгоритм состоит из чередования двух этапов. На первом этапе происходит замена тем с целью расширения имеющейся системы векторов тем и максимизации правдоподобия. Алгоритм итеративный, на каждой итерации для тем набора находятся схожие с некоторым порогом γ и выполняется замена, если она улучшает правдоподобие всего набора тем. На втором этапе происходит поиск темы для добавления в набор:

- включаются линейно независимые темы;
- исключаются выбросы (выбросами считались темы, которые имеют более двух коэффициентов $\alpha_i > 0.15$ в (5), что соответствовало несогласованным темам, зависимым нелинейно от тем полного набора);
- выбирается тема, максимально повышающая правдоподобие набора тем.

Таким образом, мы дополняем линейно независимую систему векторов до базиса. Описанные шаги повторяются до сходимости. Так как алгоритм итеративный, то после того, как алгоритм сошелся, нужно выполнить процедуру замены близких тем, используя обученные модели в обратном порядке.

Для решения оптимизационной задачи (5) использовался алгоритм SLSQP (Sequential Least SQuares Programming) [12], который реализован во многих популярных математических пакетах, в том числе и SciPy.

В результате экспериментов было установлено, что все темы корпуса можно найти за конечное число итераций алгоритма построения выпуклой оболочки тем тематических моделей. Регуляризатор декорреляции, применяемый при построении моделей ARTM, способствует нахождению более мелких тем, поэтому в полном наборе содержится больше тем, чем в базисе моделей LDA. На рис. 1, рис. 2 видно, что правдоподобие полного набора тем выше, чем одной модели с аналогичным числом тем. Замечено, что число тем в полном наборе зависит от степени разреженности матриц Ф. Базис более разреженных моделей содержит больше тем.

4. Эмпирические результаты

В этом разделе мы приводим результаты работы алгоритмов на реальных данных коллекции ПостНаука[1]. Коллекция ПостНаука — небольшой корпус текстов интернет-журнала ПостНаука, состоящий из научно-популярных статей о современной фундаментальной науке и учёных, которые её создают.

В экспериментах, описанных ниже, использовались теги к документам коллекции ПостНаука, размеченные экспертами. Всего было 930 тегов, которые обозначали общие и частные понятия, например, биология, клетка, эволюционная биология, ген, эукариоты, квантовая физика, Россия, человек и т.д. Каждый документ в среднем описывался 6 тегами. Данные были случайно разделены на обучающую и контрольную выборки в отношении 80% к 20%.

4.1. Моделирование документов

Для построения полного набора рассматривались модели LDA Gibbs sampling и ARTM с 20 темами. LDA[2] с параметрами $\eta = 0.01$, $\alpha = \frac{1}{20}$ обучалась 3500 итераций. При построении моделей ARTM использовалась следующая стратегия регуляризации: сначала включался декоррелирующий регуляризатор до сходимости модели по перплексии, затем он

[1] https://postnauka.ru/

[2] https://github.com/lda-project/lda

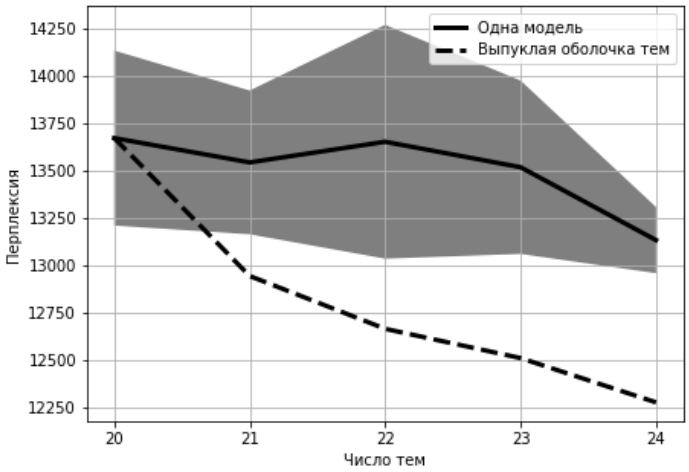

Рис. 1. Сравнение перплексии одной модели и полного набора тем моделей ARTM на тегах коллекции ПостНаука, параметр $\delta = 1.2, \gamma = 0.43$.

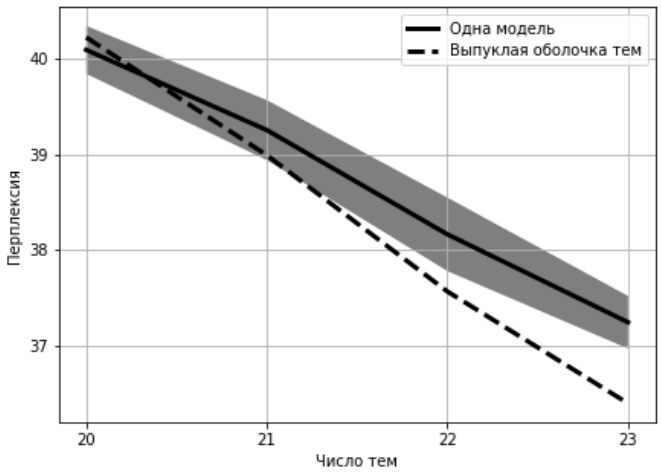

Рис. 2. Сравнение перплексии одной модели и полного набора тем моделей LDA (с использованием Gibbs Sampling) на тегах коллекции ПостНаука, параметр $\delta = 0.64, \gamma = 0.43$.

отключался, и применялись регуляризаторы разреживания матрицы Θ и сглаживания матрицы Φ. Во всех случаях матрицы Θ, Φ — случайные стохастические матрицы. Модели ARTM строились с помощью библиотеки BigARTM[3].

4.2. Полнота и интерпретируемость тематических моделей

Тематические модели находят локально экстремальное решение, которое не является полным. При этом на практике часто возникает задача найти все темы корпуса. Для решения данной задачи в статье был предложен алгоритм нахождения базиса тем.

В табл. 3, табл. 4 приводятся некоторые темы, найденные с помощью алгоритма построения полного набора тем. Алгоритм применялся для моделей LDA и ARTM. Процесс накопления тем базиса проиллюстрирован на рис. 1 и рис. 2. На графиках видно, что правдоподобие базиса тем выше, чем правдоподобие одной модели. Кроме того, было проведено сравнение одной модели и базиса тем по дополнительным критериям качества тематических моделей (см. табл. 1, табл. 2): критерию Р. Аруна [13], когерентности (автоматической меры интерпретируемости) [14] и критерию Цао Хуана [15]. На обучении полный набор тем моделей LDA лучше по всем критериям, чем одна модель. На контроле ухудшение только по когерентности. Базис моделей ARTM имеет более высокое правдоподобие на обучении и контроле, чем одна модель.

Полный набор тем ARTM позволяет более подробно описать корпус, чем базис моделей LDA. Например, тема о генетике в полном наборе моделей LDA содержит термы биология, ген, днк, генетика, а у моделей ARTM — молекулярная биология, клеточная биология, генетика, генная инженерия. Однако не все темы ARTM согласованные, даже в топе содержатся слова, не относящиеся к данной теме, например, русская философия, бактерии, народная культура, Русь. Еще один недостаток базиса ARTM заключается в том, что могут быть темы, которая не выделились в отдельные, а были размыты по нескольким темам базиса. Эти недостатки можно исправить с помощью дальнейшей регуляризации.

В экспериментах было построено два полных набора тем моделей LDA. В первом случае модели следовали в прямом порядке, во втором — в обратном. Темы базисов соотносились между собой с помощью венгерского алгоритма. Темы совпали с высокой точностью. Однако одна тема в первом базисе была разделена на две во втором: астрономия,

[3]https://github.com/bigartm/bigartm

астрофизика, космология, гравитация на астрономию, космос и физику, астрофизику. Базис моделей ARTM тоже разделяет эти две темы.

На рис. 3 и рис. 4 изображены плоские проекции[4] базисов ARTM и LDA. Видно, что в базисе моделей LDA все темы крупные и разделились на следующие кластеры: биология, физика, химия, лингвистика, экономика, психология, математика, история, философия, культура, социология. В полном наборе тем ARTM содержатся как крупные темы, так и мелкие, например, русская философия, судопроизводство и искусствоведение.

Критерии	На обучении		На контроле	
	Одна модель	Базис тем	Одна модель	Базис тем
Перплексия	13132.6	**12227.45**	12835.1	**12534.36**
Критерий Аруна	**52.15**	56.90	**9.95**	12.96
Когерентность	**-7.89**	-9.01	-13.50	**-12.75**
Критерий Хуана	**0.01**	0.02	**0.01**	0.02

Таблица 1. Сравнение результатов работы одной модели ARTM с 24 темами (бралось среднее значение по пяти моделям) и полного набора тем ARTM. Модели набора содержали 20 тем, в результате работы алгоритма было отобрано 24 темы. Жирным выделены оптимальные значения в сравнении.

Критерии	На обучении		На контроле	
	Одна модель	Базис тем	Одна модель	Базис тем
Перплексия	37.36	**36.39**	44.91	**43.55**
Критерий Аруна	100.93	**95.18**	31.84	**28.88**
Когерентность	-3.01	**-2.91**	**-5.70**	-6.44
Критерий Хуана	0.07	**0.06**	0.07	**0.06**

Таблица 2. Сравнение результатов работы одной модели LDA с 23 темами (бралось среднее значение по пяти моделям) и полного набора тем LDA с применением Gibbs sampling.

[4]https://github.com/bmabey/pyLDAvis

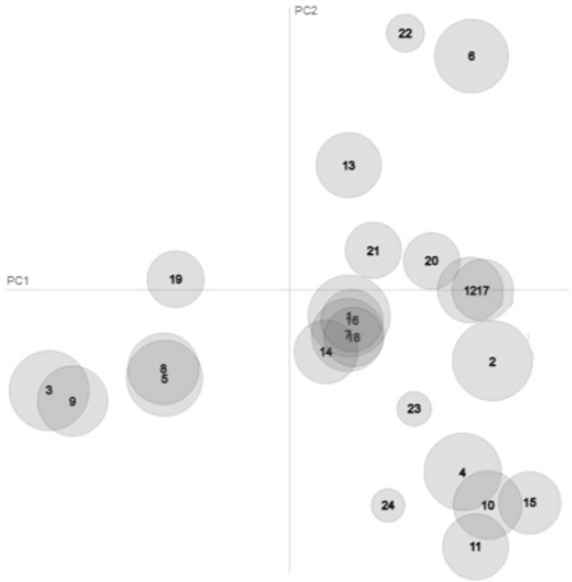

Рис. 3. Плоская проекция полного набора тем моделей ARTM, построенного на тегах коллекции ПостНаука.

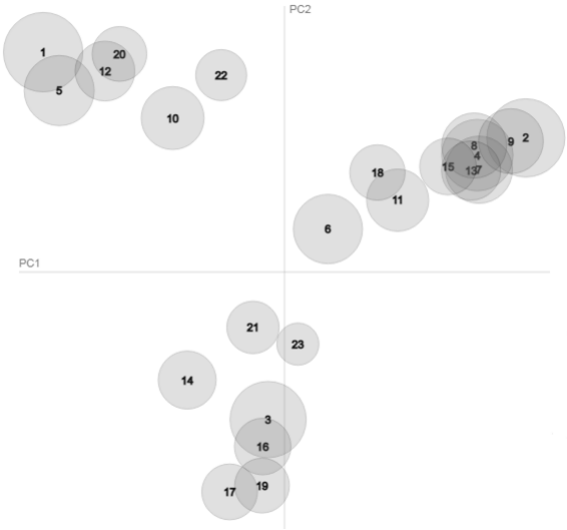

Рис. 4. Плоская проекция полного набора тем моделей LDA, построенного на тегах коллекции ПостНаука.

Базис тем, полученный в порядке $\{\Phi_i\}_{i=N}^{0}$	Базис тем, полученный в порядке $\{\Phi_i\}_{i=0}^{N}$
история, культура, религия, христианство, востоковедение, китай, ислам	история, культура, религия, христианство, востоковедение, ислам, археология
культура, общество, культурология, массовая культура, философия, кино, искусство	культура, культурология, массовая культура, философия, кино, общество, искусство
биология, ген, днк, геном, клетка, генетика, микробиология	биология, ген, днк, геном, генетика, клетка, белки
физика, квантовая физика, технологии, атом, квантовая механика, сверхпроводимость, квантовые технологи	физика, квантовая физика, технологии, оптика, квантовые технологии, квантовая механика, сверхпроводимость
медицина, биология, генетика, биомедицина, онкология, клетка, технологии	медицина, биология, генетика, биомедицина, онкология, клетка, стволовые клетки

Таблица 3. Сравнение двух полных наборов тем моделей LDA, построенных на тегах коллекции ПостНаука. В первом случае модели следовали в обратном порядке, во втором — в прямом.

5. Заключение

Данная работа посвящена изучению вопроса полноты тематических моделей. Проведен сравнительный анализ моделей pLSA, LDA и моделей подхода ARTM по критериям, связанным с исследуемой проблемой. Во всех моделях только часть тем оказалась устойчивой. Показано, что декоррелирующий регуляризатор ухудшает устойчивость модели и способствует нахождению более мелких тем.

Неустойчивость тематических моделей является известным фактом, однако связанная с ней проблема полноты до сих пор в литературе не изучалась. Для решения этой задачи в статье предложен новый алгоритм нахождения полного набора тем, основанный на построении выпуклой оболочки векторов тем тематических моделей, отличающихся только инициализацией матрицы Ф. Алгоритм состоит из двух процедур — замены близких тем и добавления новой темы — которые выполняются

Базис тем, полученный в порядке $\{\Phi_i\}_{i=0}^{N}$	Базис тем, полученный в порядке $\{\Phi_i\}_{i=N}^{0}$
история, культура, религия, христианство, востоковедение, ислам, археология	история науки, история, римское право, право, религия, история права, социология права
культура, культурология, массовая культура, философия, кино, общество, искусство	культура, массовая культура, культурология, кино, искусство, кинематограф, фрейд зигмунд
биология, ген, днк, геном, генетика, клетка, белки	молекулярная биология, клеточная биология, биология, генетика, генная инженерия, ген, днк
физика, квантовая физика, технологии, оптика, квантовые технологии, квантовая механика, сверхпроводимость	русская философия, бактерии, народная культура, коренные народы, мертон роберт, русь, доказательная медицина
медицина, биология, генетика, биомедицина, онкология, клетка, стволовые клетки	медицина, стволовые клетки, биология, биомедицина, молекулярная биология, индуцированные плюрипотентные стволовые клетки, регенеративная медицина

Таблица 4. Сравнение полного набора тем моделей LDA и полного набора тем моделей ARTM, построенных на тегах коллекции ПостНаука.

по очереди до сходимости алгоритма. Таким образом, происходит приближенное дополнение линейно независимой системы до базиса. По построению в базис добавляются только те новые темы, в δ-окрестности которых нет выпуклых комбинаций тем базиса.

Замечено, что число тем и скорость сходимости алгоритма зависит от степени разреженности матриц Ф. Базис более разреженных моделей содержит больше тем. Самый маленький базис у моделей LDA. Он содержит наиболее крупные и общие темы, что можно наблюдать на плоской проекции базиса LDA.

Правдоподобие базиса моделей выше, чем одной модели с аналогичным числом тем, на обучении и контроле. Кроме того, было проведено сравнение одной модели и базиса тем по дополнительным критериям качества тематических моделей: критерию Р. Аруна, когерентности и кри-

терию Цао Хуана. На обучении полный набор тем моделей LDA лучше по всем критериям, чем одна модель. На контроле ухудшение только по когерентности.

В экспериментах также было проведено сравнение полных наборов тем моделей LDA и ARTM. В базисах LDA, построенных в прямом и обратном порядке следования моделей, темы совпали с высокой точностью. Однако одна тема в первом базисе была разделена на две во втором: астрономия, астрофизика, космология, гравитация на астрономию, космос и физику, астрофизику. Базис моделей ARTM также разделил эти две темы.

Список литературы

[1] Blei D., Carin L., Dunson D, "Probabilistic Topic Models: A focus on graphical model design and applications to document and image analysis", *IEEE signal processing magazine*, **27**:6 (2010), 55.

[2] Steyvers M., Griffiths T., "Probabilistic topic models", *Handbook of latent semantic analysis*, **427**:7 (2007), 424–440.

[3] De Waal A., Barnard E., "Evaluating topic models with stability", 2008.

[4] Koltcov S., Koltsova O., Nikolenko S., "Latent dirichlet allocation: stability and applications to studies of user-generated content", *Proceedings of the 2014 ACM conference on Web science*, «ACM», 2014, 161–165.

[5] Greene D., O'Callaghan D., Cunningham P., "How many topics? stability analysis for topic models", *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, «Springer», Berlin, Heidelberg, 2014, 498–513.

[6] Balagopalan A., "Improving topic reproducibility in topic models.", 2012.

[7] Lancichinetti A. et al., "High-reproducibility and high-accuracy method for automated topic classification", *Physical Review X.*, **5**:1 (2015), 011007.

[8] Hofmann T., "Probabilistic latent semantic indexing", *ACM SIGIR Forum*, **51**:2 (2017), 211–218.

[9] Blei D.M., Ng A.Y., Jordan M.I., "Latent dirichlet allocation", *Journal of machine Learning research*, **3**:Jan (2003), 993–1022.

[10] Vorontsov K., Potapenko A., "Additive regularization of topic models", *Machine Learning*, **101**:1–3 (2015), 303–323.

[11] Tan Y., Ou Z., "Topic-weak-correlated latent dirichlet allocation", *2010 7th International Symposium on Chinese Spoken Language Processing*, «IEEE», 2010, 224–228.

[12] Kraft D., "A software package for sequential quadratic programming", *Forschungsbericht- Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt*, 1988.

[13] Arun R. et al., “On finding the natural number of topics with latent dirichlet allocation: Some observations”, *Pacific-Asia conference on knowledge discovery and data mining*, «Springer», Berlin, Heidelberg, 2010, 391–402.

[14] Mimno D. et al., “Optimizing semantic coherence in topic models”, *Proceedings of the conference on empirical methods in natural language processing*, «Association for Computational Linguistics», 2011, 262–272.

[15] Cao J. et al., “A density-based method for adaptive LDA model selection”, *Neurocomputing*, **72**:7–9 (2009), 1775–1781.

Building a complete set of topics of probabilistic topic models

Sukhareva A.V., Vorontsov K.V.

Interpretability, linear increase in complexity with data growth, scalability made topic modeling one of the most popular tools for statistical text analysis. However, there are a number of disadvantages caused by the dependence of the solution on the initialization. It is known that the building of a topic model is reduced to solving an ill-posed problem of the non-negative matrix factorization. The set of its solutions in the general case is infinite. Every time the model finds a local extremum. Repeated training of the model for the same collection can lead to detection of more and more new topics. In practice, it is often necessary to define all the topics of the corpus. To solve this problem, the article proposed and investigated a new algorithm for finding the complete set of topics based on the construction of a convex hull. It was shown experimentally that it is possible to construct a basis for the finite number of models. The likelihood of the basis is higher than for a single model with a similar number of topics. Compare of the basis of LDA models (latent Dirichlet allocation) and ARTM models (additive regularization for topic modeling) suggests that the topics of the sets coincide with high accuracy.

Keywords: probabilistic topic modeling, stability of topic models, complete set of topics of topic models, latent Dirichlet allocation, LDA, regularization, ARTM, BigARTM.

Часть 2.
Специальные вопросы теории интеллектуальных систем

Аппаратная конструкция для решения проблемы экспоненциального взрыва для одного класса регулярных языков

Бернадотт А., Галатенко А.

Аннотация

Известно, что язык, задаваемый регулярным выражением вида $\bigcup_{i=1}^{n} .*\alpha_i.*\beta_i.*$, где α_i, β_i — слова над некоторым алфавитом, в общем случае для распознавания конечным детерминированным автоматом требует экспоненциальное по n число состояний. В работе предлагается конструкция структурного автомата, распознающего языки из данного класса и имеющего полиномиальную пространственную сложность.

Ключевые слова: ДКА, структурный автомат, регулярный язык, экспоненциальный взрыв.

1. Введение

В настоящее время сетевые устройства могут проверять содержимое пакета с целью управления сетевым трафиком и сканирования на вредоносность.

В анализе сетевого трафика актуальной является скорость анализа, так как последний может значительно снижать пропускную способность сетевого узла. По скорости сканирования и по уровневой модели реализации можно выделить два основных класса анализаторов: 1) сетевые процессоры (англ. Network Processing Unit, NPU) — это программируемые микропроцессоры, оптимизированные для работы в сетевых устройствах в режиме обработки пакетов (packet processing); 2) интегральные схемы специального назначения, (англ. Application-Specific Integrated Circuit, ASIC) — это интегральные схемы, обладающие ограниченным функционалом, реализуемым под строго заданную задачу. Тогда как сетевой процессор является программируемым устройством (в данной статье устройство такого вида не рассматривается), которое обеспечивает высокий

уровень пластичности, ASIC обеспечивает высокую производительность, но не обеспечивает способности к репрограммируемости.

Оба типа устройства могут иметь интегрированные модули внешней и встроенной памяти. Количество встроенных модулей памяти с достаточной пропускной способностью ("быстрой памяти") ограничено значительно, а количество внешних элементов памяти с низкой пропускной способностью ограничено только размером устройства. Критичной является проблема объема и доступности встроенной памяти.

Один из основных методов анализа заключается в следующем. Эксперты задают сигнатуры злоумышленного поведения; анализатор просматривает сетевые пакеты и выявляет соответствие пакетов хотя бы одной сигнатуре. Для задания сигнатур используется аппарат регулярных выражений. По теореме Клини задача проверки принадлежности слова соответствующему регулярному языку может быть решена с помощью конечого детерминированного автомата с временной сложностью, линейной от длины слова. Однако число состояний автомата может расти экспоненциально с ростом длины регулярного выражения, то есть возможен экспоненциальный взрыв пространственной сложности. В результате возникает задача понижения пространственной сложности распознавания без существенного ухудшения временной сложности. Некоторые известные подходы к решению излагаются ниже.

Мы рассматриваем класс регулярных выражений вида $\bigcup_{i=1}^{n} .*\alpha_i.*\beta_i.*$, где α_i, β_i — слова в некотором алфавите, $n \in \mathbb{N}$. С одной стороны, в общем случае распознавание принадлежности слова такому языку требует экспоненциального по n числа состояний; с другой стороны, сигнатуры вида $.*\alpha.*\beta.*$ часто встречаются в практических приложениях. Для уменьшения пространственной сложности предлагается разделить автомат-распознаватель на два подавтомата — абстрактный, в котором функции переходов и выходов заданы таблично, и структурный. Доказывается, что возникающая конструкция корректно решает задачу распознавания и позволяет избежать экспоненциального взрыва.

Дальнейшее изложение построено следующим образом. Во втором разделе приводятся основные определения и дается краткий обзор по проблеме экспоненциального взрыва. В третьем разделе описывается предлагаемая конструкция. В четвертом разделе формулируются и доказываются основные утверждения, в пятом рассматривается несколько примеров. Наконец, шестой раздел является заключением.

2. Описание проблемы и основные определения

Напомним основные определения в соответсвии с источниками [1, 2].

Регулярный язык (событие или множество) над алфавитом A задаётся рекурсивно:

1) $\emptyset, \{\lambda\}$ и $\{a\}$ — регулярные языки, где λ — пустое слово, $a \in A$;

2) если M_1 и M_2 — регулярные языки, то объединения $M_1 \cup M_2$ и конкатенация $M_1 \cdot M_2$ языков M_1 и M_2, а также здезда Клини M_1^* — регулярные языки, где

$$M_1 \cup M_2 = \{\alpha_1 \cup \alpha_2 | \alpha_1 \in M_1, \alpha_2 \in M_2\},$$
$$M_1 \cdot M_2 = \{\alpha_1 \cdot \alpha_2 | \alpha_1 \in M_1, \alpha_2 \in M_2\},$$
$$M_1^* = \{\lambda\} \cup \{\alpha_1 \cdot \ldots \alpha_n | \alpha_i \in M_1, i \in \overline{1,n}, n \in N\}.$$

Регулярное выражение — язык над алфавитом $A \cup \{\cup, (,), \cdot, *\}$, задающий регулярное событие и определяемый как:

1) $\emptyset, \lambda, a$ — регулярные выражения, где $a \in A$;

2) если R_1, R_2 — регулярные выражения, то слова $(R_1 \cup R_2), (R_1 \cdot R_2), (R_1)*$ — регулярные выражения.

В дальнейшем мы будем предполагать, что операция $*$ имеет максимальный приоритет, а операция $\cup$ — минимальный, и не писать скобки, если результат определен однозначно. Кроме того, символ $\cdot$ будет иногда опускаться.

Регулярный язык, задаваемый регулярным выражением, определяется естественным образом.

На практике, в точности, в базах Snort, Zeek, Cisco, регулярные выражения представляют с использованием нотации PCRE (Perl Compatible Regular Expressions) [3]. Приведем PCRE —обозначения, которые будем использовать в работе:

“.” — произвольный символ из алфавита;

$R\{n\}$ — степень n языка R относительно конкатенации.

Проверка принадлежности слова регулярному языку, в классическом варианте, осуществляется принимающим абстрактным конечным автоматом (КА).

Абстрактным конечным автоматом называется набор $V = (A, Q, B, \varphi, \psi)$, где A, Q, B — конечные множества: входной алфавит, алфавит состояний и выходной алфавит, φ — функция переходов, $\varphi : Q \times A \to Q$, ψ — функция выходов, $\psi : Q \times A \to B$. Инициальный конечный автомат имеет отдельно выделенное начальное состояние, в котором начинается работа автомата: $V = (A, Q, B, \varphi, \psi, q_0)$, где q_0 — начальное состояние автомата [1]. Недетерминированный конечный автомат (НДКА) позволяет осуществлять переход по пустому слову и представляет собой набор $V = (A, Q, B, \varphi, \psi)$, где A, Q, B — конечные множества: входной алфавит, алфавит состояний и выходной алфавит, φ — функция переходов, $\varphi : Q \times A \cup \lambda \to 2^Q$, ψ — функция выходов, $\psi : Q \times A \cup \lambda \to B$. Детерминированный конечный автомат (ДКА) не имеет λ–переходов — переходов по пустому слову, и все переходы ДКА определены однозначно [1, 4].

Автомат может использоваться для представления языков: входное слово α считается принадлежащим некоторому языку L если и только если последний выданный автоматом символ лежит в принимающем множестве $B_0 \subset B$ (распознавание выходами) или состояние, в которое переводит автомат слово α, содержит элемент принимающего множества $Q_0 \subset Q$ (распознавание состояниями). Согласно теореме Клини, событие над алфавитом A представимо конечным автоматом тогда и только, когда является регулярным языком [1, 2].

КА, представляющий регулярный язык, может быть недетерминированным и детерминированным. Согласно теореме о эквивалентности детерминированных и недетерминированных конечных автоматов, всякий язык принимается некоторым НДКА тогда и только тогда, когда этот язык принимается некоторым ДКА [4]. Кроме того, существует алгоритм построения из НДКА эквивалентного ему ДКА [4, 5].

Основное преимущество недетерминированного автомата — низкая пространственная сложность, линейная по длине регулярного выражения; основной недостаток — высокая временная сложность, так как время обработки одного символа входного слова вообще говоря также линейно по длине регулярного выражения.

При переходе от недетерминированного автомата к детерминированному временная сложность обработки входного символа становится константной; с другой стороны, согласно теореме Лупанова, в случае, если входной алфавит содержит хотя бы два символа, мощность множества состояний вообще говоря экспоненциально возрастает [1, 6]. В качестве примера рассмотрим класс регулярных выражений $\bigcup_{i=1}^{n} . * \alpha_i . * \beta_i . *$, где

α_i, β_i — некоторые слова в алфавите A. Известно, что в общем случае для распознавание принадлежости слова соответствующему языку требуется экспоненциальное по n число состояний [7]. В базу сигнатур системы выявления вторжений Snort на 2019 год содержится 3936 выражений вида $.*\alpha.*\beta.*$ [8]; автомат, распознающий язык, порожденный такими выражениями, заведомо не помещается в память. Проблема экспоненциально растущего числа состояний получила название экспоненциального взрыва [1, 9, 6].

Проблемой экспоненциального взрыва занимаются более 20 лет. Глобально можно выделить выделить три подхода к её решению. Первый подход связан с ограничением на сигнатуры, задаваемые экспертами. Однако приведенный выше пример показывает, что даже очень простые, заведомо линейные по пространственной сложности регулярные выражения при объединении могут давать экспоненциальный рост.

Второй подход основан на изменении распознаваемого языка с целью упрощения распознающего автомата. При этом измененный язык отличается от исходного, поэтому возникают ошибки распознавания. В качестве примера можно привести предложенную Александровым идею оптимизации языков вида $\bigcup_{i=1}^{n} .*R_i.*R_i^{'}.*$ за счет замены пары слагаемых на слагаемое $(R_{i_1}|R_{i_2}).*(R_{i_1}^{'}|R_{i_2}^{'})$. Оценка выигрыша от такой замены приведена в работах [10, 11], оценка погрешности — в работе [12].

Третий подход заключается в модификации структуры конечного автмата. Примерами модификаций являются рассмотрение нескольких автоматов, работающий параллельно, сжатие диаграммы Мура за счет добавления переходов по умолчанию, добавлению просто устроенных недетерминированных или структурных компонет. Обзор модификаций приведен, например, в работе Александрова [7]. Отдельно отметим конструкцию, предложенную Кумаром и заключающуюся в добавлении к абстрактному автомату счетчиков и битовых масок и навешивании дополнительных логических выражений на ребра диаграммы перехода [9]. Такое решение позволяет решать задачу распознавания выражений вида $\bigcup_{i=1}^{n} .*\alpha_i.*\beta_i.*$, избегая экспоненциального взрыва пространственной сложности.

В данной работе предлагается решение, которое можно отнести к третьему типу. Автомат декомпозируется на две компоненты — “абстрактную” (задаваемую таблично; на практике таблицы значений записываются в память) и “структурную” (конфигурируемую за счет начального состояния задержек). В отличие от конструкции Кумара, используется немодифицированный автомат; основной выигрыш получается за счет

перехода от чисто абстрактного автомата к структурному. В дальнейшем предполагается провести накопление структурных компонент, позволяющих решить проблему экспоненциального взрыва для максимально широкого класса языков.

3. Аппаратная конструкция КА в контексте проблемы экспоненциального взрыва

Фиксируем конечный непустой входной алфавит A и рассмотрим класс языков $\bigcup_{i=1}^{n} . * \alpha_i . * \beta_i .*$, где α_i, β_i — непустые слова в алфавите A. Известно, что в общем случае для представления языка из данного класса требуется экспоненциальное по n число состояний конечного детерминированного автомата [7]. Ниже описывается конструкция, позволяющая избежать экспоненциального взрыва пространственной сложности за счет декомпозиции распознающего автомата на две компоненты — "абстрактную", функционирование которой задается таблично, а сложность определяется как объем памяти, требующейся для хранения таблицы значений, и "структурную", сложность которой определяется как число элементов в схеме. Заметим, что, с одной стороны, уже тривиальная схема из k элементов задержки задает автомат с 2^k состояниями; с другой стороны, почти все булевы функции от k переменных имеют экспоненциальную схемную сложность.

Сперва опишем идею конструкции на содержательном уровне. Общая схема представлена на рис. 1. Автомат состоит из трех блоков. Первый блок с помощью выходов распознает язык $L_1 \cup L_2$, где $L_1 = \bigcup_{i=1}^{n} . * \alpha_i$, $L_2 = \bigcup_{i=1}^{n} . * \beta_i$. Выходной алфавит представляет из себя множество двоичных векторов длины $2n$. Для $1 \leq i \leq n$ iая компонента выхода обращается в единицу тогда и только тогда, когда входное слово принадлежит языку $. * \alpha_i$; если же $n + 1 \leq i \leq 2 * n$, то равенство iой компоненты выхода единице эквивалентно принадлежности входного слова языку $. * \beta_i$. Первые n компонент выхода подаются на вход второго блока и включают счетчики, отсчитывающие длины соответствующих слов β_i. Если требуемая длина достигнута, единичный сигнал передается на вход блока 3, вычисляющего конъюнкции выходов первого блока с номерами от $n+1$ до $2n$ с соответствующими выходами второго блока. Выход конъюнкции подается на вход переключателя, запоминающего единичный сигнал. Таким образом, равенство выхода переключателя единице эквивалентно выполнению следующих условий:

1) во входном потоке встретилось слово α_i;

2) не менее, чем через $|\beta_i|$ тактов после этого во входном потоке встретилось слово β_i.

Несложно увидеть, что эти условия эквивалентны тому, что входное слово принадлежит языку $.*\alpha_i.*\beta_i.*$.

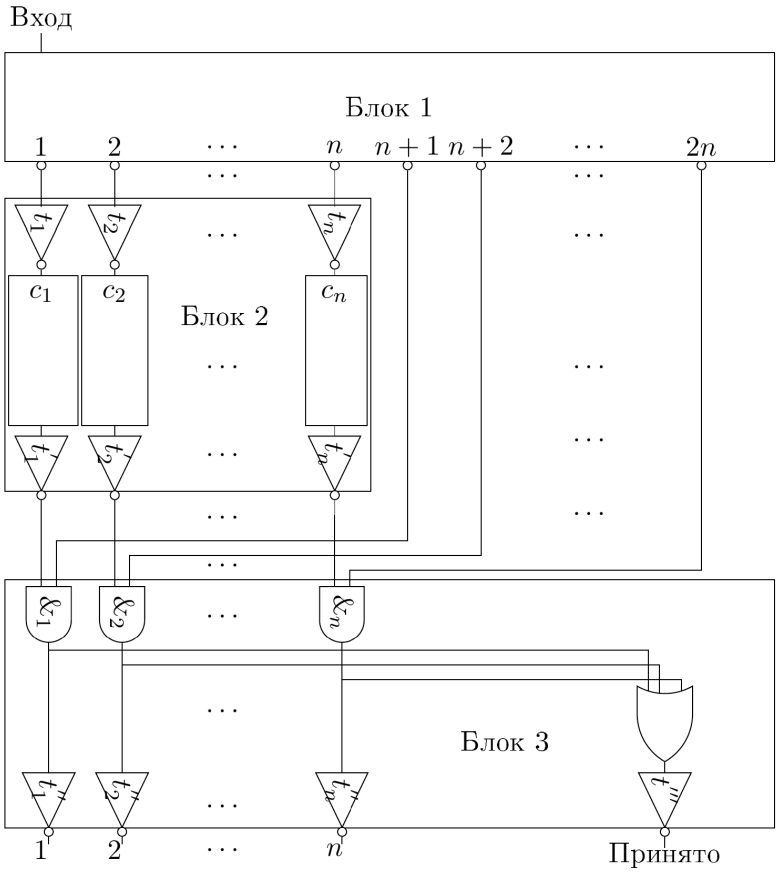

Рис. 1. Аппаратная конструкция, t_i, t'_i, t''_i, t''' — триггеры; c_i — счётчики.

Опишем аппаратную конструкцию более подробно. Распознающая аппаратная конструкция состоит из 3–х блоков. Блок 1 представляет собой ДКА, принимающий регулярный язык $(\bigcup_{i=1}^{n} .*\alpha_i) \bigcup (\bigcup_{i=1}^{n} .*\beta_i)$, который строится путём применения алгоритма Ахо и Корасик [13]. В данном ДКА словам, описываемым разными регулярными выражениями вида $.*\alpha_i$ и $.*\beta_i$, соответствуют различные принимающие состояния

ДКА. Блок 1 имеет 1 вход и $2n$ выходов. Первые n выходов блока 1 отвечают за распознавание языка $\bigcup_{i=1}^{n} . * \alpha_i$. То есть на i–ый выход, $i = 1, \ldots, n$, идет единичный сигнал тогда и только тогда, когда на данном такте вход блока 1 принадлежит соответствующему слагаемому из языка $\bigcup_{i=1}^{n} . * \alpha_i$. Сигнал от $\overline{1, n}$ выходов блока 1 идёт в блок 2 — на входы n счётчиков — выход, соответствующий номеру слова $. * \alpha_i$ из $\bigcup_{i=1}^{n} . * \alpha_i$, идёт на счётчик, соответствующий номеру этого же слова в блоке 2. Выходы блока 1 с номерами $\overline{n+1, 2n}$ отвечают за распознавание языка $\bigcup_{i=1}^{n} . * \beta_i$. Сигнал от этих $\overline{n+1, 2n}$ выходов идёт на блок 3 — на вторые входы соответствующих n конъюнкций — $\overline{\&1, \&n}$. На второй вход i–ого логического вентиля "конъюнкция" от $n + i$–ого выхода блока 1 идет единичный сигнал тогда и только тогда, когда на данном такте вход принадлежит соответствующему слагаемому — слово $. * \beta_i$ принимается, иначе выход равен 0.

Блок 2 — отсчитывает длины слов языка, представляющего объединение вторых пар слов: $\bigcup_{i=1}^{n} \beta_i$ начального регулярного языка $\bigcup_{i=1}^{n} . * \alpha_i . * \beta_i . *$. Блок 2 состоит из ряда n однополюсных однопозиционных переключателей, следующим за ним ряда n регистровых счётчиков и следующим за ним третьим рядом n однополюсных однопозиционных переключателей. Каждый счётчик соответствует своему слову β_i и осуществляет счёт до $|\beta_i|$. Счётчик представляет собой композицию включающего мультиплексора, функции сложения с константой 1 и компаратора. Величина, до которой ведется счет, задается через начальное состояние. Блок имеет n входов и n выходов. На входы блока 2 поступает сигнал от первых n выходов блока 1 — от выходов с номерами $\overline{1, n}$. Сигнал от соответствующего i–го входа бока 2 поступает на i–ый переключатель. При поступлении единичного сигнала на вход i–того переключателя, переключатель размыкается и генерирует на своём выходе логическую единицу. В обратном случае, переключатель находится в замкнутом состоянии и подаёт на выход логических ноль. Единичный сигнал от i–го переключателя активирует соответствующий i–ый счётчик. Переведенный в считающее состояние счётчик увеличивает свое значение на единицу каждый такт до достижения значения $|\beta_i|$. Достижение i–ым счётчиком максимального значения посылает на вход i–ого однополюсного однопозиционного переключателя из третьего ряда логическую единицу и размыкает переключатель. Данные переключатели в соответствии со своим номером посылают на выходы блока 2 сигналы: ноль, если соответствующий переключатель замкнут, единицу — если он разомкнут. После размыкания переключатель не может перейти в замкнутое состояние и посылает ло-

гическую единицу на выход блока 2 каждый последующий такт. Таким образом единичный сигнал на выход i блока 2 начинает поступать тогда и только тогда, когда блок 1 послал единицу на соотвествующий i–ый переключатель, i–ый счётчик достиг максимального значения и разомкнул i–ый переключатель.

Блок 3 — блок выходных сигналов конструкции. Блок состоит слоя $n + 1$ логических вентилей (n логических вентилей типа "конъюнкция" и 1 логического вентиля типа "дизъюнкция") и следующего за ним слоя $n + 1$ однополюсных однопозиционных переключателей. На вход блока 3 поступает сигнал от блока 2 на первые входы "конъюнкций" и блока 1, распознающего слова $\bigcup_{i=1}^{n} . * \beta_i$, на вторые входы "конъюнкций".

Таким образом, на выход блока 3 идет сигнал о распознанном слове вида $. * \alpha_i . * \beta_i$ от соответствующего i–того логического вентиля (конъюнкции) тогда и только тогда, когда i–тый счётчик блока 2 перешел в финальное состояние и отправил сигнал (логическую единицу) на первый вход i–того логического вентиля блока 3, и на второй вход данного логического вентиля блока 3 пришёл сигнал (логическая единица) из блока 1 о распознанном слове β_i. Соответсвующий i–ый логический вентиль "конъюнкция" посылает единичный сигнал на единственный соответсвующий себе i–ый однополюсный однопозиционный переключатель и переводит его в разомкнутое состояние. С этого такта на i–ый выход блока 3 i–ый однополюсный однопозиционный переключатель посылает логическую единицу.

Логическая единица на выходе "дизъюнкции" от всех "конъюнкций" сигнализирует единицей о распознанном слове из языка $\bigcup_{i=1}^{n} .*\alpha_i.*\beta_i.*$ и переводит соответсвующий себе $n + 1$–ый переключатель в разомкнутое состояние. С этого такта $n + 1$ выход блока 3 сигнализирует единицей о распознанном слове из языка $\bigcup_{i=1}^{n} . * \alpha_i . * \beta_i.*$, а соответствующая i конъюнкция с своим переключателем — об определенном i–ом принятом слове, соответствующему регулярному выражению $. * \alpha_i . * \beta_i.*$.

Описанная конструкция распознает соответствующий язык и позволяет избежать экспоненциального взрыва числа состояний ДКА, что перспективно для реализации на практике для сканирования трафика на вредоносность. В силу своей простоты и малого объема диаграмма автомата блока 1 может быть записана в память, а конфигурация блока 2 — задана начальными состояниями задержек. Таким образом, можно реализовывать схемы для классов языков, ограниченных параметрами n и m.

4. Основные результаты

Теорема 1 (О корректности работы)**.** *Представленная конструкция корректно принимает язык* $\bigcup_{i=1}^{n} .* \alpha_i .* \beta_i .*$*, где* α_i*,* β_i *— слова над алфавитом* A*. При этом* i*–ый выход конструкции равен* 1 *тогда и только тогда, когда входное слово лежит в* i*–ом слагаемом, а* $n+1$*–ый выход равен* 1 *тогда если и только если слово лежит в языке.*

Доказательство. Предположим, что слово $\gamma \in A^*$ принадлежит языку $.* \alpha_i .* \beta_i .*$ для некоторого i. Значит, γ имеет вид $\delta\alpha_i\delta'\beta_i\delta''$, где δ не содержит подслово α_i, а δ' — подслово β_i. Значит, после подачи на вход $\delta\alpha_i$ будет запущен счетчик номер i, и к моменту окончания подачи $\delta\alpha_i\delta'\beta_i$, во-первых, iый выход блока 2 станет равным единице, и, во-вторых, $2n+i$ый выход блока 1 станет равным 1. Следовательно, на выход iой конъюнкции блока 3 также появится единица, которая будет сохранена iым переключателем.

Обратно, пусть выход номер i третьего блока равнен единице. Рассмотрим момент, в который значение в первый раз изменилось с 0 на 1. Тогда в этот момент оба входа конъюнкции номер i третьего блока стали равными единице, то есть входное слово имеет вид $\delta''\beta_i$, и не менее $|\beta_i|$ тактов назад слово имело вид $\delta\alpha_i$. Следовательно, входное слово может быть записнано в виде $\delta\alpha_i\delta'\beta_i$, то есть оно принадлежит языку $.* \alpha_i .* \beta_i .*$. После этого момента выход останется равным 1, при этом любое надслово слова $\delta\alpha_i\delta'\beta_i$ также лежит в языке $.* \alpha_i .* \beta_i .*$.

Утверждение про выход номер $n+1$ вытекает из утверждения про первые n выходов.

Теорема доказана. □

Теорема 2 (Об отсутствии экспоненциального взрыва)**.** *Для распознавания языка* $\bigcup_{i=1}^{n} .* \alpha_i .* \beta_i .*$ *представленной конструкцией требуется* $O(mn\log_2(mn)+n)$ *бит памяти для хранения диаграммы блока 1, и* $O(n\log_2 m)$ *элементов для реализации блоков 2 и 3, где* m *— максимальная длина слов* α_i *и* β_i*.*

Доказательство. Сперва рассморим первый блок. Алгоритм Ахо и Корасик строит автомат, число состояний которого оценивается сверху как $2mn$. Для хранения диаграммы переходов требуется $|A| \times |Q|$ ячеек длины $]\log_2|Q|[$. Учитывая, что мощность алфавита является константой, эта величина имеет вид $O(mn\log_2(mn))$. Выход имеет размерность $2n$; следовательно, для хранения функции выходов достаточно $|A| \times |Q|$ ячеек длины n, то есть $O(mn \cdot n)$.

Рассмотрим второй блок. Он состоит из линейного по n числа триггеров и счётчиков. Триггер имеет константную сложность, счётчик — логарифмическую по m. В результате получаем сложность $O(n \log_2 m)$.

Наконец, третий блок очевидным образом линеен по n и константен по m.

Теорема доказана. □

Замечание 1. *Конструкция может быть естественным образом обобщена на классы языков, в которых число слов-сомножителей больше, чем 2.*

Замечание 2. *При изменении распознаваемого языка не обязательно перестраивать автомат; одно устройство способно обрабатывать произвольный язык из рассматриваемого класса при условии, что число слагаемых не превышает n, диаграмма автомата блока 1 помещается в память, а длины слов-сомножителей — в счётчики. Перенастройка производится за счет записи в память новой диаграммы и выставления соответствующих начальных состояний на счётчиках.*

5. Некоторые примеры

Рассмотрим язык $\bigcup_{i=1}^{n} . * a_i\{l\}. * a_i\{l\}$, где $a_i \in A$, l — фиксированная длина слов. ДКА для данного языка требует $O(nl \cdot 2^n)$ состояний [7]. Представленная в данной статье аппаратная конструкция требует ln состояний ДКА блока 1, $nl \cdot \log_2 2nl$ бит на таблицу переходов в памяти, n счётчиков в блоке 2 и $O(n)$ элементов блока 3.

6. Заключение

В данной статье рассмотрен класс языков с высокой практической значимостью, вызывающий экспоненциальный взрыв распознающего ДКА. Для этого класса языков предложена конструкция — комбинация абстрактного и структурного автомата, решающая проблему взрыва сложности распознающего устройства. Доказаны утверждения о корректности работы представленной конструкции и об отсутствии экспоненциального взрыва числа элементов данной конструкции.

Далее планируется расширить конструкцию с целью возможности ухода от проблемы экспоненциального взрыва для распознавания максимально широкого класса практически интересных языков.

Список литературы

[1] Кудрявцев В. Б., Алешин С. И., Подколзин А. С., *Введение в теорию автоматов*, Наука, Москва, 1985.

[2] Кудрявцев В. Б., Гасанов Э. Э., Подколзин А. С., *Основы теории интеллектуальных систем*, Макс Пресс, Москва, 2016.

[3] Документация для Perl-совместимых регулярных выражений, http://perldoc.perl.org/perlre.html.

[4] Хопкрофт Дж., Мотвани Р., Ульман Дж., *Введение в теорию автоматов, языков и вычислений*, Вильямс, Москва, 2017.

[5] Rabin M.O., Scott D., "Finite automata and their decision problems", *IBM J. Research and Development*, **3**:2 (1959), 115–125.

[6] Лупанов О.Б., "О сравнении двух типов конечных источников", Проблемы кибернетики, 1963, № 9, 321–326.

[7] Александров Д. Е., "Эффективные методы проверки сетевых пакетов", *Интеллектуальные системы. Теория и приложения*, **18**:1 (2014), 37–59.

[8] База регулярных выражений, http://www.snort.org/.

[9] Kumar, *A Thesis on Accelerationof Network Processing Algorithms*, Doctor of Science Thesis, Washington University, Saint Louis, Missouri, 2008.

[10] Александров Д. Е., "Об оценках автоматной сложности распознавания класса регулярных языков", *Интеллектуальные системы. Теория и приложения*, **18**:4 (2014), 121–146.

[11] Александров Д. Е., "Об уменьшении автоматной сложности за счет расширения регулярных языков", *Программная инженерия*, 2014, № 11, 26–34.

[12] Александров Д. Е., "Об оценках мощности некоторых классов регулярных языков", *Дискретная математика*, **27**:2 (2015), 3–21.

[13] Aho A.V., Corasick M.J., "Efficient string matching: an aid to bibliographic search", Communication of the ACM, **18**:6 (1975), 333-340.

Structural automaton design for solving the problem of exponential blowup for one class of regular languages

Bernadotte A., Galatenko A.

Аннотация

It is well known that recognition of a language specified by a regular expression of the form $\bigcup_{i=1}^{n} . * \alpha_i . * \beta_i .*$, where α_i, β_i are words over some alphabet, in the general case requires a deterministic automaton with the number of states which is exponential in n. We propose a design of a structural automaton of a polynomial spatial complexity that recognizes languages from a given class.

Keywords: DFA, structural automaton, regular language, exponential blowup.

О поиске натурального классического логического вывода с использованием частичной скулемизации

Охотников О.А.

Рассматривается метод поиска вывода в односукцедентном секвенциальном варианте классического исчисления предикатов. В этом алгоритме используются метапеременные и частичная скулемизация. Для рассматриваемого алгоритма доказываются теоремы о корректности и полноте.

Ключевые слова: автоматическое доказательство теорем, поиск вывода, язык первого порядка, исчисление предикатов, исчисление секвенций, продукционная система, искусственный интеллект.

Введение

В работе [1] сформулирован алгоритм автоматического доказательства теорем, в котором впервые была использована частичная скулемизация для решения задачи поиска натурального вывода. Раньше скулемизация использовалась только в алгоритмах установления выводимости [2], но не поиска вывода. Практика использования этого алгоритма показала, что его эффективность не уступает существующим аналогам [3], [4]. После статьи [1] до сих пор не было публикаций, посвященных этому алгоритму. В данной работе решается задача теоретического обоснования этого алгоритма.

Для достижения этой цели мы формулируем указанный алгоритм в виде продукционной системы дедуктивных задач с частичной скулемизацией. Это означает, что указываются примитивные задачи и правила сведения задач к подзадачам. При этом процесс поиска решения дедуктивной задачи сопряжен с построением дерева поиска типа И/ИЛИ. Такая формулировка алгоритма нам представляется наиболее удобной для построения этой теории.

В данной работе доказываются теоремы о корректности и полноте формулируемой продукционной системы со скулемизацией. Эти две теоремы обеспечивают для всякой выводимой в исчислении секвенции возможность извлечения ее логического вывода из деривационного поддерева поискового дерева. Тем самым устанавливается взаимосвязь между поиском решения задач в продукционной системе со скулемизацией и поиском вывода в классическом односукцедентном исчислении секвенций [5]. Доказательство корректности содержит в себе эффективный алгоритм получения вывода в односукцедентном исчислении секвенций из деривационного поддерева поискового дерева. В то же время существует эффективный алгоритм получения натурального вывода из односукцедентного секвенциального вывода. Все это вместе дает точную формулировку и обоснование новой процедуры поиска натурального вывода [1].

Статья состоит из введения, пяти разделов и списка использованной литературы. В первом разделе определяются используемые в этой статье основные понятия и вводятся обозначения.

Во втором разделе описывается односукцедентное секвенциальное классическое исчисление предикатов, для которого в статье решается задача поиска вывода. Для этого формулируется сопряженная исчислению продукционная система с метапеременными. Здесь же формулируются теоремы о корректности и полноте этой продукционной системы по отношению к исчислению. Указанная продукционная система служит эталоном для установления корректности и полноты продукционной системы со скулемизацией.

В третьем разделе формулируется продукционная система со скулемизацией. Для этого для псевдоформул определяется скулемовская нормальная форма, которая используется в формулировке дедуктивных задач. Такая скулемовская нормальная форма позволяет эффективно решать задачу поиска логического вывода в классическом односукцедентном исчислении секвенций.

В четвертом разделе доказывается теорема о корректности продукционной системы со скулемизацией. Полнота этого метода поиска вывода устанавливается в теореме о полноте, которая доказывается в пятом разделе.

1. Основные понятия и обозначения

В краткой форме введем используемую в статье терминологию. Мы следуем определениям и обозначениям [5]. Кроме того, нам необходимо понятие продукционной системы. Продукционная система представляет собой способ решения задач в отдельной области путем указания примитивных задач и правил сведения задач к подзадачам [6]. Для этого фиксируется конечный алфавит A и разрешимое подмножество $B \subseteq A^*$ слов в этом алфавите, элементы которого кодируют задачи в рассматриваемой области. Среди задач выделяется разрешимое подмножество $C \subseteq B$ примитивных задач, решение которых известно. Кроме того, задается конечное количество правил сведения задач к подзадачам, которые представляют собой разрешимые отношения на B: $R_1 \subseteq B^{n_1}$, ..., $R_k \subseteq B^{n_k}$. В рамках продукционной системы процесс поиска решения задачи сопряжен с формированием дерева поиска типа И/ИЛИ. Ниже в этом разделе мы в краткой форме уточняем, каким образом формируется такое дерево поиска.

Поисковое дерево типа И/ИЛИ представляет собой дерево редукций исходной задачи, которая размещается в корне [6]. Между ребрами дерева существуют “И” и “ИЛИ” отношения. В общем случае из вершины могут выходить несколько ИЛИ-альтернативных И-связок ребер. Ребра из одной связки связываются И-отношением, а ребра из разных связок — ИЛИ-отношением. Вершина, из которой выходят только ребра, связанные И-отношением, называется И-вершиной. Вершина, из которой выходят только ребра, связанные ИЛИ-отношением, называется ИЛИ-вершиной. Всякое И/ИЛИ-дерево можно привести к виду, когда каждая вершина имеет дочерние вершины либо только типа “И”, либо только типа “ИЛИ” [6]. Поддерево Θ поискового И/ИЛИ-дерева T называется деривационным, если Θ обладает следующими свойствами:

- исходная задача — это корень дерева Θ;
- если задача S является ИЛИ-вершиной, то в Θ содержится только одна из ее дочерних вершин из T вместе со своим собственным деривационным деревом;
- если задача S является И-вершиной, то все ее дочерние вершины из T вместе со своими деривационными деревьями содержатся в Θ.

Решением исходной задачи считается деривационное поддерево Θ поискового И/ИЛИ-дерева T. Дерево Θ выступает И-деревом, т. к. все его вершины являются вершинами типа И.

По сути продукционная система представляет собой формулировку некоторого алгоритма путем указания примитивных задач и правил декомпозиции задач с точностью до стратегии построения дерева поиска. Зафиксировав конкретную стратегию построения поискового дерева мы получим некоторую реализацию так сформулированного алгоритма. Стратегия построения дерева поиска заключается в методике выбора листа для формирования у него всех дочерних связок вершин. При этом родительская вершина и дочерние вершины из одной связки связаны одним из отношений декомпозиции R_i. Если у листа нельзя породить дочерние вершины, то ему приписывается оценка Yes или No: Yes, если лист представляет собой примитивную задачу из C; оценка No, если ни одно из правил декомпозиции к нему не применимо. Формирование оценки у листа приводит к пересчету всех оценок вершин на ветви, ведущей от листа к корню. При этом И-вершина получает оценку Yes, если все ее дочерние задачи уже имеют оценку Yes; оценка No формируется, если хотя бы одна дочерняя уже имеет оценку No. Если же задача является ИЛИ-вершиной, то она получает оценку Yes, когда хотя бы одна дочерняя уже имеет оценку Yes; оценка же No формируется, когда каждой дочерней приписана оценка No. В остальных случаях у родительской вершины не формируется оценка. Отсутствие оценки у задачи говорит о том, что возможность ее решения еще пока сохраняется. Процесс построения дерева поиска завершается как только корень получает оценку Yes или No. В первом случае для исходной задачи в дереве поиска существует конечное деривационное поддерево. Такая задача называется *разрешимой* в рамках продукционной системы.

В классе всех продукционных систем естественным образом выделяется собственный подкласс тех продукционных систем, которые сопряжены обычным исчислениям, предназначенным для вывода цепочек символов. В такой сопряженной исчислению $\mathcal{C}$ продукционной системе $\mathcal{P}_\mathcal{C}$ примитивными задачами считаются аксиомы исчисления $\mathcal{C}$, а способы сведения задач к подзадачам получаются в результате применения правил вывода исчисления $\mathcal{C}$ снизу вверх, т. е. от заключения к посылкам [7]. При этом далеко не всегда продукционная система сопряжена некоторому исчислению. Дело в том, что порядок формируемых подзадач в отдельной связке может играть принципиальную роль, в то время как порядок посылок в правиле вывода роли не играет.

Имея язык заданной сигнатуры Ω в области автоматического доказательства теорем рассматривают множество H_Ω всех термов, которые можно построить из констант и функциональных символов сигнатуры Ω. Множество H_Ω называют эрбрановским универсумом [2]. При этом термы эрбрановского универсума могут содержать предметные переменные, которые в таком контексте рассматриваются как константы.

С процессом построения дерева поиска $T_0 \subseteq T_1 \subseteq T_2 \ldots$ связан процесс расширения исходного языка. А именно, когда в процедуре построения дерева поиска у листа S_n порождаются дочерние вершины, происходит обогащение сигнатуры Ω_n языка. В результате формируется новая сигнатура Ω_{n+1}, получаемая из Ω_n добавлением новых свободных предметных переменных, входящих в Ω_{n+1} как константы. При этом в том случае, когда получаемая из исходной сигнатуры Ω сигнатура Ω_0 не содержит констант, мы искусственно добавляем в Ω_0 новую предметную переменную, играющую роль константы. В результате в общем случае в дереве поиска получаем бесконечную иерархию языков и универсумов

$$\begin{array}{rc} & T_0 \subseteq T_1 \subseteq \ldots \subseteq T_n \subseteq T_{n+1} \subseteq \ldots \\ \Omega \subseteq & \Omega_0 \subseteq \Omega_1 \subseteq \ldots \subseteq \Omega_n \subseteq \Omega_{n+1} \subseteq \ldots \\ & H_0 \subseteq H_1 \subseteq \ldots \subseteq H_n \subseteq H_{n+1} \subseteq \ldots \end{array}$$

Необходимо отметить, что сопряженные тому или иному варианту исчисления предикатов продукционные системы формируют бесконечно ветвящиеся деревья поиска типа И/ИЛИ. В прикладной программе такие деревья приводят к комбинаторному взрыву. Чтобы сделать деревья поиска конечно ветвящимися используются метапеременные [3]. При этом под метапеременной понимается синтаксическая переменная, принимающая в качестве своих значений термы языка первого порядка. Мы не будем формулировать метод метапеременных в общем виде для широкого класса продукционных систем. В построенной теории ниже формулируются две конкретные продукционные системы с метапеременными, которые решают тот же объем дедуктивных задач, что и сопряженная логико-математическому исчислению продукционная система.

Традиционно методы автоматического доказательства теорем подразделяются на две категории: это алгоритмы установления выводимости и алгоритмы поиска вывода. Например, метод резолюций [2] — это алгоритм установления выводимости, а процедура [3] — это алгоритм поиска вывода. Каждый алгоритм поиска вывода есть алгоритм установления выводимости, но не наоборот. Так резолюционный вывод нельзя полиномиальным способом преобразовать в натуральный вывод. Ис-

пользуемые в методе резолюций метапеременные можно переименовать в рамках отдельного дизъюнкта — это локальные метапеременные. В процедуре Правица [2] и процедуре [3] метапеременные глобальные. Их значения вычисляются в ходе вычислительного процесса.

Использование метапеременных приводит к изменению языка. Теперь выражения языка — это псевдоформулы и псевдотермы, которые могут быть обычными формулами и термами, но могут и содержать метапеременные, тем самым отличаясь от последних. Полученные в результате применения правила декомпозиции дочерние задачи, кроме глобальных метапеременных родительской задачи, могут содержать новые глобальные метапеременные, которым приписывается в качестве диапазона значений тот эрбрановский универсум, который построен к текущему моменту. В поисковом дереве с метапеременными у вершин вместо оценок формируются конечные множества допустимых подстановок, которые указывают на какие термы должны быть заменены глобальные метапеременные. Метод вычисления допустимых подстановок в разных продукционных системах с метапеременными определяется по разному. Мы в этой статье используем метод [3]. Точные определения связанных с этим методом понятий будут даны в разделе 2. Процесс построения дерева поиска завершается, когда набор допустимых подстановок у корня становится непустым. Соответствующее допустимой подстановке $\theta = \begin{pmatrix} X_1 & \dots & X_r \\ t_1 & \dots & t_r \end{pmatrix}$ деривационное поддерево Θ поискового дерева таково, что $\Theta\theta$ есть вывод в логико-математическом исчислении. При этом если диапазоном значений метапеременной X_i является H_n, то $t_i \in H_n$.

Использование целеориентированного поиска позволяет некоторым разумным образом оптимизировать порядок применения правил декомпозиции [5]. Тем не менее применение кванторных правил приводит к появлению катастрофически огромного числа вариантов при переборе всех возможных альтернатив [8]. По нашему мнению дальнейшей оптимизации такого перебора можно достичь за счет использования скулемовских нормальных форм определенного вида. Предлагаемая в статье скулемовская нормальная форма позволяет элиминировать некоторые кванторы и тем самым сделать поиск более эффективным. При этом используется такая форма частичной скулемизации, которая позволяет не терять связь с искомым выводом в односукцедентном исчислении секвенций.

2. Односукцедентное исчисление секвенций

В этом разделе мы, следуя [5], опишем односукцедентное секвенциальное классическое исчисление предикатов, для которого в статье решается задача поиска вывода. Для этого здесь же будет определена сопряженная этому исчислению продукционная система. Исчисление секвенций выгодно отличается от системы натуральной дедукции [1] с точки зрения формулировки методов поиска вывода и их обоснования. При этом существует эффективный алгоритм получения натурального вывода из односукцедентного секвенциального вывода.

Основные определения и обозначения, связанные с исчислением секвенций, можно найти в статье [5]. В этой статье определены три односукцедентных секвенциальных исчисления предикатов: минимальное $\mathcal{S}_1$, интуиционистское $\mathcal{S}_2$ и классическое $\mathcal{S}_3$. Необходимо обратить внимание на то, что в статье [5] при определении правил вывода $(\supset\!\rightarrow)$, $(\&\ \rightarrow)$, $(\vee\ \rightarrow)$, $(\forall\ \rightarrow)$ и $(\exists\ \rightarrow)$ допущена досадная оплошность. В этих правилах формула G может быть не только атомом, дизъюнкцией или $\bot$, но также формулой вида $\exists x\, C(x)$. Кроме этого, мы здесь используем другую, но эквивалентную, формулировку правила вывода $(\supset\!\rightarrow)$, которая оказалась более удобной для изложения этой теории.

Для формулировки исчисления секвенций $\mathcal{S}_3$ прежде всего мы должны фиксировать некоторый логико-математический язык Ω, формулы которого выражают суждения и отношения рассматриваемой области математики. Мы предполагаем, что сложные формулы языка Ω строятся из атомарных формул этого языка при помощи логических связок $\&$, $\vee$, $\supset$, логических констант $\bot$, $\top$ и кванторов $\forall$, $\exists$. Отрицание $\neg A$ есть по определению формула $A \supset \bot$. Вхождения предметных переменных в формулу обычным образом делятся на свободные и связанные. Переменные, входящие в формулу свободно, называются ее параметрами. Через $A(x/t)$ обозначается результат подстановки терма t вместо всех свободных вхождений переменной x в формулу A. При этом предполагается, что производится переименование связанных переменных формулы A, если параметры t попадают в область действия кванторов A. Вместо $A(x/t)$ мы будем писать просто $A(t)$, когда контекст x очевиден.

Секвенцией [5] называется упорядоченная пара вида $\Gamma \rightarrow A$, где Γ есть конечное, в частности быть может пустое, множество формул языка Ω, а A есть формула этого языка. При этом Γ называется антецедентом, а формула A — сукцедентом секвенции. Теоретико-множественные операции над конечными множествами формул рассматриваются обычным

образом. Например, объединение $\Gamma \cup \Delta$ конечных множеств формул Γ и Δ состоит из всех формул из Γ и Δ. Как обычно, мы пишем $\Gamma\Delta$ вместо $\Gamma \cup \Delta$. Так что $\Gamma\Delta$ и $\Delta\Gamma$ обозначают одно и то же.

Исчисление $\mathcal{S}_3$ содержит аксиомы следующих двух видов: $\Gamma C \to C$ и $\Delta \to \top$, где C — атомарная формула языка Ω или $\bot$. Первые десять правил вывода делятся на две группы и вводят логические связки слева и справа. Последнее правило вывода называется правилом противоречия. Правила вывода изображаются в виде двухэтажной конструкции. Над чертой пишутся секвенции, являющиеся посылками правила вывода, которые отделяются точкой с запятой. Под чертой пишется его заключение, которое является выводимой секвенцией, если посылки выводимы. Рядом с чертой мы указываем символическое обозначение правила:

$$\frac{(A\supset B)\,\Gamma\to A;\quad B\,(A\supset B)\,\Gamma\to G}{(A\supset B)\,\Gamma\to G}\ (\supset\to), \qquad \frac{\Gamma\,A\to B}{\Gamma\to A\supset B}\ (\to\supset),$$

$$\frac{A\,B\,(A\,\&\,B)\,\Gamma\to G}{(A\,\&\,B)\,\Gamma\to G}\ (\&\to), \qquad \frac{\Gamma\to A;\quad \Gamma\to B}{\Gamma\to A\,\&\,B}\ (\to\&),$$

$$\frac{A\,(A\vee B)\,\Gamma\to G;\quad B\,(A\vee B)\,\Gamma\to G}{(A\vee B)\,\Gamma\to G}\ (\vee\to), \qquad \frac{\Gamma\to A_i}{\Gamma\to A_1\vee A_2}\ (\to\vee_i),$$

$$\frac{A(x/t)\,(\forall x\,A(x))\,\Gamma\to G}{(\forall x\,A(x))\,\Gamma\to G}\ (\forall\to), \qquad \frac{\Gamma\to A(x/y)}{\Gamma\to\forall x\,A(x)}\ (\to\forall),$$

$$\frac{A(x/y)\,(\exists x\,A(x))\,\Gamma\to G}{(\exists x\,A(x))\,\Gamma\to G}\ (\exists\to), \qquad \frac{\Gamma\to A(x/t)}{\Gamma\to\exists x\,A(x)}\ (\to\exists),$$

$$\frac{(\neg G)\,\Gamma\to\bot}{\Gamma\to G}\ (\bot_{\mathbf{c}}).$$

Здесь в правилах G — атом, дизъюнкция, $\bot$ или формула вида $\exists z\,C(z)$ языка Ω, A, B — любые формулы языка Ω, а Γ — любое конечное множество формул языка Ω. В кванторных правилах терм t и переменная y допустимы для x в $A(x)$. В кванторных правилах $(\to\forall)$ и $(\exists\to)$ переменная y не входит свободно в нижнюю секвенцию. Описанные условия называются ограничениями кванторных правил.

Выводом секвенции S в исчислении $\mathcal{S}_3$ называется конечная последовательность секвенций $S_1, \ldots, S_k$, в которой $S = S_k$ и каждая S_j либо является аксиомой исчисления $\mathcal{S}_3$, либо получается в результате применения некоторого правила вывода исчисления $\mathcal{S}_3$ из некоторых предшествующих секвенций в этой последовательности. Вывод в $\mathcal{S}_3$ удобно оформлять в виде дерева, у которого к каждому узлу приписана секвенция. При этом аксиомы приписаны к листьям этого дерева, а от-

ношение между родительским узлом и дочерними узлами соответствует отношению между секвенцией-заключением и секвенциями-посылками в результате применения некоторого правила вывода. Высота вывода есть по определению количество секвенций в самой длинной ветви. Мы не будем делать различия между формулами и секвенциями, отличающимися лишь переименованием связанных переменных. Таким образом, если имеется вывод с нижней секвенцией S, то этот же вывод считается автоматически и выводом любой секвенции, полученной из S переименованием связанных переменых.

После сделанных определений теперь точным образом опишем сопряженную исчислению $\mathcal{S}_3$ продукционную систему $\mathcal{P}_3$ с метапеременными. В ней выражениями языка являются псевдотермы и псевдоформулы. *Псевдотермы* какого-либо языка Ω определяются индуктивно. Прежде всего, предметные константы языка Ω, предметные переменные и метапеременные считаются псевдотермами. Если f есть n-местный функциональный символ языка Ω и t_1, …, t_n суть псевдотермы языка Ω, то цепочка вида $f(t_1, \ldots, t_n)$ есть псевдотерм. Аналогичное определение *псевдоформул* языка Ω выглядит следующим образом. Прежде всего, $\bot$ и $\top$ суть псевдоформулы. Если P есть n-местный предикатный символ и t_1, …, t_n суть псевдотермы языка Ω, то цепочка $P(t_1, \ldots, t_n)$ есть псевдоформула. Если A и B суть псевдоформулы, то $(A \,\&\, B)$, $(A \vee B)$, $(A \supset B)$ суть псевдоформулы. Если A есть псевдоформула и x есть предметная переменная, то $(\forall x\, A)$ и $(\exists x\, A)$ суть псевдоформулы.

Задачами этой системы являются упорядоченные пары вида $(\Gamma,\, A)$, где Γ есть конечное множество псевдоформул и A есть псевдоформула. Примитивными задачами являются задачи следующих двух видов: $(E\Gamma,\, E)$, $(\Delta,\, \top)$, где E — атом или $\bot$. Содержащая метапеременные X_1, …, X_k дедуктивная задача π *примитивизируема*, если существует корректная подстановка $\theta = \begin{pmatrix} X_1 & \ldots & X_k \\ t_1 & \ldots & t_k \end{pmatrix}$ вместо метапеременных X_1, …, X_k такая, что дедуктивная задача $\pi\theta$ является примитивной. При этом *корректность* θ означает, что если диапазоном значений метапеременной X_i является H_n, то t_i есть псевдотерм языка Ω_n. Корректная подстановка θ называется *основной*, если $t_i \in H_n$.

Для унифицируемых выражений E_1 и E_2 через $\mathbf{MGU}(E_1,\, E_2)$ ниже обозначается их наиболее общий унификатор [3], т. е. это такой унификатор θ выражений E_1 и E_2, что для каждого их унификатора λ выполняется $\lambda \leq \theta$. При этом $\lambda \leq \theta$ означает существование подстановки μ

такой, что $\lambda = \theta \circ \mu$. Через $\theta \circ \mu$ обозначается композиция подстановок [2].

Для вычисления допустимых подстановок у задач, имеющих в дереве поиска дочерние задачи, требуется понятие комбинации подстановок [2], которое определяется следующим образом. Пусть

$$\theta_1 = \begin{pmatrix} X_{11}, & \dots & X_{1n_1} \\ t_{11}, & \dots & t_{1n_1} \end{pmatrix},$$
$$\vdots$$
$$\theta_r = \begin{pmatrix} X_{r1}, & \dots & X_{rn_r} \\ t_{r1}, & \dots & t_{rn_r} \end{pmatrix}$$

суть подстановки ($r \geq 2$). Исходя из $\theta_1, \dots, \theta_r$, сформируем два выражения

$$E_1 = (X_{11}, \dots, X_{1n_1}, \dots, X_{r1}, \dots, X_{rn_r}),$$
$$E_2 = (t_{11}, \dots, t_{1n_1}, \dots, t_{r1}, \dots, t_{rn_r}).$$

Тогда подстановки $\theta_1, \dots, \theta_r$ называются *совместимыми*, если выражения E_1 и E_2 унифицируемы. При этом подстановка $\mathbf{MGU}(E_1, E_2)$ называется *комбинацией* подстановок $\theta_1, \dots, \theta_r$.

Формулируемые ниже правила 1 – 12 определяют правила декомпозиции $\mathcal{P}_3$.

1. Пусть E и E' — унифицируемые атомы или $\bot$. Пусть подстановка θ есть $\mathbf{MGU}(E', E)$. Тогда считается разрешимой задача $(E\,\Gamma, E')$ и подстановка θ считается ее допустимой подстановкой.

2. Пусть псевдоформула C есть атом, дизъюнкция, $\bot$ или имеет вид $\exists x\, D(x)$. Тогда если $((A \supset B)\Gamma, A)$ и $(B\,(A \supset B)\,\Gamma, C)$ суть разрешимые задачи, у которых существуют совместимые допустимые подстановки θ_1 и θ_2, то также считается разрешимой задача $((A \supset B)\Gamma, C)$ и ее допустимой подстановкой считается комбинация θ_1 и θ_2.

3. Если разрешима задача $(A\,\Gamma, B)$ и θ — ее допустимая подстановка, то считается разрешимой задача $(\Gamma, A \supset B)$ и θ считается ее допустимой подстановкой.

4. Пусть псевдоформула C есть атом, дизъюнкция, $\bot$ или имеет вид $\exists x\, D(x)$. Тогда если разрешима задача $(AB(A\,\&\,B)\Gamma, C)$ и θ — ее допустимая подстановка, то задача $((A\,\&\,B)\Gamma, C)$ также считается разрешимой и θ считается ее допустимой подстановкой.

5. Если (Γ, A) и (Γ, B) суть две разрешимые задачи, у которых существуют совместимые допустимые подстановки θ_1 и θ_2, то тогда разреши-

ма задача (Γ, $A\&B$) и ее допустимой подстановкой является комбинация θ_1 и θ_2.

6. Пусть псевдоформула C есть атом, дизъюнкция, $\bot$ или имеет вид $\exists x\, D(x)$. Тогда если $(A\,(A \vee B)\Gamma,\, C)$ и $(B\,(A \vee B)\Gamma,\, C)$ суть две разрешимые дочерние задачи, у которых существуют совместимые допустимые подстановки θ_1 и θ_2, то разрешима также задача $((A \vee B)\Gamma,\, C)$ и ее допустимой подстановкой считается комбинация θ_1 и θ_2.

7. Если для $i = 1$ или $i = 2$ разрешима задача $(\Gamma,\, A_i)$ и θ — ее допустимая подстановка, то задача $(\Gamma,\, A_1 \vee A_2)$ также считается разрешимой и θ считается ее допустимой подстановкой.

8. Пусть псевдоформула C есть атом, дизъюнкция, $\bot$ или имеет вид $\exists x\, D(x)$. Тогда задача $((\forall x\, A(x))\,\Gamma,\, C)$ может быть сведена к новой задаче $(A(X)\,(\forall x\, A(x))\,\Gamma,\, C)$, где X — новая глобальная метапеременная в поисковом дереве. При этом если дочерняя задача $(A(X)\,(\forall x\, A(x))\,\Gamma,\, C)$ разрешима и θ — ее допустимая подстановка, то считается разрешимой родительская задача $((\forall x\, A(x))\,\Gamma,\, C)$ и θ считается ее допустимой подстановкой.

9. Если разрешима задача $(\Gamma,\, A(y))$, где псевдоформулы из Γ не содержат свободно переменной y, а θ — ее допустимая подстановка, то разрешима задача $(\Gamma,\, \forall x\, A(x))$ и θ — ее допустимая подстановка.

10. Пусть псевдоформула C есть атом, дизъюнкция, $\bot$ или имеет вид $\exists x\, D(x)$. Тогда если дочерняя задача $(A(y)\,(\exists x\, A(x))\,\Gamma,\, C)$ разрешима, где псевдоформулы из Γ не содержат свободно переменной y, а θ — ее допустимая подстановка, то задача $((\exists x\, A(x))\,\Gamma,\, C)$ разрешима и θ — ее допустимая подстановка.

11. Задача $(\Gamma,\, \exists x\, A(x))$ может быть сведена к задаче $(\Gamma,\, A(X))$, где X — новая глобальная метапеременная в поисковом дереве. При этом если дочерняя задача $(\Gamma,\, A(X))$ разрешима и θ — ее допустимая подстановка, то считается разрешимой родительская задача $(\Gamma,\, \exists x\, A(x))$ и θ считается ее допустимой подстановкой.

12. Пусть псевдоформула C есть атом, дизъюнкция или имеет вид $\exists x\, D(x)$. Тогда если разрешима задача $((\neg C)\,\Gamma,\, \bot)$ и θ — ее допустимая подстановка, то задача $(\Gamma,\, C)$ также считается разрешимой и θ считается ее допустимой подстановкой.

Формулировка продукционной системы $\mathcal{P}_3$ завершена. Продукционная система $\mathcal{P}_3$ задает алгоритм поиска решения дедуктивной задачи не уточняя стратегию построения дерева поиска типа И/ИЛИ. Зафиксировав конкретную стратегию мы получим некоторую реализацию так сформулированного алгоритма. Стратегия построения дерева поиска заклю-

чается в методике выбора листа для формирования у него, во-первых, множества всех примитивизирующих этот лист подстановок, во-вторых, всех дочерних связок вершин, соответствующих правилам декомпозиции. Затем каждая примитивизирующая подстановка "поднимается" по ветви, соединяющей лист с корнем, с помощью операции комбинации [3]. Тем самым в ходе процесса построения дерева поиска у его вершин формируются наборы допустимых подстановок. Указанная продукционная система служит эталоном для установления корректности и полноты продукционной системы со скулемизацией, которая будет сформулирована в следующем разделе. В качестве символических названий сформулированных правил декомпозиции 2 – 12 будем соответственно использовать следующие обозначения: $(\supset\rightarrow)$, $(\rightarrow\supset)$, $(\& \rightarrow)$, $(\rightarrow \&)$, $(\vee \rightarrow)$, $(\rightarrow \vee)$, $(\forall \rightarrow)$, $(\rightarrow \forall)$, $(\exists \rightarrow)$, $(\rightarrow \exists)$, $(\bot_{\mathbf{c}})$.

Заметим, что если θ является допустимой подстановкой разрешимой задачи, то ее допустимой по определению считается также всякая подстановка $\lambda \leq \theta$. Если π есть разрешимая задача и θ — ее допустимая подстановка, то индукцией по высоте деривационного дерева устанавливается, что всякая подстановка $\lambda \leq \theta$ является допустимой подстановкой всех потомков этой задачи π в деривационном дереве. Это позволяет по любой допустимой подстановке θ однозначно находить соответствующее деривационное поддерево Θ, которое считается решением задачи. Такое поддерево Θ получается в результате рекурсивного перебора всех альтернативных связок дочерних подзадач и выбора на каждом шаге той связки из них, которая содержит в качестве комбинации подстановку θ.

Деривационное дерево Θ позволяет решать задачу поиска вывода в следующем смысле. Если деривационное поддерево Θ в дереве поиска соответствует допустимой подстановке θ, то индукцией по высоте Θ устанавливается, что $\theta \geq \lambda$ для любой основной подстановки λ такой, что $\Theta\lambda$ есть вывод в исчислении секвенций $\mathcal{S}_3$ [3]. Следствием последнего утверждения является теорема о корректности и полноте для сформулированной продукционной системы $\mathcal{P}_3$ по отношению к исчислению $\mathcal{S}_3$, т. е. секвенция $\Gamma \rightarrow A$ выводима в $\mathcal{S}_3$ тогда и только тогда, когда задача (Γ, A) разрешима в $\mathcal{P}_3$. Указанная система декомпозиции дедуктивных задач часто ставится во главу угла при разработке компьютерных систем автоматизации дедукции [3], [8], [9]. Недостатки $\mathcal{P}_3$ были отмечены во введении и первом разделе.

3. Продукционная система дедуктивных задач со скулемизацией

При решении задач установления выводимости или поиска вывода с целью повышения эффективности этого процесса часто применяют те или иные инвариантные преобразования формул. Преобразование, которое здесь рассматривается, называется *скулемизацией* [2], [10]. Скулемизация позволяет упростить алгоритмическую часть процедуры установления выводимости. Имея в виду контекст метода метапеременных, мы будем заниматься преобразованием псевдоформул. При этом мы будем использовать такой новый вариант частичной скулемизации, который позволяет не терять связь с искомым односукцедентным секвенциальным выводом, и тем самым с натуральным выводом. Это позволяет использовать скулемизацию в рамках процедуры поиска натурального вывода [1].

Положительные и *отрицательные вхождения* псевдоформулы в псевдоформулу определим индукцией по числу логических связок. Вхождение псевдоформулы в себя считается положительным. Если псевдоформула B положительно (отрицательно) входит в псевдоформулу A, то B положительно (отрицательно) входит в псевдоформулы вида: $A \,\&\, A'$, $A' \,\&\, A$, $\forall x\, A$, $A \vee A'$, $A' \vee A$, $\exists x\, A$, $A' \supset A$; и отрицательно (положительно) входит в псевдоформулы вида: $A \supset A'$.

Вхождение псевдоформулы C в псевдоформулу D называется *строго положительным*, если оно является положительным и не находится в антецеденте некоторой импликации в D. Вхождение C в D будем называть *строго положительным аналитическим*, если оно является строго положительным и не находится в области действия некоторой дизъюнкции в D.

Под *знаком вхождения логической связки* в указанную псевдоформулу будем понимать знак того вхождения псевдоформулы в указанную псевдоформулу, главной связкой которого оно является. Под *знаком вхождения предикатного символа* будем понимать знак соответствующего вхождения атома. Будем говорить, что псевдоформула A обладает *свойством чистоты переменных*, если, во-первых, все ее связанные переменные отличны от свободных и, во-вторых, любые два различные вхождения кванторных приставок связывают различные переменные.

Пусть A — произвольная псевдоформула со свойством чистоты переменных и $\exists x$ — некоторая строго положительная аналитическая экзистенциальная приставка в A. *Размерностью* переменной x в A будем

называть число различных кванторов всеобщности, управляющих переменной x. Пусть k — размерность переменной x в A и f — k-местный функциональный символ, не встречающийся в A. Список всех связанных кванторами всеобщности переменных в порядке вхождения в A, управляющих переменной x в A, запишем в виде $z_1, \ldots, z_k$. *Результатом применения* к паре $\langle A, x\rangle$ *скулемовского метода элиминации квантора существования* будем называть псевдоформулу $A^-\begin{pmatrix} x \\ f(z_1, \ldots, z_k) \end{pmatrix}$, где A^- обозначает результат вычеркивания из A единственного вхождения кванторной приставки $\exists x$. Эту псевдоформулу будем обозначать $\mathbf{S}(A, x)$.

Скулемовской нормальной формой псевдоформулы A будем называть псевдоформулу, обозначаемую $\mathbf{Sk}(A)$, которая определяется индукцией по числу логических связок:

- $\mathbf{Sk}(B\&C) = \mathbf{Sk}(B)\,\&\,\mathbf{Sk}(C)$, $\mathbf{Sk}(B \supset C) = B \supset \mathbf{Sk}(C)$, $\mathbf{Sk}(B \vee C) = B \vee C$, $\mathbf{Sk}(\top) = \top$, $\mathbf{Sk}(\bot) = \bot$.
- Если A имеет вид $\exists x\, B$, то $\mathbf{Sk}(A) = B\begin{pmatrix} x \\ c \end{pmatrix}$.
- Если A имеет вид $\forall z\, B$, то процесс вычисления $\mathbf{Sk}(A)$ разбивается на два этапа. На первом этапе после элиминации строго положительных аналитических кванторов существования и введения скулемовских функций получаем псевдоформулу

  $$A' = \mathbf{S}(\ldots(\mathbf{S}(\mathbf{S}(A, x_1), x_2), \ldots), x_l).$$

  На втором этапе после вычеркивания из A' строго положительных аналитических кванторов всеобщности получаем A''. Наконец, после введения новых локальных метапеременных $Z_1, \ldots, Z_m$ получаем искомый результат

  $$\mathbf{Sk}(A) = A''\begin{pmatrix} z_1 & \cdots & z_m \\ Z_1 & \ldots & Z_m \end{pmatrix}.$$

Например, скулемовской нормальной формой формулы

$$\forall x \forall y\,((\exists z\,(P(x, z)\ \&\ P(z, y))) \supset \exists u\, Q(x, u, y))$$

является псевдоформула

$$(\exists z\,(P(X, z)\ \&\ P(z, Y))) \supset Q(X, f(X, Y), Y),$$

где f — новая скулемовская функция, а X и Y — новые локальные метапеременные.

Пусть D есть строго положительная аналитическая подпсевдоформула псевдоформулы C и D есть атом, дизъюнкция, $\bot$ или $\top$. Пусть D находится в области действия последовательности кванторов

$$\forall z_1 \ldots \forall z_{n_1} \exists v_1 \forall z_{n_1+1} \ldots \forall z_{n_2} \exists v_2 \forall z_{n_2+1} \ldots \forall z_{n_k} \exists v_k \forall z_{n_k+1} \ldots \forall z_{n_k+m}. \quad (1)$$

Пусть в процессе скулемизации при элиминации кванторов $\exists v_1, \ldots, \exists v_k$ вводятся новые скулемовские функции $\varphi_1^{n_1}, \ldots, \varphi_k^{n_k}$, а при элиминации кванторов $\forall z_1, \ldots, \forall z_{n_k+m}$ вводятся локальные метапеременные $Z_1, \ldots, Z_{n_k+m}$. Тогда *протоколом скулемизации C относительно D* назовем последовательность псевдоформул $P_1, \ldots, P_q$, которая формируется из частей псевдоформулы C и моделирует процесс скулемизации. Точнее говоря, для псевдоформул $P_1, \ldots, P_q$ имеем $P_1 = C$, $P_q = D\begin{pmatrix} z_1 & \ldots & z_{n_k+m} & v_1 & \ldots & v_k \\ Z_1 & \ldots & Z_{n_k+m} & \varphi_1^{n_1}(Z_1, \ldots, Z_{n_1}) & \ldots & \varphi_k^{n_k}(Z_1, \ldots, Z_{n_k}) \end{pmatrix}$ и

- если $P_j = \forall z_i\, B(z_i)$, то $P_{j+1} = B(Z_i)$;
- если $P_j = \exists v_i\, B(v_i)$, то $P_{j+1} = B(\varphi_i^{n_i}(Z_1, \ldots, Z_{n_i}))$;
- если $P_j = B_1 \supset B_2$, то $P_{j+1} = B_2$;
- если $P_j = B_1 \,\&\, B_2$ и D входит в B_i, то $P_{j+1} = B_i$.

Отношением подобия будем называть рефлексивно транзитивное замыкание отношения между псевдоформулами, когда одна из них получается из другой в результате замены подпсевдоформулы вида $(M \,\&\, N)$ на $(N \,\&\, M)$.

С каждым строго положительным аналитическим вхождением D в C связана подобная C псевдоформула вида

$$\begin{array}{l} Q_{11}x_{11} \ldots Q_{1r_1}x_{1r_1}(C_1\, \lambda_1 \\ \quad Q_{21}x_{21} \ldots Q_{2r_2}x_{2r_2}(C_2\, \lambda_2\, (\ldots \\ \quad\quad \ldots \\ \quad\quad Q_{s1}x_{s1} \ldots Q_{sr_s}x_{sr_s}(C_s\, \lambda_s\, Q_{s+1,1}x_{s+1,1} \ldots Q_{s+1,r_{s+1}}x_{s+1,r_{s+1}}\, D)) \ldots), \end{array} \quad (2)$$

у которой цепочка кванторов

$$Q_{11}x_{11}\ \ldots\ Q_{sr_s}x_{sr_s}\, Q_{s+1,1}x_{s+1,1}\ \ldots\ Q_{s+1,r_{s+1}}x_{s+1,r_{s+1}} \quad (3)$$

совпадает с (1), $\lambda_i \in \{\&, \supset\}$ и протокол скулемизации относительно D совпадает с протоколом скулемизации C относительно D. Будем называть (2) *нормальной формой C относительно D*.

Пусть D есть строго положительная аналитическая подпсевдоформула псевдоформулы C и D есть атом, дизъюнкция, $\bot$ или $\top$. Тогда *пренексной формой* C *относительно* D назовем псевдоформулу, получаемую из C в результате вынесения в начало псевдоформулы всех кванторов, которые управляют D в C.

Очевидно, что если нормальная форма C относительно D имеет вид (2) и C' есть пренексная форма C относительно D, то нормальная форма C' относительно D имеет вид

$$Q_{11}x_{11}\ \ldots\ Q_{s+1,r_{s+1}}x_{s+1,r_{s+1}}\,(C_1\,\lambda_1\,(C_2\,\lambda_2\,(\ldots\,(C_s\,\lambda_s\,D))\ldots). \tag{4}$$

Псевдоформулу (4) будем называть *пренексной нормальной формой* C *относительно* D. Эту псевдоформулу будем обозначать $\mathbf{PNF}(C,\,D)$.

Пусть $F = \mathbf{Sk}(C)$ и нормальная форма C остностительно D имеет вид (2). Тогда результат скулемизации псевдоформулы (2) имеет вид

$$F_1\,\lambda_1\,(F_2\,\lambda_2\,(\ldots\,(F_s\,\lambda_s\,G))\ldots), \tag{5}$$

где $G = D\zeta$, $F_i = \tilde{C}_i\zeta$,

$$\zeta = \left(\begin{array}{cccccc} z_1 & \ldots & z_{n_k+m} & v_1 & \ldots & v_k \\ Z_1 & \ldots & Z_{n_k+m} & \varphi_1^{n_1}(Z_1,\,\ldots,\,Z_{n_1}) & \ldots & \varphi_k^{n_k}(Z_1,\,\ldots,\,Z_{n_k}) \end{array}\right),$$

$$\tilde{C}_i = \left\{\begin{array}{ll} C_i, & \text{если } \lambda_i = \supset; \\ \mathbf{Sk}(C_i), & \text{если } \lambda_i = \&. \end{array}\right.$$

Будем называть (5) *нормальной формой* F *относительно* G. Псевдоформулу (5) будем обозначать $\mathbf{NF}(F,\,G)$. *Импликантой* нормальной формы F относительно G назовем кратную конъюнкцию

$$F_{i_1}\,\&\,F_{i_2}\,\&\,\ldots\,\&\,F_{i_n},$$

где $\lambda_{i_1} = \lambda_{i_2} = \ldots = \lambda_{i_n} =$ "$\supset$" — все импликации среди $\lambda_i \in \{\&,\,\supset\}$.

Дедуктивной задачей назовем пару вида $(\mathbf{Sk}(\Gamma)\,?\,G)$, где Γ — конечное множество псевдоформул, которое называется *антецедентом*, а G — псевдоформула, которая называется *сукцедентом* этой дедуктивной задачи. При этом псевдоформулы из Γ и псевдоформула G не содержат локальных метапеременных, но, возможно, содержат глобальные метапеременные.

Сформулируем теперь продукционную систему $\mathcal{F}_3$ с метапеременными и со скулемизацией, в рамках которой будут решаться дедуктивные

задачи. *Примитивные дедуктивные задачи* этой системы суть произвольные задачи двух видов

$$\Gamma\,?\,\top \quad \text{и} \quad \left(\&_{i=1}^{i=m} A_i\right)\,\Gamma\,?\,G,$$

где G совпадает с некоторой A_i и при этом G — атом или $\bot$.

Чтобы решать дедуктивные задачи теперь определим *правила сведения задач к подзадачам* продукционной системы $\mathcal{F}_3$. Правила построены весьма естественно и удаляют логические связки слева и справа. Правила декомпозиции изображаются в виде двухэтажной конструкции. Под чертой мы пишем родительскую задачу. Над чертой пишем ее дочерние подзадачи. Рядом с чертой мы указываем символическое обозначение правила.

1. Если $\Gamma\,?\,A_1$ и $\Gamma\,?\,A_2$ суть две разрешимые задачи, у которых существуют совместимые допустимые подстановки θ_1 и θ_2, то тогда разрешима задача $\Gamma\,?\,A_1\,\&\,A_2$ и комбинация подстановок θ_1 и θ_2 считается ее допустимой подстановкой. Символически сформулированное правило декомпозиции записывается в виде

$$\frac{\Gamma\,?\,A_1;\quad \Gamma\,?\,A_2}{\Gamma\,?\,A_1\,\&\,A_2}\ (?\&)$$

и понимается так: задача вида $\Gamma\,?\,A_1\,\&\,A_2$ сводится к двум задачам $\Gamma\,?\,A_1$ и $\Gamma\,?\,A_2$.

2. Если разрешима задача $\mathbf{Sk}(A)\,\Gamma\,?\,B$ и θ — ее допустимая подстановка, то считается разрешимой задача $\Gamma\,?\,A\supset B$ и θ считается ее допустимой подстановкой. То есть, задача $\Gamma\,?\,A\supset B$ сводится к новой задаче $\mathbf{Sk}(A)\,\Gamma\,?\,B$:

$$\frac{\mathbf{Sk}(A)\,\Gamma\,?\,B}{\Gamma\,?\,A\supset B}\ (?\supset).$$

3. Если разрешима хотя бы одна из задач $\Gamma\,?\,A_i$, где $i=1$ или $i=2$, а θ — ее допустимая подстановка, то считается разрешимой задача $\Gamma\,?\,A_1\vee A_2$ и θ считается ее допустимой подстановкой. То есть, задача $\Gamma\,?\,A_1\vee A_2$ может быть сведена к подзадаче $\Gamma\,?\,A_i$ (для $i=1$ или 2):

$$\frac{\Gamma\,?\,A_i}{\Gamma\,?\,A_1\vee A_2}\ (?\vee_i).$$

4. Если разрешима задача $\Gamma\,?\,A(y)$, где псевдоформулы из Γ не содержат свободно переменной y, а θ — ее допустимая подстановка, то считается разрешимой задача $\Gamma\,?\,\forall x\,A(x)$ и θ считается ее допустимой подстановкой. Кратко говоря, задача $\Gamma\,?\,\forall x\,A(x)$ сводится подзадаче $\Gamma\,?\,A(y)$,

где y — новая свободная предметная переменная в поисковом дереве:

$$\frac{\Gamma\,?\,A(y)}{\Gamma\,?\,\forall x\,A(x)}\ (?\forall).$$

5. Если разрешима задача $\Gamma\,?\,A(X)$, где X — новая глобальная метапеременная в поисковом дереве, а θ — ее допустимая подстановка, то считается разрешимой задача $\Gamma\,?\,\exists x\,A(x)$ и θ считается ее допустимой подстановкой. То есть, задача $\Gamma\,?\,\exists x\,A(x)$ может быть сведена к дочерней задаче $\Gamma\,?\,A(X)$:

$$\frac{\Gamma\,?\,A(X)}{\Gamma\,?\,\exists x\,A(x)}\ (?\exists).$$

6. Пусть G и G' — унифицируемые атомы или $\bot$, $\mathbf{NF}(A,\,G')$ имеет вид $\&_{i=1}^{i=m}A_i$ и $A_m = G'$. Пусть подстановка η замещает все локальные метапеременные в A новыми глобальными и $\theta = \mathbf{MGU}(G,\,G'\eta)$. Тогда считается разрешимой задача $A\,\Gamma\,?\,G$ и подстановка θ считается ее допустимой подстановкой. Обозначив через B_i псевдоформулу $\&_{j=i+1}^{j=m}A_j$ мы можем записать указанное правило в виде

$$\frac{(\&_{i=1}^{i=m}A_i)\eta\,(A_1\eta)\,(B_1\eta)\,\ldots\,(A_{m-1}\eta)\,(B_{m-1}\eta)\,A\,\Gamma\,?\,G}{A\,\Gamma\,?\,G}\ (\&?).$$

7. Пусть G и G' — унифицируемые атомы или $\bot$ и G' есть строго позитивная аналитическая подпсевдоформула псевдоформулы A. Пусть подстановка η замещает все локальные метапеременные в A новыми глобальными, $\sigma = \mathbf{MGU}(G,\,G'\eta)$ и $\xi = \eta \circ \sigma$. Пусть нормальная форма A относительно G' имеет вид

$$A_1\,\lambda_1\,(A_2\,\lambda_2\,(\ldots(A_n\,\lambda_n\,G'))\ldots),$$

где $\lambda_i \in \{\&,\,\supset\}$. Пусть $A_{i_1}\,\&\,A_{i_2}\,\&\,\ldots\,\&\,A_{i_k}$ — импликанта этой нормальной формы. Для каждого $i = 1,\,\ldots,\,n$ пусть B_i обозначает псевдоформулу

$$A_{i+1}\,\lambda_{i+1}\,(A_{i+2}\,\lambda_{i+2}\,(\ldots(A_n\,\lambda_n\,G))\ldots)$$

и для каждого $j = 1,\,\ldots,\,k$ пусть Δ_j обозначает множество, задаваемое равенствами

$$\Delta_1 = \{\mathbf{NF}(A,\,G'),\,A_1,\,B_1,\,A_2,\,B_2,\,\ldots,\,A_{i_1-1},\,B_{i_1-1}\},$$

$$\Delta_j = \Delta_{j-1} \cup \{B_{i_{(j-1)}},\,\ldots,\,A_{i_j-1},\,B_{i_j-1}\}.$$

Тогда если $\Delta_1\xi\, A\Gamma\,?\, A_{i_1}\xi$, …, $\Delta_k\xi\, A\Gamma\,?\, A_{i_k}\xi$ образуют k разрешимых задач, у которых существуют соответственно допустимые подстановки θ_1, …, θ_k такие, что совместимы подстановки σ, θ_1, …, θ_k, то считается разрешимой также задача $A\,\Gamma\,?\,G$ и комбинация σ, θ_1, …, θ_k считается ее допустимой подстановкой. Символически сформулированное правило сведения задач к подзадачам записывается следующим образом:

$$\frac{\Delta_1\xi\, A\Gamma\,?\, A_{i_1}\xi; \quad \Delta_2\xi\, A\Gamma\,?\, A_{i_2}\xi; \quad \ldots; \quad \Delta_k\xi\, A\Gamma\,?\, A_{i_k}\xi}{A\,\Gamma\,?\,G}\ (\supset?).$$

8. Пусть G есть атом, дизъюнкция, $\bot$ или псевдоформула вида $\exists x\, B$, а также $D_1 \vee D_2$ есть строго позитивная аналитическая подпсевдоформула псевдоформулы A. Пусть нормальная форма A относительно $D_1 \vee D_2$ имеет вид

$$A_1\,\lambda_1\,(A_2\,\lambda_2\,(\ldots(A_n\,\lambda_n\,(D_1 \vee D_2))\ldots),$$

где $\lambda_i \in \{\&, \supset\}$. Пусть $A_{i_1}\,\&\,A_{i_2}\,\&\,\ldots\,\&\,A_{i_k}$ — импликанта этой нормальной формы. Пусть подстановка η замещает все локальные метапеременные в A новыми глобальными. Для каждого $i = 1, \ldots, n$ пусть B_i обозначает псевдоформулу

$$A_{i+1}\,\lambda_{i+1}\,(A_{i+2}\,\lambda_{i+2}\,(\ldots(A_n\,\lambda_n\,(D_1 \vee D_2))\ldots)$$

и пусть множества Δ и Δ_j, где $j = 1, \ldots, k$, задаются равенствами

$$\Delta_1 = \{\mathbf{NF}(A,\, D_1 \vee D_2),\, A_1,\, B_1,\, A_2,\, B_2,\, \ldots,\, A_{i_1-1},\, B_{i_1-1}\},$$

$$\Delta_j = \Delta_{j-1} \cup \{B_{i_{(j-1)}},\, \ldots,\, A_{i_j-1},\, B_{i_j-1}\},$$

$$\Delta = \Delta_k \cup \{B_{i_k},\, \ldots,\, A_n,\, B_n\}, \quad B_n = D_1 \vee D_2.$$

Тогда если $(\Delta_1\eta\, A\Gamma\,?\, A_{i_1}\eta)$, …, $(\Delta_k\eta\, A\Gamma\,?\, A_{i_k}\eta)$, $(\mathbf{Sk}(D_1\eta)\,(\Delta\eta)\, A\Gamma\,?\,G)$, $(\mathbf{Sk}(D_2\eta)\,(\Delta\eta)\, A\Gamma\,?\,G)$ образуют $k+2$ разрешимые задачи, у которых существуют совместимые допустимые подстановки θ_1, …, θ_k, θ_{k+1}, θ_{k+2}, то считается разрешимой также задача $A\,\Gamma\,?\,G$ и ее допустимой подстановкой считается комбинация подстановок θ_1, …, θ_k, θ_{k+1}, θ_{k+2}. Таким образом, задача $A\Gamma\,?\,G$ сводится к следующим $k+2$ подзадачам:

$$\frac{\begin{array}{cc}\Delta_1\eta\, A\Gamma\,?\, A_{i_1}\eta; & \ldots; \quad \Delta_k\eta\, A\Gamma\,?\, A_{i_k}\eta; \\ \mathbf{Sk}(D_1\eta)\,(\Delta\eta)\, A\Gamma\,?\,G; & \mathbf{Sk}(D_2\eta)\,(\Delta\eta)\, A\Gamma\,?\,G\end{array}}{A\,\Gamma\,?\,G}\ (\vee?).$$

9. Наконец, мы имеем правило противоречия ($\bot_{\mathbf{c}}$): если G — атом, дизъюнкция или псевдоформула вида $\exists x\, B$, разрешима задача $(\neg G)\,\Gamma\,?\,\bot$

и θ — ее допустимая подстановка, то разрешима задача $\Gamma\,?\,G$ и θ — ее допустимая подстановка. То есть, задача $\Gamma\,?\,G$ сводится к подзадаче $(\neg G)\,\Gamma\,?\,\bot$:

$$\frac{(\neg G)\,\Gamma\,?\,\bot}{\Gamma\,?\,G}\ (\bot_{\mathbf{c}}).$$

Формулировка продукционной системы $\mathcal{F}_3$ завершена. Будем ее называть *продукционной системой эвристического поиска вывода*. Продукционная система $\mathcal{F}_3$ задает алгоритм поиска решения задачи с точностью до стратегии построения дерева поиска типа И/ИЛИ. Зафиксировав конкретную стратегию мы получим некоторую реализацию так сформулированного алгоритма. Например, алгоритм [1] использует стратегию перебора в глубину.

Заметим, что если θ является допустимой подстановкой разрешимой задачи, то ее допустимой считается также всякая подстановка $\lambda \leq \theta$. Если π есть разрешимая задача и θ — ее допустимая подстановка, то индукцией по высоте деривационного дерева устанавливается, что всякая подстановка $\lambda \leq \theta$ является допустимой подстановкой всех потомков этой задачи π в деривационном дереве.

Очевидно, что рассматриваемые в заданном дереве поиска Σ нормальные формы псевдоформул зависят от дерева Σ: $\mathbf{Sk}_\Sigma(A)$, $\mathbf{NF}_\Sigma(A,\,B)$, $\mathbf{PNF}_\Sigma(A,\,B)$. Тем не менее мы будем в этих обозначениях часто опускать индекс Σ, когда подразумеваемый контекст Σ очевиден.

В следующих двух разделах доказываются теоремы о корректности и полноте для сформулированной продукционной системы $\mathcal{F}_3$ по отношению к исчислению $\mathcal{S}_3$. Тем самым устанавливается взаимосвязь между поиском решения задач в $\mathcal{F}_3$ и поиском вывода в классическом исчислении секвенций $\mathcal{S}_3$. Для этого там устанавливается эквивалентность $\mathcal{F}_3$ и $\mathcal{P}_3$ с точки зрения объема решаемых задач. Использование скулемизации в $\mathcal{F}_3$ приводит к элиминации строго положительных аналитических кванторов в антецедентах дедуктивных задачах. Это позволяет избежать комбинаторного взрыва при применении кванторных правил, возникающего в $\mathcal{P}_3$.

4. Теорема о корректности

Доказываемая в этом разделе теорема о корректности предоставляет эффективный способ извлечения логического вывода из деривационного поддерева поискового дерева разрешимой дедуктивной задачи. Указанная теорема легко доказывается для пропозициональной части продук-

ционной системы эвристического поиска $\mathcal{F}_3$. Распространение этого доказательства на всю $\mathcal{F}_3$ связано с определенными трудностями, которые, тем не менее, удается успешно преодолеть. В доказательстве теоремы о корректности будут использованы два предложения, которые формулируются ниже.

Предложение 4.1. *Пусть D есть строго положительная аналитическая подпсевдоформула псевдоформулы C и C' есть пренексная форма C относительно D или подобная C псевдоформула. Тогда если в $\mathcal{P}_3$ разрешима задача $(C'\,C\,\Gamma,\ A)$ и θ — ее допустимая подстановка, то задача $(C\,\Gamma,\ A)$ также является разрешимой и θ является ее допустимой подстановкой.*

Доказательство может быть получено тривиальной индукцией по высоте деривационного дерева. Это утверждение, а также следующее очевидное следствие, нам потребуются ниже в доказательстве предложения 4.2.

Следствие. *Пусть D есть строго положительная аналитическая подпсевдоформула псевдоформулы C. Тогда если в $\mathcal{P}_3$ разрешима задача $(\mathbf{PNF}(C,\ D)\,C\,\Gamma,\ A)$ и θ — ее допустимая подстановка, то задача $(C\,\Gamma,\ A)$ также является разрешимой и θ является ее допустимой подстановкой.*

Предложение 4.1 и его следствие, по сути, представляют собой формулировки допустимых правил декомпозиции, которые мы будем соответственно символически обозначать (ПФ$\to$) и (ПНФ$\to$). Через $\Theta|S$ будем обозначать поддерево дерева Θ, которое содержит узел S дерева Θ в качестве корня, а также все его потомки. На диаграммах фрагмент $\ddots\vdots\cdot$ будет обозначать часть дерева. Например, фигуры

$$\begin{array}{ccc} & & S_1 \quad S_2 \quad S_3 \\ \ddots\vdots\cdot & \text{и} & \ddots\vdots\cdot \\ S & & S \end{array}$$

обозначают соответственно некоторое дерево с корнем S и некоторое дерево с корнем S и его потомками S_1, S_2, S_3.

Предложение 4.2. *Если задача $(\mathbf{Sk}(\Gamma)\,?\,A)$ разрешима в продукционной системе $\mathcal{F}_3$, то задача $(\Gamma,\ A)$ разрешима в продукционной системе $\mathcal{P}_3$.*

Доказательство. Пусть $\theta = \begin{pmatrix} X_1 & \dots & X_r \\ t_1 & \dots & t_r \end{pmatrix}$ — подстановка из набора допустимых подстановок задачи $\pi_0 = (\mathbf{Sk}(\Gamma)\,?\,A)$. Мы предпола-

гаем, что подстановка θ является основной. В противном случае мы можем заменить "нерешенные" метапеременные на константы из H_0. Пусть Θ — соответствующее этой подстановке θ деривационное дерево. Индукцией по построению деривационного дерева Θ покажем, как в рамках $\mathcal{P}_3$ построить допустимую для задачи $(\Gamma,\, A)$ подстановку $\theta' = \begin{pmatrix} X_1 & \ldots & X_r & \ldots \\ t'_1 & \ldots & t'_r & \ldots \end{pmatrix}$ и соответствующее ей деривационное дерево Θ'. Разобъем процесс построения дерева Θ на этапы. Будем считать, что для формирования дерева Θ_{n+1} из дерева Θ_n на текущем этапе n выбирается некоторый лист построенного поддерева $\Theta_n \subseteq \Theta$ и формируются все его дочерние вершины. Пусть процесс расширения языка при построении дерева Θ имеет вид

$$\begin{array}{rl} & \Theta_0 \subseteq \Theta_1 \subseteq \ldots \subseteq \Theta_n \subseteq \Theta_{n+1} \subseteq \ldots \subseteq \Theta_h = \Theta, \\ \Omega \subseteq & \Omega_0 \subseteq \Omega_1 \subseteq \ldots \subseteq \Omega_n \subseteq \Omega_{n+1} \subseteq \ldots \subseteq \Omega_h, \\ & H_0 \subseteq H_1 \subseteq \ldots \subseteq H_n \subseteq H_{n+1} \subseteq \ldots \subseteq H_h. \end{array} \tag{6}$$

Прежде всего введем следующие обозначения. Обозначим через S_n множество всех скулемовских символов и предметных переменных, появившихся в задаче π_n в результате декомпозиции ее родительской задачи. Обозначим через $T_n \subseteq H_n$ множество всех термов $t \in H_n$ вида $f^m(\tau_1,\, \ldots,\, \tau_m)$, где f^m — некоторый скулемовский символ, которые находятся в дереве $(\Theta|\pi_n)\theta$ и содержат символы из S_n.

Покажем, как для каждого дерева Θ_n можно построить подстановки θ'_n, μ_n и дерево Θ'_n такие, что корнем дерева Θ'_n является задача $\pi'_0 = (\Gamma,\, A)$, дочерние вершины получаются из родительских по одному из допустимых правил декомпозиции системы $\mathcal{P}_3$, а листья дерева Θ'_n либо примитивизируются подстановкой θ'_n, либо соответствуют нераскрытым листьям $\pi_n = (\mathbf{Sk}(\Gamma_n)?A_n)$ дерева Θ_n и являются задачами π'_n вида $\pi'_n = (\Delta_n\Gamma'_n,\, A'_n)$. При этом роль подстановок θ'_n и μ_n в обозначаемом штрихом формируемом соответствии заключается в том, что на каждом шаге n θ'_n составляет часть искомой θ' и содержит свободные предметные переменные, вводимые для устранения скулемовских функций, а μ_n заменяет эти предметные переменные на термы со скулемовскими функциями. Для этого формируемое соответствие будет таким, чтобы соблюдались следующие свойства:

- если псевдовыражение E' входит в π'_n и соответствует псевдовыражению E, то $E\theta = E'(\theta'_n \circ \mu_n)$; в частности, $A_n\theta = A'_n(\theta'_n \circ \mu_n)$ и $\Gamma_n\theta = \Gamma'_n(\theta'_n \circ \mu_n)$;

- каждому терму $t \in H_n$ вида $f^m(\tau_1, \ldots, \tau_m)$ из дерева $(\Theta|\pi_n)\theta$ такому, что $t \notin T_n$, сопоставляется некоторый терм t' без скулемовских функций такой, что $t = t'\mu_n$.

Введем несколько новых обозначений. Для множества термов T_n определим подстановки δ_n и ν_n, которые будем называть *сопряженными* листу $\pi'_n = (\Delta_n\Gamma'_n, A'_n)$ дерева Θ'_n. Для этого рассмотрим последовательность псевдоформул $E_1, \ldots, E_l$ такую, что для каждого терма $t = f(\tau_1, \ldots, \tau_N)$ из T_n в этой последовательности содержится соответствующий этому терму фрагмент $P_1, \ldots, P_q$, *составленный согласно протоколу скулемизации*. Последнее означает следующее. Пусть скулемовский символ f появился в дереве поиска в результате скулемизации псевдоформулы $C \in \Gamma_n$. Пусть $\mathbf{PNF}(C, D)$ имеет вид (4). Пусть кванторная приставка (3) псевдоформулы C имеет вид (1). При этом будем считать, что символы $\varphi_1^{n_1}, \ldots, \varphi_k^{n_k}$ вводятся при скулемизации C с помощью элиминации кванторов $\exists v_1, \ldots, \exists v_k$ из (1) и f есть один из символов $\varphi_1^{n_1}, \ldots, \varphi_k^{n_k}$. Введем новые предметные переменные $y_1, \ldots, y_k$ и новые глобальные метапеременные $Y_1, \ldots, Y_{n_k}$. Составим цепочку псевдоформул $P_1, \ldots, P_q$ такую, что $P_1 = \mathbf{PNF}(C', D')$,

$$P_q = \forall z_{n_k+1} \ldots \forall z_{n_k+m}(C'_1\,\lambda_1\,(C'_2\,\lambda_2\,(\ldots(C'_s\,\lambda_s\,D'))\ldots)\gamma,$$

$$\gamma = \begin{pmatrix} z_1 & \ldots & z_{n_k} & v_1 & \ldots & v_k \\ Y_1 & \ldots & Y_{n_k} & y_1 & \ldots & y_k \end{pmatrix};$$

- если $P_j = \forall z_i\, B(z_i)$, то $P_{j+1} = B(Y_i)$;
- если $P_j = \exists v_i\, B(v_i)$, то $P_{j+1} = B(y_i)$.

Пусть термы $\tau'_1, \ldots, \tau'_{n_k}$ соответствуют термам $\tau_1, \ldots, \tau_{n_k} \in H_n$, т. е. $\tau_i = \tau'_i\mu_n$. *Сопряженными терму* $t = f(\tau_1, \ldots, \tau_N) \in T_n$ мы будем называть следующие подстановки:

$$\begin{pmatrix} Y_1 & Y_2 & \ldots & Y_{n_1} & \ldots & Y_{n_k} \\ \tau'_1 & \tau'_2 & \ldots & \tau'_{n_1} & \ldots & \tau'_{n_k} \end{pmatrix}, \tag{7}$$

$$\begin{pmatrix} y_1 & & \ldots & & y_k \\ \varphi_1^{n_1}(\tau_1, \ldots, \tau_{n_1}) & & \ldots & & \varphi_k^{n_k}(\tau_1, \ldots, \tau_{n_k}) \end{pmatrix}. \tag{8}$$

Объединив все вычисленные для каждого терма из T_n такие сопряженные им подстановки (7) мы и получим *сопряженную задаче* π'_n подстановку δ_n. Аналогично, объединив все подстановки (8) мы получаем подстановку ν_n.

Для так определенных подстановок δ_n и ν_n, сопряженных задаче $\pi'_n = (\Delta_n\Gamma'_n, A'_n)$, определим последовательность конечных множеств псевдоформул $\Delta^1_n \subseteq \Delta^2_n \subseteq \ldots \subseteq \Delta^l_n$ следующим образом: $\Delta^j_n = \Delta_n \cup \{E_1, \ldots, E_j\}$, $j = 1, \ldots, l$. Следствием индуктивного предположения о задаче π'_n будет то, что задачи

$$(\Delta_n\Gamma'_n, A'_n),\ (\Delta^1_n\Gamma'_n, A'_n),\ \ldots,\ (\Delta^l_n\Gamma'_n, A'_n) \tag{9}$$

образуют последовательность, в которой каждая следующая задача является результатом применения одного из правил декомпозиции (ПНФ$\to$), ($\forall\to$) или ($\exists\to$) к предыдущей. При этом δ_n и ν_n были построены так, чтобы цепочка задач (9) обладала следующими свойствами:

- если псевдовыражение E' входит в $(\Delta^l_n\Gamma'_n, A'_n)$ и соответствует псевдовыражению E, то $E\theta = E'((\theta'_n \cup \delta_n) \circ (\mu_n \cup \nu_n))$; в частности, $A_n\theta = A'_n((\theta'_n \cup \delta_n) \circ (\mu_n \cup \nu_n))$ и $\Gamma_n\theta = \Gamma'_n((\theta'_n \cup \delta_n) \circ (\mu_n \cup \nu_n))$;

- каждому терму $t \in H_n$ из дерева $(\Theta|\pi_n)\theta$ сопоставляется некоторый терм t' без скулемовских функций такой, что $t = t'(\mu_n \cup \nu_n)$.

В начале процесса построения дерева Θ' положим $\Delta_0 = \emptyset$, $\Gamma'_0 = \Gamma$, $A'_0 = A$, $\pi'_0 = (\Delta_0\Gamma'_0, A'_0)$, $\Theta'_0 = \{\pi'_0\}$, $\theta'_0 = \varepsilon$ и $\mu_0 = \varepsilon$. Очевидно, что дерево Θ'_0 по отношению к Θ_0 обладает требуемым свойством.

Покажем как получить дерево Θ'_{n+1} из дерева Θ'_n в зависимости от правила декомпозиции, которое используется при формировании дерева Θ_{n+1} из дерева Θ_n. Всего имеется 10 правил, которые мы рассмотрим в следующей последовательности: $(?\supset)$, $(?\top)$, $(?\vee_i)$, $(?\exists)$, $(?\&)$, $(?\forall)$, $(\bot_{\mathbf{c}})$, $(\vee?)$, $(\&?)$, $(\supset?)$.

1. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции $(?\supset)$. Это значит, что $A_n = (B_1 \supset B_2)$, $A_n\theta = (B_1 \supset B_2)\theta = (B_1\theta) \supset (B_2\theta) = (B'_1(\theta'_n \circ \mu_n)) \supset (B'_2(\theta'_n \circ \mu_n)) = (B'_1 \supset B'_2)(\theta'_n \circ \mu_n) = A'_n(\theta'_n \circ \mu_n)$, $A'_n = (B'_1 \supset B'_2)$, $\Gamma_{n+1} = \Gamma_n \cup \{B_1\}$, $A_{n+1} = B_2$ и язык Ω_{n+1} является расширением языка Ω_n за счет скулемизации B_1. Пусть подстановки δ_n и ν_n сопряжены листу π'_n. Пусть $(\Delta_n\Gamma'_n, B'_1 \supset B'_2)$, $(\Delta^1_n\Gamma'_n, B'_1 \supset B'_2)$, $\ldots$, $(\Delta^l_n\Gamma'_n, B'_1 \supset B'_2)$ есть построенная так, как указано выше (9), соответствующая подстановкам δ_n и ν_n цепочка задач, в которой каждая следующая задача является результатом применения одного из правил декомпозиции (ПНФ$\to$), ($\forall\to$) или ($\exists\to$) к предыдущей. Положим $\theta'_{n+1} = \theta'_n \cup \delta_n$, $\mu_{n+1} = \mu_n \cup \nu_n$, $\Gamma'_{n+1} = \Gamma'_n \cup \{B'_1\}$, $A'_{n+1} = B'_2$ и $\Delta_{n+1} = \Delta^l_n$. Дерево Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения сначала правил из списка (ПНФ$\to$), ($\forall\to$), ($\exists\to$), а

затем правила $(\to\supset)$:

$$\dfrac{\dfrac{(B'_1\ \Delta^l_n\Gamma'_n,\ B'_2)}{(\Delta^l_n\Gamma'_n,\ B'_1\supset B'_2)}}{\vdots}$$
$$\dfrac{(\Delta^1_n\Gamma'_n,\ B'_1\supset B'_2)}{(\Delta_n\Gamma'_n,\ B'_1\supset B'_2)}$$
$$\ddots$$
$$(\Gamma,\ A)$$

В результате для листа $(\mathbf{Sk}(\Gamma_{n+1})\,?\,A_{n+1})$ дерева Θ_{n+1} в дереве Θ'_{n+1} сформирован лист требуемого вида $(\Delta_{n+1}\Gamma'_{n+1},\ A'_{n+1})$ такой, что $A_{n+1}\theta = A'_{n+1}(\theta'_{n+1}\circ\mu_{n+1})$ и $\Gamma_{n+1}\theta = \Gamma'_{n+1}(\theta'_{n+1}\circ\mu_{n+1})$. Таким образом, дерево Θ'_{n+1} по отношению к Θ_{n+1} снова обладает требуемым свойством.

2. Пусть лист $\pi_n = (\mathbf{Sk}(\Gamma_n)\,?\,A_n)$ дерева Θ_n является примитивной задачей, у которой $A_n = \top$. Это значит, что $A'_n(\theta'_n\circ\mu_n) = A_n\theta = \top\theta = \top$ и $A'_n = \top$. В этом случае $\Theta_{n+1} = \Theta_n$. Положим $\Theta'_{n+1} = \Theta'_n$, $\theta'_{n+1} = \theta'_n$ и $\mu_{n+1} = \mu_n$. При этом дерево Θ'_{n+1} вновь обладает требуемым свойством и имеет вид:

$$(\Delta_n\Gamma'_n,\ \top)$$
$$\ddots$$
$$(\Gamma,\ A)$$

3. Пусть Θ_{n+1} получается из Θ_n в результате применения продукции $(?\vee_i)$. Следовательно имеем $A_n = (B_1\vee B_2)$, $A'_n = (B'_1\vee B'_2)$. Пусть подстановкам δ_n и ν_n соответствует цепочка задач $(\Delta^1_n\Gamma'_n,\ B'_1\vee B'_2)$, $\ldots$, $(\Delta^l_n\Gamma'_n,\ B'_1\vee B'_2)$. Положим $\theta'_{n+1} = \theta'_n\cup\delta_n$, $\mu_{n+1} = \mu_n\cup\nu_n$ и $A'_{n+1} = B'_i$. Дерево Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения правил $(\text{ПНФ}\to)$, $(\forall\to)$, $(\exists\to)$ и $(\to\vee_i)$:

$$\dfrac{\dfrac{(\Delta^l_n\Gamma'_n,\ B'_i)}{(\Delta^l_n\Gamma'_n,\ B'_1\vee B'_2)}}{\vdots}$$
$$\dfrac{(\Delta^1_n\Gamma'_n,\ B'_1\vee B'_2)}{(\Delta_n\Gamma'_n,\ B'_1\vee B'_2)}$$
$$\ddots$$
$$(\Gamma,\ A)$$

В результате сформированный в Θ'_{n+1} лист имеет вид $(\Delta_{n+1}\Gamma'_{n+1},\ A'_{n+1})$. Что соответствует новому листу $(\mathbf{Sk}(\Gamma_{n+1})?A_{n+1})$ в дереве Θ_{n+1}.

4. Пусть Θ_{n+1} получается из Θ_n с помощью применения правила декомпозиции $(?\exists)$. Это значит, что $A_n = (\exists x\,B(x))$, $A'_n = (\exists x\,B'(x))$,

$\Gamma_{n+1} = \Gamma_n$, $A_{n+1} = B(X)$, $X = X_s$, $X_s \in \mathbf{dom}\,\theta$ для некоторого $s \in \{1, \ldots, r\}$, $t_s \in H_n$, $t_s \in (\Theta|\pi_n)\theta$. Пусть терм t'_s соответствует терму t_s, т. е. $t'_s(\mu_n \cup \nu_n) = t_s$. Тогда положим $\theta'_{n+1} = \theta'_n \cup \delta_n \cup \begin{pmatrix} X_s \\ t'_s \end{pmatrix}$, $\mu_{n+1} = \mu_n \cup \nu_n$ и $A'_{n+1} = B'(X_s)$. Отсюда имеем $B'(X_s)\,\theta'_{n+1} = B'(t'_s)\,\theta'_n$ и $A_{n+1}\theta = A'_{n+1}(\theta'_{n+1} \circ \mu_{n+1})$. Таким образом, искомое дерево Θ'_{n+1} получается из дерева Θ'_n в результате применения сначала правил из списка (ПНФ$\to$), ($\forall \to$), ($\exists \to$), а затем правила ($\to \exists$):

$$\dfrac{\dfrac{\dfrac{(\Delta_n^l\Gamma'_n,\, B'(X_s))}{(\Delta_n^l\Gamma'_n,\, \exists x\, B'(x))}}{\vdots}}{\dfrac{(\Delta_n^1\Gamma'_n,\, \exists x\, B'(x))}{(\Delta_n\Gamma'_n,\, \exists x\, B'(x))}}$$
$$\ddots$$
$$(\Gamma,\, A)$$

5. Пусть Θ_{n+1} получается из Θ_n по правилу (?&). Это значит, что $A_n = (B_1 \,\&\, B_2)$, $A'_n = (B'_1 \,\&\, B'_2)$. Положим $\theta'_{n+1} = \theta'_n \cup \delta_n$ и $\mu_{n+1} = \mu_n \cup \nu_n$. Тогда искомое дерево Θ'_{n+1} получается из Θ'_n в результате применения правил (ПНФ$\to$), ($\forall \to$), ($\exists \to$) и ($\to \&$):

$$\dfrac{\dfrac{\dfrac{(\Delta_n^l\Gamma'_n,\, B'_1) \qquad (\Delta_n^l\Gamma'_n,\, B'_2)}{(\Delta_n^l\Gamma'_n,\, B'_1 \,\&\, B'_2)}}{\vdots}}{\dfrac{(\Delta_n^1\Gamma'_n,\, B'_1 \,\&\, B'_2)}{(\Delta_n\Gamma'_n,\, B'_1 \,\&\, B'_2)}}$$
$$\ddots$$
$$(\Gamma,\, A)$$

В результате каждый сформированный в Θ'_{n+1} новый лист имеет вид $(\Delta_{n+1}\Gamma'_{n+1},\, A'_{n+1})$, где $A'_{n+1} = B'_1$ или $A'_{n+1} = B'_2$. Что соответствует каждому новому листу $(\mathbf{Sk}(\Gamma_{n+1})?A_{n+1})$ дерева Θ_{n+1}.

6. Пусть Θ_{n+1} получается из Θ_n с помощью применения правила декомпозиции (?$\forall$). Это значит, что $A_n = (\forall x\, B(x))$, $A_{n+1} = B(a)$, $\Omega_{n+1} = \Omega_n \cup \{a\}$, $\Gamma_{n+1} = \Gamma_n$. Положим $\theta'_{n+1} = \theta'_n \cup \delta_n$, $\mu_{n+1} = \mu_n \cup \nu_n$ и $A'_{n+1} = B'(a)$. Дерево Θ'_{n+1} получим из дерева Θ'_n как результат нескольких применений сначала правил из списка (ПНФ$\to$), ($\forall \to$), ($\exists \to$), а

затем правила ($\to \forall$):

$$\dfrac{\dfrac{(\Delta_n^l\Gamma'_n,\, B'(a))}{(\Delta_n^l\Gamma'_n,\, \forall x\, B'(x))}}{\vdots}$$
$$\dfrac{(\Delta_n^1\Gamma'_n,\, \forall x\, B'(x))}{(\Delta_n\Gamma'_n,\, \forall x\, B'(x))}$$
$$\ddots$$
$$(\Gamma,\, A)$$

7. Пусть Θ_{n+1} получается из Θ_n по правилу $(\bot_{\mathbf{c}})$. Следовательно имеем $\Gamma_{n+1} = \Gamma_n \cup \{\neg A_n\}$, $A_{n+1} = \bot$, $\Gamma_n\theta = \Gamma'_n(\theta'_n \circ \mu_n)$ и $A_n\theta = A'_n(\theta'_n \circ \mu_n)$. Положим $\theta'_{n+1} = \theta'_n \cup \delta_n$, $\mu_{n+1} = \mu_n \cup \nu_n$, $\Gamma'_{n+1} = \Gamma'_n \cup \{\neg A'_n\}$ и $A'_{n+1} = \bot$. Дерево Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения сначала правил (ПНФ$\to$), $(\forall \to)$, $(\exists \to)$, а затем правила $(\bot_{\mathbf{c}})$:

$$\dfrac{\dfrac{((\neg A'_n)\, \Delta_n^l\Gamma'_n,\, \bot)}{(\Delta_n^l\Gamma'_n,\, A'_n)}}{\vdots}$$
$$\dfrac{(\Delta_n^1\Gamma'_n,\, A'_n)}{(\Delta_n\Gamma'_n,\, A'_n)}$$
$$\ddots$$
$$(\Gamma,\, A)$$

В результате для листа $(\mathbf{Sk}(\Gamma_{n+1})?A_{n+1})$ дерева Θ_{n+1} в дереве Θ'_{n+1} сформирован лист требуемого вида $(\Delta_{n+1}\Gamma'_{n+1},\, A'_{n+1})$ такой, что $\Gamma_{n+1}\theta = \Gamma'_{n+1}(\theta'_{n+1} \circ \mu_{n+1})$ и $A_{n+1}\theta = \bot = A'_{n+1}(\theta'_{n+1} \circ \mu_{n+1})$. Тем самым рассмотрение случая $(\bot_{\mathbf{c}})$ завершено.

Рассмотрение последних трех случаев $(\vee?)$, $(\&?)$ и $(\supset?)$ предварим определением новых понятий. Пусть F есть псевдоформула из левой части задачи, нормальная форма $\mathbf{NF}(F,\, G)$ которой используется в указанных правилах декомпозиции. Пусть $F = \mathbf{Sk}(C)$ и соответствующая (5) нормальная форма псевдоформулы C имеет вид (2). При этом будем считать, что при скулемизации C вводятся k скулемовских функций $\varphi_1^{n_1}, \ldots,$ $\varphi_k^{n_k}$ с помощью элиминации кванторов $\exists v_1, \ldots, \exists v_k$. Список всех строго положительных аналитических связанных кванторами всеобщности переменных в порядке вхождения в C запишем в виде $z_1, \ldots, z_m$. Пусть $Y_1, \ldots, Y_m$ суть глобальные метапеременные подстановки η, которые вводятся вместо локальных метапеременных $Z_1, \ldots, Z_m$ псевдоформулы F. Возьмем часть $\begin{pmatrix} Y_1 & Y_2 & \ldots & Y_m \\ \tau_1 & \tau_2 & \ldots & \tau_m \end{pmatrix}$ подстановки θ. Пусть термы $\tau'_1, \ldots,$

τ'_m соответствуют термам $\tau_1, \ldots, \tau_m \in H_n$, т. е. $\tau_i = \tau'_i(\mu_n \cup \nu_n)$. Положим

$$\tilde{\delta}_n = \delta_n \cup \begin{pmatrix} Y_1 & Y_2 & \ldots & Y_m \\ \tau'_1 & \tau'_2 & \ldots & \tau'_m \end{pmatrix}.$$

Введем новые свободные предметные переменные $y_1, \ldots, y_k$ и положим

$$\tilde{\nu}_n = \nu_n \cup \begin{pmatrix} y_1 & \ldots & y_k \\ \varphi_1^{n_1}(\tau_1, \ldots, \tau_{n_1}) & \ldots & \varphi_k^{n_k}(\tau_1, \ldots, \tau_{n_k}) \end{pmatrix}.$$

Составим список $D'_1, \ldots, D'_N$ всех строго положительных аналитических вхождений атомов, дизъюнкций, $\bot$ и $\top$ в $\mathbf{PNF}(C', D')$ такой, что $D'_1 = D'$. Сформируем список псевдоформул $P_1, \ldots, P_N$ такой, что $P_1 = \mathbf{PNF}(C', D'_1)$ и каждая следующая P_{j+1} в этом списке есть пренексная форма предыдущей P_j относительно D'_{j+1}. Достроим этот список, добавив новые псевдоформулы $P_{N+1}, \ldots, P_q$ такие, что P_q не содержит строго положительных аналитических кванторов и

- если $P_j = \forall z_i\, B(z_i)$, то $P_{j+1} = B(Y_i)$;
- если $P_j = \exists v_i\, B(v_i)$, то $P_{j+1} = B(y_i)$.

Обозначим $\tilde{C} = \mathbf{PNF}(C, D)$ и $\tilde{F} = \mathbf{NF}(F, G)\eta$. Псевдоформулы $P_1, \ldots, P_q$ в частности включают псевдоформулы $\tilde{C}' = P_1$ и $\tilde{F}' = P_q$, для которых выполняются равенства

$$\begin{array}{rcl} \tilde{C}\theta & = & \tilde{C}'((\theta'_n \cup \tilde{\delta}_n) \circ (\mu_n \cup \tilde{\nu}_n)), \\ \tilde{F}\theta & = & \tilde{F}'((\theta'_n \cup \tilde{\delta}_n) \circ (\mu_n \cup \tilde{\nu}_n)). \end{array} \tag{10}$$

Для сформированных псевдоформул $P_1, P_2, \ldots, P_q$ построим соответствующие им множества $\Delta_n^{l+1}, \Delta_n^{l+2}, \ldots, \Delta_n^{l'}$, где $l' = l + q$, согласно следующим равенствам:

$$\Delta_n^{l+1} = \Delta_n^l \cup \{P_1\},\ \Delta_n^{l+2} = \Delta_n^l \cup \{P_1, P_2\},\ \ldots,\ \Delta_n^{l'} = \Delta_n^l \cup \{P_1, \ldots, P_q\}.$$

Следствием индуктивного предположения о задаче π'_n будет то, что задачи

$$\begin{array}{c} (\Delta_n\Gamma'_n, A'_n),\ (\Delta_n^1\Gamma'_n, A'_n),\ \ldots,\ (\Delta_n^l\Gamma'_n, A'_n), \\ (\Delta_n^{l+1}\Gamma'_n, A'_n),\ \ldots,\ (\Delta_n^{l'}\Gamma'_n, A'_n) \end{array} \tag{11}$$

образуют последовательность, в которой каждая следующая задача является результатом применения одного из правил декомпозиции (ПНФ$\to$), (ПФ$\to$), ($\forall\ \to$) или ($\exists\ \to$) к предыдущей. При этом цепочка задач (11) обладает следующими свойствами:

- если псевдовыражение E' входит в $(\Delta_n^{l'}\Gamma'_n,\, A'_n)$ и соответствует псевдовыражению E, то $E\theta = E'((\theta'_n \cup \tilde{\delta}_n) \circ (\mu_n \cup \tilde{\nu}_n))$; в частности, $A_n\theta = A'_n((\theta'_n \cup \tilde{\delta}_n) \circ (\mu_n \cup \tilde{\nu}_n))$ и $\Gamma_n\theta = \Gamma'_n((\theta'_n \cup \tilde{\delta}_n) \circ (\mu_n \cup \tilde{\nu}_n))$;

- каждому терму $t \in H_n$ из дерева $(\Theta|\pi_n)\theta$ сопоставляется некоторый терм t' без скулемовских функций такой, что $t = t'(\mu_n \cup \tilde{\nu}_n)$.

Эти свойства указанных задач (11) нам потребуются ниже при рассмотрении тех случаев, когда дерево Θ_{n+1} формируется из дерева Θ_n с помощью правил декомпозиции $(\vee?)$, $(\&?)$ и $(\supset?)$.

8. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции $(\vee?)$. Тогда существует псевдоформула $F \in \mathbf{Sk}(\Gamma_n)$ с нормальной формой вида

$$F_1\, \lambda_1\, (F_2\, \lambda_2\, (\ldots (F_s\, \lambda_s\, (G_1 \vee G_2)) \ldots)$$

такая, что $p + 2$ задачи

$$((\Sigma_1\eta)\, \mathbf{Sk}(\Gamma_n)\, ?\, F_{i_1}\eta),\ \ldots,\ ((\Sigma_p\eta)\, \mathbf{Sk}(\Gamma_n)\, ?\, F_{i_p}\eta),$$

$$(\mathbf{Sk}(G_1\eta)\, (\Sigma\eta)\, \mathbf{Sk}(\Gamma_n)\, ?\, A_n), \quad (\mathbf{Sk}(G_2\eta)\, (\Sigma\eta)\, \mathbf{Sk}(\Gamma_n)\, ?\, A_n)$$

образуют у задачи π_n все дочерние вершины в дереве Θ. При этом будем считать, что $F_{i_1}\, \&\, F_{i_2}\, \&\, \ldots\, \&\, F_{i_p}$ — импликанта этой нормальной формы, подстановка η замещает все локальные метапеременные в $\mathbf{NF}(F,\, G_1 \vee G_2)$ новыми глобальными, а также

$$R_i = F_{i+1}\, \lambda_{i+1}\, (F_{i+2}\, \lambda_{i+2}\, (\ldots (F_s\, \lambda_s\, (G_1 \vee G_2)) \ldots), \quad i = 1,\, \ldots,\, s,$$

$$\Sigma_1 = \{\mathbf{NF}(F,\, G_1 \vee G_2),\, F_1,\, R_1,\, F_2,\, R_2,\, \ldots,\, F_{i_1-1},\, R_{i_1-1}\},$$

$$\Sigma_j = \Sigma_{j-1} \cup \{R_{i_{(j-1)}},\, \ldots,\, F_{i_j-1},\, R_{i_j-1}\}, \quad j = 2,\, \ldots,\, p,$$

$$\Sigma = \Sigma_p \cup \{R_{i_p},\, \ldots,\, F_s,\, R_s\}, \quad R_s = G_1 \vee G_2.$$

Введем обозначения. Пусть $F = \mathbf{Sk}(C)$ и соответствующая (5) нормальная форма псевдоформулы C имеет вид (2). Пусть кванторная приставка (3) псевдоформулы $\mathbf{PNF}(C,\, D_1 \vee D_2)$ имеет вид (1). Пусть $Y_1, \ldots, Y_m$ суть глобальные метапеременные подстановки η. Пусть

$$(\Delta_n\Gamma'_n,\, A'_n),\ (\Delta_n^1\Gamma'_n,\, A'_n),\ \ldots,\ (\Delta_n^l\Gamma'_n,\, A'_n),\ (\Delta_n^{l+1}\Gamma'_n,\, A'_n),\ \ldots,\ (\Delta_n^{l'}\Gamma'_n,\, A'_n)$$

есть построенная так, как указано выше (11), соответствующая подстановкам $\tilde{\delta}_n$ и $\tilde{\nu}_n$, цепочка задач, в которой каждая следующая задача является результатом применения одного из правил декомпозиции (ПНФ$\to$), (ПФ$\to$), ($\forall\to$) или ($\exists\to$) к предыдущей. Положим

$$\theta'_{n+1} = \theta'_n \cup \tilde{\delta}_n = \theta'_n \cup \delta_n \cup \begin{pmatrix} Y_1 & Y_2 & \dots & Y_m \\ \tau'_1 & \tau'_2 & \dots & \tau'_m \end{pmatrix},$$

$$\mu_{n+1} = \mu_n \cup \tilde{\nu}_n, \qquad \Delta_{n+1} = \Delta_n^{l'}.$$

Введем обозначения: $\tilde{C} = \mathbf{PNF}(C,\, D_1 \vee D_2)$, $\tilde{F} = \mathbf{NF}(F,\, G_1 \vee G_2)\eta$, $\tilde{F}_i = F_i\eta$, $\tilde{R}_i = R_i\eta$, $\tilde{\Sigma}_j = \Sigma_j\eta$ и $\tilde{\Sigma} = \Sigma\eta$. Пусть псевдоформулы $\tilde{F}'$, $\tilde{F}'_1$, $\tilde{R}'_1$, $\dots$, $\tilde{F}'_s$, $\tilde{R}'_s$ соответствуют псевдоформулам $\tilde{F}$, $\tilde{F}_1$, $\tilde{R}_1$, $\dots$, $\tilde{F}_s$, $\tilde{R}_s$. Введем следующие обозначения:

$$\tilde{\Sigma}'_1 = \{\tilde{F}',\, \tilde{F}'_1,\, \tilde{R}'_1,\, \tilde{F}'_2,\, \tilde{R}'_2,\, \dots,\, \tilde{F}'_{i_1-1},\, \tilde{R}'_{i_1-1}\},$$

$$\tilde{\Sigma}'_j = \tilde{\Sigma}'_{j-1} \cup \{\tilde{R}'_{i_{(j-1)}},\, \dots,\, \tilde{F}'_{i_j-1},\, \tilde{R}'_{i_j-1}\}, \quad j = 2,\, \dots,\, p,$$

$$\tilde{\Sigma}' = \tilde{\Sigma}'_p \cup \{\tilde{R}'_{i_p},\, \dots,\, \tilde{F}'_s,\, \tilde{R}'_s\}, \quad \tilde{R}'_s = \tilde{G}'_1 \vee \tilde{G}'_2.$$

Следствием (10) являются следующие равенства:

$$\tilde{C}\theta \;=\; \tilde{C}'(\theta'_{n+1} \circ \mu_{n+1}), \qquad \tilde{F}\theta \;=\; \tilde{F}'(\theta'_{n+1} \circ \mu_{n+1}),$$

$$\tilde{F}_i\theta = \tilde{F}'_i(\theta'_{n+1} \circ \mu_{n+1}), \quad \tilde{R}_i\theta = \tilde{R}'_i(\theta'_{n+1} \circ \mu_{n+1}), \quad i = 1,\, \dots,\, s,$$

$$\tilde{\Sigma}_j\theta = \tilde{\Sigma}'_j(\theta'_{n+1} \circ \mu_{n+1}), \quad \tilde{\Sigma}\theta = \tilde{\Sigma}'(\theta'_{n+1} \circ \mu_{n+1}), \quad j = 1,\, \dots,\, p.$$

Дерево Θ'_{n+1} сформируем из дерева Θ'_n как результат нескольких применений сначала правил из списка (ПНФ$\to$), (ПФ$\to$), ($\forall\to$), ($\exists\to$), затем из списка ($\supset\to$), ($\&\to$), и в конце концов ($\vee\to$):

$$\frac{(\tilde{G}'_1\, \tilde{\Sigma}'\, \Delta_n^{l'}\, \Gamma'_n,\, A'_n) \qquad (\tilde{G}'_2\, \tilde{\Sigma}'\, \Delta_n^{l'}\, \Gamma'_n,\, A'_n)}{((\tilde{G}'_1 \vee \tilde{G}'_2)\, \tilde{\Sigma}'\, \Delta_n^{l'}\, \Gamma'_n,\, A'_n)}$$

$$\ddots$$

$$\nwarrow$$

$$\frac{(\tilde{\Sigma}'_p\, \Delta_n^{l'}\, \Gamma'_n,\, \tilde{F}'_{i_p}) \qquad (\tilde{R}'_{i_p}\, \tilde{\Sigma}'_p\, \Delta_n^{l'}\, \Gamma'_n,\, A'_n)}{(\tilde{\Sigma}'_p\, \Delta_n^{l'}\, \Gamma'_n,\, A'_n)}$$

$$\ddots$$

$$\nwarrow$$

$$\frac{(\tilde{\Sigma}'_1\, \Delta_n^{l'}\, \Gamma'_n,\, \tilde{F}'_{i_1}) \qquad (\tilde{R}'_{i_1}\, \tilde{\Sigma}'_1\, \Delta_n^{l'}\, \Gamma'_n,\, A'_n)}{(\tilde{\Sigma}'_1\, \Delta_n^{l'}\, \Gamma'_n,\, A'_n)}$$

$$\vdots$$

$$\frac{\dfrac{(\tilde{R}'_2\, \tilde{F}'_2\, \tilde{R}'_1\, \tilde{F}'_1\, \Delta_n^{l'}\, \Gamma'_n,\, A'_n)}{(\tilde{R}'_1\, \tilde{F}'_1\, \Delta_n^{l'}\, \Gamma'_n,\, A'_n)}}{(\Delta_n^{l'}\, \Gamma'_n,\, A'_n)}$$

$$\vdots$$

$$\frac{(\Delta_n^1\, \Gamma'_n,\, A'_n)}{(\Delta_n\, \Gamma'_n,\, A'_n)}$$

$$\ddots \vdots \cdot^{\cdot^{\cdot}}$$

$$(\Gamma,\, A)$$

В результате в дереве Θ'_{n+1} сформированы $p+2$ новых листа. При этом для каждого нового листа $(\mathbf{Sk}(\Gamma_{n+1})\,?\,A_{n+1})$ дерева Θ_{n+1} в дереве Θ'_{n+1} сформирован новый лист требуемого вида $(\Delta_{n+1}\Gamma'_{n+1},\, A'_{n+1})$, где $\Delta_{n+1} = \Delta_n^{l'}$.

9. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции (&?). В этом случае подстановка θ примитивизирует задачу $\pi_n = (\mathbf{Sk}(\Gamma_n)\,?\,A_n)$. Это означает наличие псевдоформулы $F = \mathbf{Sk}(C) \in \mathbf{Sk}(\Gamma_n)$ с нормальной формой вида

$$F_1 \,\&\, (F_2 \,\&\, (\ldots (F_s \,\&\, G) \ldots)$$

такой, что подстановка θ является унификатором A_n и $G\eta$. При этом η — подстановка, которая заменяет все локальные метапеременные в $\mathbf{NF}(F,\, G)$ на новые глобальные. Введем следующие обозначения:

$$R_i = F_{i+1} \,\&\, (F_{i+2} \,\&\, (\ldots (F_s \,\&\, G) \ldots), \quad i = 1,\, \ldots,\, s; \qquad R_s = G.$$

Пусть

$$(\Delta_n\Gamma'_n,\, A'_n),\ (\Delta_n^1\Gamma'_n,\, A'_n),\ \ldots,\ (\Delta_n^l\Gamma'_n,\, A'_n),\ (\Delta_n^{l+1}\Gamma'_n,\, A'_n),\ \ldots,\ (\Delta_n^{l'}\Gamma'_n,\, A'_n)$$

есть построенная так, как указано выше (11), соответствующая подстановкам $\tilde{\delta}_n$ и $\tilde{\nu}_n$, цепочка задач, в которой каждая следующая задача является результатом применения одного из правил декомпозиции (ПНФ$\to$), (ПФ$\to$), ($\forall\to$) или ($\exists\to$) к предыдущей. Положим

$$\theta'_{n+1} = \theta'_n \cup \tilde{\delta}_n = \theta'_n \cup \delta_n \cup \begin{pmatrix} Y_1 & Y_2 & \dots & Y_m \\ \tau'_1 & \tau'_2 & \dots & \tau'_m \end{pmatrix},$$

$$\mu_{n+1} = \mu_n \cup \tilde{\nu}_n.$$

Введем обозначения: $\tilde{C} = \mathbf{PNF}(C, D)$, $\tilde{F} = \mathbf{NF}(F, G)\eta$, $\tilde{F}_i = F_i\eta$, $\tilde{R}_i = R_i\eta$, $\tilde{\Sigma}_j = \Sigma_j\eta$, $\tilde{\Sigma} = \Sigma\eta$ и $\tilde{G} = G\eta$. Пусть псевдоформулы $\tilde{F}'$, $\tilde{F}'_1$, $\tilde{R}'_1$, …, $\tilde{F}'_s$, $\tilde{R}'_s$ соответствуют псевдоформулам $\tilde{F}$, $\tilde{F}_1$, $\tilde{R}_1$, …, $\tilde{F}_s$, $\tilde{R}_s$. Следствием (10) являются следующие равенства:

$$\tilde{C}\theta = \tilde{C}'(\theta'_{n+1} \circ \mu_{n+1}), \qquad \tilde{F}\theta = \tilde{F}'(\theta'_{n+1} \circ \mu_{n+1}),$$

$$\tilde{F}_i\theta = \tilde{F}'_i(\theta'_{n+1} \circ \mu_{n+1}), \quad \tilde{R}_i\theta = \tilde{R}'_i(\theta'_{n+1} \circ \mu_{n+1}), \quad i = 1, \dots, s.$$

Таким образом имеем

$$A'_n(\theta'_{n+1}\circ\mu_{n+1}) = A'_n(\theta'_n\circ\mu_n) = A_n\theta = (G\eta)\theta = \tilde{G}\theta = \tilde{R}_s\theta = \tilde{R}'_s(\theta'_{n+1}\circ\mu_{n+1}),$$

$$A'_n\theta'_{n+1} = \tilde{R}'_s\theta'_{n+1} = \tilde{G}'\theta'_{n+1}.$$

Дерево Θ'_{n+1} сформируем из дерева Θ'_n как результат нескольких применений сначала правил из списка (ПНФ$\to$), (ПФ$\to$), ($\forall\to$), ($\exists\to$), а затем ($\&\to$):

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{(\tilde{R}'_s\,\tilde{F}'_s\,\dots\,\tilde{R}'_2\,\tilde{F}'_2\,\tilde{R}'_1\,\tilde{F}'_1\,\Delta_n^{l'}\,\Gamma'_n,\ A'_n)}{\vdots}}{(\tilde{R}'_2\,\tilde{F}'_2\,\tilde{R}'_1\,\tilde{F}'_1\,\Delta_n^{l'}\,\Gamma'_n,\ A'_n)}}{(\tilde{R}'_1\,\tilde{F}'_1\,\Delta_n^{l'}\,\Gamma'_n,\ A'_n)}}{(\Delta_n^{l'}\,\Gamma'_n,\ A'_n)}}{\vdots}}{(\Delta_n^1\,\Gamma'_n,\ A'_n)}}{(\Delta_n\,\Gamma'_n,\ A'_n)}}{\ddots\ \vdots\ \iddots}$$
$$(\Gamma,\ A)$$

В результате в дереве Θ'_{n+1} сформирован лист, который примитивизируется подстановкой θ'_{n+1}.

10. Пусть Θ_{n+1} получается из Θ_n по правилу ($\supset$?). Тогда существует псевдоформула $F \in \mathbf{Sk}(\Gamma_n)$ с нормальной формой вида

$$F_1\,\lambda_1\,(F_2\,\lambda_2\,(\ldots(F_s\,\lambda_s\,G)\ldots)$$

такой, что p задач

$$((\Sigma_1\xi)\,\mathbf{Sk}(\Gamma_n)\,?\,F_{i_1}\xi),\ \ldots,\ ((\Sigma_p\xi)\,\mathbf{Sk}(\Gamma_n)\,?\,F_{i_p}\xi)$$

образуют у задачи π_n все дочерние вершины в дереве Θ. При этом псевдоформула $F_{i_1}\,\&\,F_{i_2}\,\&\,\ldots\,\&\,F_{i_p}$ есть импликанта этой нормальной формы, подстановка η замещает все локальные метапеременные в F новыми глобальными, $\xi = \eta \circ \sigma$, $\sigma = \mathbf{MGU}(A_n,\, G\eta)$, подстановка θ совместима с σ и также является унификатором A_n и $G\eta$. Кроме того, имеем

$$R_i = F_{i+1}\,\lambda_{i+1}\,(F_{i+2}\,\lambda_{i+2}\,(\ldots(F_s\,\lambda_s\,G)\ldots),\quad i = 1,\,\ldots,\,s,$$

$$\Sigma_1 = \{\mathbf{NF}(F,\,G),\,F_1,\,R_1,\,F_2,\,R_2,\,\ldots,\,F_{i_1-1},\,R_{i_1-1}\},$$
$$\Sigma_j = \Sigma_{j-1} \cup \{R_{i_{(j-1)}},\,\ldots,\,F_{i_j-1},\,R_{i_j-1}\},\quad j = 2,\,\ldots,\,p,$$
$$\Sigma = \Sigma_p \cup \{R_{i_p},\,\ldots,\,F_s,\,R_s\},\quad R_s = G.$$

Введем обозначения. Пусть $F = \mathbf{Sk}(C)$ и соответствующая (5) нормальная форма псевдоформулы C имеет вид (2). Пусть кванторная приставка (3) псевдоформулы $\mathbf{PNF}(C,\,D)$ имеет вид (1). Пусть $Y_1,\,\ldots,\,Y_m$ суть глобальные метапеременные подстановки η. Пусть

$$(\Delta_n\Gamma'_n,\,A'_n),\ (\Delta_n^1\Gamma'_n,\,A'_n),\ \ldots,\ (\Delta_n^l\Gamma'_n,\,A'_n),\ (\Delta_n^{l+1}\Gamma'_n,\,A'_n),\ \ldots,\ (\Delta_n^{l'}\Gamma'_n,\,A'_n)$$

есть построенная так, как указано выше (11), соответствующая подстановкам $\tilde{\delta}_n$ и $\tilde{\nu}_n$, цепочка задач, в которой каждая следующая задача является результатом применения одного из правил декомпозиции (ПНФ$\to$), (ПФ$\to$), ($\forall\to$) или ($\exists\to$) к предыдущей. Положим

$$\theta'_{n+1} = \theta'_n \cup \tilde{\delta}_n = \theta'_n \cup \delta_n \cup \begin{pmatrix} Y_1 & Y_2 & \ldots & Y_m \\ \tau'_1 & \tau'_2 & \ldots & \tau'_m \end{pmatrix},$$

$$\mu_{n+1} = \mu_n \cup \tilde{\nu}_n,\quad \Delta_{n+1} = \Delta_n^{l'}.$$

Как было установлено ранее (10), для псевдоформул $\tilde{C} = \mathbf{PNF}(C,\,D)$ и $\tilde{F} = \mathbf{NF}(F,\,G)\eta$ выполняются равенства

$$\tilde{C}\theta \;=\; \tilde{C}'(\theta'_{n+1} \circ \mu_{n+1}),\quad \tilde{F}\theta \;=\; \tilde{F}'(\theta'_{n+1} \circ \mu_{n+1}).$$

Кроме того, имеем следующие равенства: $\theta = \sigma \circ \theta$,

$$(\mathbf{NF}(F,\, G)\xi)\theta = (\mathbf{NF}(F,\, G)(\eta \circ \sigma))\theta = (\mathbf{NF}(F,\, G)\eta)(\sigma \circ \theta) \\ = (\mathbf{NF}(F,\, G)\eta)\theta = \tilde{F}\theta.$$

Введем обозначения: $\tilde{F}_i = F_i\xi$, $\tilde{R}_i = R_i\xi$, $\tilde{\Sigma}_j = \Sigma_j\xi$, $\tilde{\Sigma} = \Sigma\xi$, $\tilde{G} = G\xi$, $\tilde{F}_0 = \mathbf{NF}(F,\, G)\xi$. Пусть псевдоформулы $\tilde{F}'_0,\ \tilde{F}'_1,\ \tilde{R}'_1,\ \ldots,\ \tilde{F}'_s,\ \tilde{R}'_s$ соответствуют псевдоформулам $\tilde{F}_0,\ \tilde{F}_1,\ \tilde{R}_1,\ \ldots,\ \tilde{F}_s,\ \tilde{R}_s$. Введем следующие обозначения

$$\tilde{\Sigma}'_1 = \{\tilde{F}'_0,\ \tilde{F}'_1,\ \tilde{R}'_1,\ \tilde{F}'_2,\ \tilde{R}'_2,\ \ldots,\ \tilde{F}'_{i_1-1},\ \tilde{R}'_{i_1-1}\},$$

$$\tilde{\Sigma}'_j = \tilde{\Sigma}'_{j-1} \cup \{\tilde{R}'_{i_{(j-1)}},\ \ldots,\ \tilde{F}'_{i_j-1},\ \tilde{R}'_{i_j-1}\}, \quad j = 2,\, \ldots,\, p,$$

$$\tilde{\Sigma}' = \tilde{\Sigma}'_p \cup \{\tilde{R}'_{i_p},\ \ldots,\ \tilde{F}'_s,\ \tilde{R}'_s\}, \quad \tilde{R}'_s = \tilde{G}'.$$

Для перечисленных псевдовыражений выполняются следующие равенства

$$\tilde{F}_i\theta = \tilde{F}'_i(\theta'_{n+1} \circ \mu_{n+1}), \quad \tilde{R}_i\theta = \tilde{R}'_i(\theta'_{n+1} \circ \mu_{n+1}), \quad i = 1,\, \ldots,\, s,$$

$$\tilde{\Sigma}_j\theta = \tilde{\Sigma}'_j(\theta'_{n+1} \circ \mu_{n+1}), \quad \tilde{\Sigma}\theta = \tilde{\Sigma}'(\theta'_{n+1} \circ \mu_{n+1}), \quad j = 1,\, \ldots,\, p.$$

Таким образом имеем

$$A'_n(\theta'_{n+1} \circ \mu_{n+1}) = A'_n(\theta'_n \circ \mu_n) = A_n\theta = (G\eta)\theta = (G\xi)\theta \\ = \tilde{G}\theta = \tilde{G}'(\theta'_{n+1} \circ \mu_{n+1}),$$

$$A'_n\theta'_{n+1} = \tilde{R}'_s\theta'_{n+1} = \tilde{G}'\theta'_{n+1}.$$

Дерево Θ'_{n+1} сформируем из дерева Θ'_n как результат нескольких применений сначала правил из списка (ПНФ$\to$), (ПФ$\to$), ($\forall \to$), ($\exists \to$), а затем из списка ($\supset\to$), (& $\to$):

$$
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
(\tilde{G}'\,\tilde{\Sigma}'\,\Delta_n^{l'}\,\Gamma'_n,\,A'_n)
}{\vdots}
}{\nwarrow}
}{(\tilde{\Sigma}'_p\,\Delta_n^{l'}\,\Gamma'_n,\,\tilde{F}'_{i_p})\qquad(\tilde{R}'_{i_p}\,\tilde{\Sigma}'_p\,\Delta_n^{l'}\,\Gamma'_n,\,A'_n)}
}{(\tilde{\Sigma}'_p\,\Delta_n^{l'}\,\Gamma'_n,\,A'_n)}
}{\vdots}
}{\nwarrow}
}{(\tilde{\Sigma}'_1\,\Delta_n^{l'}\,\Gamma'_n,\,\tilde{F}'_{i_1})\qquad(\tilde{R}'_{i_1}\,\tilde{\Sigma}'_1\,\Delta_n^{l'}\,\Gamma'_n,\,A'_n)}
}{(\tilde{\Sigma}'_1\,\Delta_n^{l'}\,\Gamma'_n,\,A'_n)}
}{\vdots}
}{(\tilde{R}'_2\,\tilde{F}'_2\,\tilde{R}'_1\,\tilde{F}'_1\,\Delta_n^{l'}\,\Gamma'_n,\,A'_n)}
}{(\tilde{R}'_1\,\tilde{F}'_1\,\Delta_n^{l'}\,\Gamma'_n,\,A'_n)}
}{(\Delta_n^{l'}\,\Gamma'_n,\,A'_n)}
}{\vdots}
}{(\Delta_n^1\,\Gamma'_n,\,A'_n)}
}{(\Delta_n\,\Gamma'_n,\,A'_n)}
\qquad
\begin{array}{c}\ddots\\ \vdots\\ (\Gamma,\,A)\end{array}
$$

В результате в дереве Θ'_{n+1} сформирован $p+1$ лист, один из которых примитивизируется подстановкой θ'_{n+1}, а остальные p листьев соответствуют новым p листьям дерева Θ_{n+1} и имеют требуемый вид. Тем самым рассмотрение случая $(\supset?)$ завершено.

Для построения Θ_{n+1}, а затем θ'_{n+1} и Θ'_{n+1}, в соответствии с применяемой стратегией мы выбирали лист $\pi_n = (\mathbf{Sk}(\Gamma_n)?A_n) \in \Theta_n$, у которого порождали дочерние разрешимые вершины, соответствующие подстановке θ и обеспечивающие разрешимость вершины π_n. В результате была сформирована подстановка θ'_{n+1} вида $\theta'_{n+1} = \theta'_n \cup \delta_n \cup \beta$, а также было сформировано дерево Θ'_{n+1}, которое получается из дерева Θ'_n в результате замены листа $\pi'_n = (\Delta_n\Gamma'_n,\, A'_n)$, соответствующего листу π_n дерева Θ_n, на некоторое новое поддерево с корнем π'_n. При этом каждая вершина ρ полученного дерева Θ'_{n+1} обладает следующим свойством: во-первых, если ρ является листом дерева Θ'_{n+1}, который не соответствует нераскрытому листу дерева Θ_{n+1}, то $\rho\theta'_{n+1}$ является примитивной задачей; во-вторых, если θ'_{n+1} является допустимой для вершин $\rho_1, \ldots, \rho_k$, всех дочерних у вершины ρ в дереве Θ'_{n+1}, то θ'_{n+1} является также допустимой и для ρ. На заключительном шаге этого процесса мы получаем

дерево $\Theta' = \Theta'_h$ и подстановку $\theta' = \theta'_h$ такие, что θ' является допустимой подстановкой задачи $(\Gamma,\, A)$ и Θ' является соответствующим этой подстановке θ' деривационным деревом. Предложение 4.2 доказано. □

Теорема о корректности. *Если задача* $(\mathbf{Sk}(\Gamma)\,?\,A)$ *разрешима в* $\mathcal{F}_3$, *то секвенция* $\Gamma \to A$ *выводима в* $\mathcal{S}_3$.

Доказательство. Пусть задача $(\mathbf{Sk}(\Gamma)\,?\,A)$ разрешима в $\mathcal{F}_3$. По предложению 4.2 отсюда задача $(\Gamma,\, A)$ разрешима в $\mathcal{P}_3$. По теореме о корректности $\mathcal{P}_3$ тогда секвенция $\Gamma \to A$ выводима в $\mathcal{S}_3$. Что и требовалось доказать. □

Доказательство корректности содержит в себе эффективный алгоритм получения вывода в односукцедентном исчислении секвенций из деривационного поддерева поискового дерева. В то же время существует эффективный алгоритм получения натурального вывода из односукцедентного секвенциального вывода. Все это вместе дает обоснование корректности сформулированной в виде продукционной системы процедуры поиска натурального вывода. Полнота этого метода поиска вывода рассматривается в следующем разделе.

5. Теорема о полноте

Формулируемая в этом разделе теорема о полноте обеспечивает для всякой выводимой в исчислении $\mathcal{S}_3$ секвенции вычисление ее логического вывода в рамках продукционной системы $\mathcal{F}_3$. Будем обозначать через $\overline{\mathcal{P}}_3$ продукционную систему, которая получается из $\mathcal{P}_3$ в результате обобщения правил декомпозиции системы $\mathcal{P}_3$ на произвольную псевдоформулу C в сукцеденте. Нам удобно установить полноту $\mathcal{F}_3$ в $\overline{\mathcal{P}}_3$. Это влечет полноту $\mathcal{F}_3$ в $\mathcal{P}_3$. Прежде всего мы должны установить справедливость нескольких формулируемых ниже предложений, в которых устанавливается допустимость в $\mathcal{F}_3$ правил декомпозиции $\overline{\mathcal{P}}_3$.

Предложение 5.1. *Если в* $\mathcal{F}_3$ *разрешима задача* $\mathbf{Sk}(A)\,\mathbf{Sk}(B)\,\Gamma?G$ *и* θ *— ее допустимая подстановка, то задача* $\mathbf{Sk}(A\,\&\,B)\,\Gamma?G$ *также является разрешимой и* θ *является ее допустимой подстановкой.*

Доказательство. Утверждение докажем индукцией по высоте деривационного дерева разрешимой задачи $\mathbf{Sk}(A)\,\mathbf{Sk}(B)\,\Gamma?G$. Во всех случаях

строения этого деривационного дерева рассуждения имеют подобный характер. В связи с этим рассмотрим лишь три случая. Остальные подобны рассмотренным.

1. Рассмотрим случай, когда задача $\mathbf{Sk}(A)\,\mathbf{Sk}(B)\,\Gamma?G$ является примитивизируемой и θ — ее примитивизирующая подстановка. Будем считать, что $\theta = \mathbf{MGU}(G,\, G'\eta)$, где $G' = C_m$, $\mathbf{NF}(C,\, G') = \&_{i=1}^{i=m} C_i$ и подстановка η замещает все локальные метапеременные в C на новые глобальные. Тогда $C \in \Gamma$, $C = \mathbf{Sk}(A)$ или $C = \mathbf{Sk}(B)$. Если $C \in \Gamma$, то задача $\mathbf{Sk}(A \,\&\, B)\,\Gamma?G$ также примитивизируется подстановкой θ. Если же $C = \mathbf{Sk}(A)$, то $\mathbf{Sk}(A \,\&\, B) = \mathbf{Sk}(A) \,\&\, \mathbf{Sk}(B)$, $\mathbf{NF}(\mathbf{Sk}(A \,\&\, B),\, G') = \&_{i=0}^{i=m} C_i = C_0 \,\&\, \mathbf{NF}(C,\, G')$, где $C_0 = \mathbf{Sk}(B)$. Таким образом подстановка θ будет также примитивизировать задачу $\mathbf{Sk}(A \,\&\, B)\Gamma?G$. Аналогично рассматривается случай $C = \mathbf{Sk}(B)$.

2. Рассмотрим случай, когда задача $\mathbf{Sk}(A)\,\mathbf{Sk}(B)\,\Gamma?G$ получается в деривационном дереве по правилу $(\supset?)$ из подзадач

$$\mathbf{Sk}(A)\,\mathbf{Sk}(B)\,(\Delta_1\xi)\,\Gamma\,?\,C_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(A)\,\mathbf{Sk}(B)\,(\Delta_k\xi)\,\Gamma\,?\,C_{i_k}\xi,$$

где $C_1\lambda_1(C_2\lambda_2(\ldots(C_n\lambda_n G'))\ldots)$ — нормальная форма псевдоформулы C, $C_{i_1} \,\&\, C_{i_2} \,\&\, \ldots,\, C_{i_k}$ — ее импликанта, $\xi = \eta \circ \sigma$ и θ — комбинация σ, θ_1, $\ldots$, θ_k. Здесь следует рассмотреть несколько подслучаев.

(а) Случай $C \in \Gamma$. По индуктивному предположению тогда разрешимы задачи

$$\mathbf{Sk}(A \,\&\, B)\,(\Delta_1\xi)\,\Gamma\,?\,C_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(A \,\&\, B)\,(\Delta_k\xi)\,\Gamma\,?\,C_{i_k}\xi \qquad (12)$$

и θ_1, $\ldots$, θ_k являются их допустимыми подстановками. По правилу $(\supset ?)$ тогда разрешима задача $\mathbf{Sk}(A \,\&\, B)\,\Gamma?G$ и θ является ее допустимой подстановкой.

(б) Теперь пусть $C = \mathbf{Sk}(B)$. По индуктивному предположению тогда разрешимы задачи (12) и θ_1, $\ldots$, θ_k суть их допустимые подстановки. Тогда разрешимы задачи

$$\mathbf{Sk}(A \,\&\, B)\,(\tilde{\Delta}_1\xi)\,\Gamma\,?\,C_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(A \,\&\, B)\,(\tilde{\Delta}_k\xi)\,\Gamma\,?\,C_{i_k}\xi,$$

где $\tilde{\Delta}_i = \Delta_i \cup \{\tilde{C},\, C_0\}$, $\tilde{C} = \mathbf{NF}(\mathbf{Sk}(A\&B),\, G')$, $C_0 = \mathbf{Sk}(A)$. По правилу $(\supset?)$ тогда также разрешима задача $\mathbf{Sk}(A \,\&\, B)\,\Gamma?G$, у которой θ является допустимой подстановкой.

(в) Случай $C = \mathbf{Sk}(A)$ рассматривается аналогично случаю (б).

3. Рассмотрим случай, когда $G = C_1 \vee C_2$ и задача $\mathbf{Sk}(A)\,\mathbf{Sk}(B)\,\Gamma?G$ получается в деривационном дереве по правилу $(?\vee_i)$ из $\mathbf{Sk}(A)\,\mathbf{Sk}(B)\,\Gamma?C_i$.

По индуктивному предположению тогда должна быть разрешима задача $\mathbf{Sk}(A \,\&\, B)\,\Gamma?C_i$, у которой θ будет допустимой подстановкой. По правилу $(?\vee_i)$ тогда разрешима задача $\mathbf{Sk}(A \,\&\, B)\,\Gamma?G$ и θ является ее допустимой подстановкой. □

Предложение 5.2. *Если задача* $\mathbf{Sk}(A)\,\mathbf{Sk}(B)\,\mathbf{Sk}(A \,\&\, B)\,\Gamma?G$ *разрешима в* $\mathcal{F}_3$ *и* θ *— ее допустимая подстановка, то задача* $\mathbf{Sk}(A \,\&\, B)\,\Gamma?G$ *также является разрешимой и* θ *является ее допустимой подстановкой.*

Доказательство. Пусть разрешима задача $\mathbf{Sk}(A)\,\mathbf{Sk}(B)\,\mathbf{Sk}(A \,\&\, B)\,\Gamma?G$ и θ — ее допустимая подстановка. По предложению 5.1 тогда должна быть также разрешима задача $\mathbf{Sk}(A \,\&\, B)\,\mathbf{Sk}(A \,\&\, B)\,\Gamma?G$ и θ будет ее допустимой подстановкой. Что и требовалось доказать. □

Предложение 5.3. *Если* $\mathbf{Sk}(A \supset B)\Gamma?A$ *и* $\mathbf{Sk}(B)\,\mathbf{Sk}(A \supset B)\,\Gamma?G$ *суть разрешимые задачи в* $\mathcal{F}_3$*, у которых существуют совместимые допустимые подстановки* θ_1 *и* θ_2*, то задача* $\mathbf{Sk}(A \supset B)\Gamma?G$ *также является разрешимой и комбинация* θ_1 *и* θ_2 *является ее допустимой подстановкой.*

Доказательство. Утверждение докажем индукцией по высоте деривационного дерева разрешимой задачи $\mathbf{Sk}(B)\,\mathbf{Sk}(A \supset B)\,\Gamma?G$. Разбор случаев имеет шаблонный характер. Рассмотрим только два случая строения деривационного дерева.

1. Рассмотрим случай, когда задача $\mathbf{Sk}(B)\,\mathbf{Sk}(A \supset B)\,\Gamma?G$ является примитивизируемой и θ_2 — ее примитивизирующая подстановка. Будем считать, что $\theta_2 = \mathbf{MGU}(G,\, G'\eta)$, где $G' = C_m$, $\mathbf{NF}(C,\, G') = \&_{i=1}^{i=m} C_i$ и подстановка η замещает все локальные метапеременные в C на новые глобальные. Тогда $C \in \Gamma$ или $C = \mathbf{Sk}(B)$. Если $C \in \Gamma$, то задача $\mathbf{Sk}(A \supset B)\,\Gamma?G$ также примитивизируется подстановкой θ_2 и, следовательно, примитивизируется комбинацией подстановок θ_1 и θ_2. Если же $C = \mathbf{Sk}(B)$, то $\mathbf{Sk}(A \supset B) = A \supset \mathbf{Sk}(B) = A \supset C$, $\mathbf{NF}(A \supset C,\, G') = A \supset \&_{i=1}^{i=m} C_i$, $A\eta = A$, $(A \supset \&_{i=1}^{i=m} C_i)\eta = A \supset \&_{i=1}^{i=m}(C_i\eta)$ и комбинация подстановок θ_1 и θ_2 будет не только допустимой подстановкой для задачи $\mathbf{Sk}(A \supset B)\Gamma?A\theta_2$, но по правилу $(\supset?)$ будет также допустимой подстановкой для задачи $\mathbf{Sk}(A \supset B)\Gamma?G$.

2. Рассмотрим случай, когда задача $\mathbf{Sk}(B)\,\mathbf{Sk}(A \supset B)\,\Gamma?G$ получается в деривационном дереве по правилу $(\supset?)$ из подзадач

$$\mathbf{Sk}(B)\,\mathbf{Sk}(A \supset B)\,(\Delta_1\xi)\,\Gamma\,?\,C_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(B)\,\mathbf{Sk}(A \supset B)\,(\Delta_k\xi)\,\Gamma\,?\,C_{i_k}\xi,$$

где $C_{i_1} \,\&\, C_{i_2} \,\&\, \ldots, C_{i_k}$ — импликанта $\mathbf{NF}(C, G')$, $\xi = \eta \circ \sigma$ и θ_2 — комбинация подстановок σ, $\beta_1, \ldots, \beta_k$. Здесь возникают два варианта.

(а) Случай $C \in \Gamma$ или $C = \mathbf{Sk}(A \supset B)$. По индуктивному предположению тогда разрешимы задачи

$$\mathbf{Sk}(A \supset B)\,(\Delta_1\xi)\,\Gamma\,?\,C_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(A \supset B)\,(\Delta_k\xi)\,\Gamma\,?\,C_{i_k}\xi. \qquad (13)$$

По правилу $(\supset?)$ тогда разрешима задача $\mathbf{Sk}(A \supset B)\,\Gamma?G$ и комбинация подстановок θ_1, σ, $\beta_1, \ldots, \beta_k$, которая совпадает с комбинацией θ_1 и θ_2, является ее допустимой подстановкой.

(б) Случай $C = \mathbf{Sk}(B)$. По индуктивному предположению тогда разрешимы задачи (13). Отметим, что $\mathbf{Sk}(A \supset B) = A \supset \mathbf{Sk}(B) = A \supset C$, $A\eta = A$, $A\xi = A\sigma$, а также комбинация θ_1 и θ_2 является допустимой подстановкой для $\mathbf{Sk}(A \supset B)\,\Gamma?A\xi$. Тогда разрешимы задачи

$$(A \supset \mathbf{Sk}(B))\,\Gamma?A\xi,\ (A \supset \mathbf{Sk}(B))\,(\tilde{\Delta}_1\xi)\Gamma?C_{i_1}\xi,\ \ldots,\ (A \supset \mathbf{Sk}(B))\,(\tilde{\Delta}_k\xi)\Gamma?C_{i_k}\xi,$$

где $\tilde{\Delta}_i = \Delta_i \cup \{\tilde{C}, C_0\}$, $\tilde{C} = \mathbf{NF}(A \supset \mathbf{Sk}(B), G')$, $C_0 = A$. По правилу $(\supset?)$ тогда разрешима задача $\mathbf{Sk}(A \supset B)\,\Gamma?G$, у которой комбинация θ_1 и θ_2 является ее допустимой подстановкой. □

Предложение 5.4. *Если задача* $\mathbf{Sk}(A(X))\,\mathbf{Sk}(\forall x\,A(x))\,\Gamma?G$ *разрешима в* $\mathcal{F}_3$ *и* θ *— ее допустимая подстановка, где* X *— новая глобальная метапеременная, то разрешима также задача* $\mathbf{Sk}(\forall x\,A(x))\,\Gamma?G$ *и* θ *является ее допустимой подстановкой.*

Доказательство. Утверждение докажем индукцией по высоте деривационного дерева разрешимой задачи $\mathbf{Sk}(A(X))\,\mathbf{Sk}(\forall x\,A(x))\,\Gamma?G$. Мы анализируем только два случая строения этого деривационного дерева. Остальные подобны рассмотренным.

1. Рассмотрим случай, когда задача $\mathbf{Sk}(A(X))\,\mathbf{Sk}(\forall x\,A(x))\,\Gamma?G$ является примитивизируемой и θ — ее примитивизирующая подстановка. Будем считать, что $\theta = \mathbf{MGU}(G, G'\eta)$, где $G' = C_m$, $\mathbf{NF}(C, G') = \&_{i=1}^{i=m} C_i$ и подстановка η замещает все локальные метапеременные в C на новые глобальные. Тогда $C \in \Gamma$, $C = \mathbf{Sk}(A(X))$ или $C = \mathbf{Sk}(\forall x\,A(x))$. Если $C \in \Gamma$ или $C = \mathbf{Sk}(\forall x\,A(x))$, то задача $\mathbf{Sk}(\forall x\,A(x))\,\Gamma?G$ также примитивизируется подстановкой θ. Теперь пусть $C = \mathbf{Sk}(A(X))$ и при скулемизации $\forall x\,A(x)$ вместо предметной переменной x вводится локальная метапеременная Z. Положим $\eta_1 = \eta \circ \begin{pmatrix} Z \\ X \end{pmatrix}$. Подстановка η_1 замещает все локальные метапеременные в псевдоформуле $C\begin{pmatrix} X \\ Z \end{pmatrix}$, кото-

рая совпадает с $\mathbf{Sk}(\forall x\, A(x))$, на новые глобальные и $C\eta = C\begin{pmatrix} X \\ Z \end{pmatrix}\eta_1$. Таким образом подстановка θ будет также примитивизировать задачу $\mathbf{Sk}(\forall x\, A(x))\Gamma?G$.

2. Рассмотрим случай, когда задача $\mathbf{Sk}(A(X))\,\mathbf{Sk}(\forall x\, A(x))\,\Gamma?G$ получается в деривационном дереве по правилу $(\supset?)$ из подзадач

$$\mathbf{Sk}(A(X))\,\mathbf{Sk}(\forall x\, A(x))\,(\Delta_1\xi)\,\Gamma\,?\,C_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(A(X))\,\mathbf{Sk}(\forall x\, A(x))\,(\Delta_k\xi)\,\Gamma\,?\,C_{i_k}\xi,$$

где $C_{i_1}\,\&\,C_{i_2}\,\&\,\ldots,\,C_{i_k}$ — импликанта нормальной формы псевдоформулы C, $\xi = \eta\circ\sigma$ и θ — комбинация подстановок σ, β_1, $\ldots$, β_k. Здесь следует рассмотреть два подслучая.

(а) Случай $C\in\Gamma$ или $C = \mathbf{Sk}(\forall x\, A(x))$. По индуктивному предположению тогда разрешимы задачи

$$\mathbf{Sk}(\forall x\, A(x))\,(\Delta_1\xi)\,\Gamma\,?\,C_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(\forall x\, A(x))\,(\Delta_k\xi)\,\Gamma\,?\,C_{i_k}\xi \qquad (14)$$

и соответственно β_1, $\ldots$, β_k являются их допустимыми подстановками. По правилу $(\supset?)$ тогда разрешима задача $\mathbf{Sk}(\forall x\, A(x))\,\Gamma?G$ и θ является ее допустимой подстановкой.

(б) Пусть $C = \mathbf{Sk}(A(X))$ и при скулемизации $\forall x\, A(x)$ вместо предметной переменной x вводится локальная метапеременная Z. Положим $\eta_1 = \eta\circ\begin{pmatrix} Z \\ X \end{pmatrix}$. Тогда подстановка η_1 замещает все локальные метапеременные в псевдоформуле $C\begin{pmatrix} X \\ Z \end{pmatrix}$, которая совпадает с $\mathbf{Sk}(\forall x\, A(x))$, на новые глобальные. Тогда ξ есть $\mathbf{MGU}\left(G,\, G'\begin{pmatrix} X \\ Z \end{pmatrix}\eta_1\right)$. По индуктивному предположению тогда разрешимы задачи (14) и соответственно β_1, $\ldots$, β_k являются их допустимыми подстановками. По правилу $(\supset?)$ тогда из разрешимости этих задач следует разрешимость задачи $\mathbf{Sk}(\forall x\, A(x))\,\Gamma?G$, у которой θ является допустимой подстановкой. □

Предложение 5.5. *Если задача* $\mathbf{Sk}(A(y))\,\mathbf{Sk}(\exists x\, A(x))\,\Gamma?G$ *разрешима в* $\mathcal{F}_3$ *и* θ *является ее допустимой подстановкой, где псевдоформулы* $\Gamma\cup\{(\exists x\, A(x)),\, G\}$ *не содержат свободно переменной* y*, то задача* $\mathbf{Sk}(\exists x\, A(x))\,\Gamma?G$ *также является разрешимой и* θ *является ее допустимой подстановкой.*

Доказательство. Утверждение докажем индукцией по высоте деривационного дерева разрешимой задачи $\mathbf{Sk}(A(y))\,\mathbf{Sk}(\exists x\, A(x))\,\Gamma?G$. Во всех

случаях рассуждения носят шаблонный характер. Рассмотрим лишь два случаи строения деривационного дерева. Остальные аналогичны рассмотренным.

1. Рассмотрим случай, когда задача $\mathbf{Sk}(A(y))\,\mathbf{Sk}(\exists x\,A(x))\,\Gamma?G$ является примитивизируемой и θ — ее примитивизирующая подстановка. Будем считать, что $\theta = \mathbf{MGU}(G,\, G'\eta)$, где $G' = C_m$, $\mathbf{NF}(C,\, G') = \&_{i=1}^{i=m} C_i$ и подстановка η замещает все локальные метапеременные в C на новые глобальные. Тогда $C \in \Gamma$, $C = \mathbf{Sk}(A(y))$ или $C = \mathbf{Sk}(\exists x\,A(x))$. Если $C \in \Gamma$ или $C = \mathbf{Sk}(\exists x\,A(x))$, то задача $\mathbf{Sk}(\exists x\,A(x))\,\Gamma?G$ также примитивизируется подстановкой θ. Теперь пусть $C = \mathbf{Sk}(A(y))$. Тогда при скулемизации $\exists x\,A(x)$ для элиминации кванторной приставки $\exists x$ возьмем в качестве скулемовского символа y. Таким образом псевдоформула C будет совпадать с $\mathbf{Sk}(\exists x\,A(x))$ и подстановка θ будет также примитивизировать задачу $\mathbf{Sk}(\exists x\,A(x))\Gamma?G$.

2. Рассмотрим случай, когда задача $\mathbf{Sk}(A(y))\,\mathbf{Sk}(\exists x\,A(x))\,\Gamma?G$ получается в деривационном дереве по правилу $(\supset?)$ из подзадач

$$\mathbf{Sk}(A(y))\,\mathbf{Sk}(\exists x\,A(x))\,(\Delta_1\xi)\,\Gamma\,?\,C_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(A(y))\,\mathbf{Sk}(\exists x\,A(x))\,(\Delta_k\xi)\,\Gamma\,?\,C_{i_k}\xi,$$

где $C_{i_1}\,\&\,C_{i_2}\,\&\,\ldots,\,C_{i_k}$ — импликанта нормальной формы псевдоформулы C, $\xi = \eta\circ\sigma$ и θ — комбинация подстановок σ, β_1, $\ldots$, β_k. Здесь существует два возможных варианта.

(а) Случай $C \in \Gamma$ или $C = \mathbf{Sk}(\exists x\,A(x))$. По индуктивному предположению тогда разрешимы задачи

$$\mathbf{Sk}(\exists x\,A(x))\,(\Delta_1\xi)\,\Gamma\,?\,C_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(\exists x\,A(x))\,(\Delta_k\xi)\,\Gamma\,?\,C_{i_k}\xi \tag{15}$$

и соответственно β_1, $\ldots$, β_k являются их допустимыми подстановками. По правилу $(\supset?)$ тогда разрешима задача $\mathbf{Sk}(\exists x\,A(x))\,\Gamma?G$ и θ является ее допустимой подстановкой.

(б) Теперь пусть $C = \mathbf{Sk}(A(y))$. Тогда при скулемизации $\exists x\,A(x)$ для элиминации кванторной приставки $\exists x$ в качестве скулемовского символа возьмем y. Таким образом псевдоформула C будет совпадать с $\mathbf{Sk}(\exists x\,A(x))$. Тогда разрешимы задачи (15) и соответственно β_1, $\ldots$, β_k являются их допустимыми подстановками. По правилу $(\supset?)$ тогда также разрешима задача $\mathbf{Sk}(\exists x\,A(x))\,\Gamma?G$, у которой θ является допустимой подстановкой. □

Предложение 5.6. *Если* $\mathbf{Sk}(A)\,\mathbf{Sk}(A\vee B)\,\Gamma?C$ *и* $\mathbf{Sk}(B)\,\mathbf{Sk}(A\vee B)\,\Gamma?C$ *суть две разрешимые в* $\mathcal{F}_3$ *задачи, у которых существуют совместимые допустимые подстановки* θ_1 *и* θ_2*, то разрешима также задача*

$\mathbf{Sk}(A \vee B)\,\Gamma?C$ *и комбинация* θ_1 *и* θ_2 *является ее допустимой подстановкой.*

Доказательство. Пусть $\mathbf{Sk}(A)\,\mathbf{Sk}(A \vee B)\,\Gamma?C$ и $\mathbf{Sk}(B)\,\mathbf{Sk}(A \vee B)\,\Gamma?C$ суть две разрешимые в $\mathcal{F}_3$ задачи, у которых существуют совместимые допустимые подстановки θ_1 и θ_2 соответственно. Прежде всего имеем $\mathbf{Sk}(A \vee B) = (A \vee B)$. Возьмем $\eta = \varepsilon$. Тогда $(A \vee B)\eta = (A \vee B)$ и можно считать, что подстановка η заменяет все локальные метапеременные в $A \vee B$, которых нет, на новые глобальные. Тогда разрешимы задачи $\mathbf{Sk}(A\eta)\,(A \vee B)\,\Gamma?C$ и $\mathbf{Sk}(B\eta)\,(A \vee B)\,\Gamma?C$. По правилу $(\vee?)$ тогда разрешима задача $(A \vee B)\,\Gamma?C$ и комбинация θ_1 и θ_2 является ее допустимой подстановкой. Что и требовалось доказать. □

Предложение 5.7. *Если задача* $\mathbf{Sk}(\neg(A \,\&\, B))\,\Gamma\,?\,C$ *разрешима в* $\mathcal{F}_3$ *и* θ *— ее допустимая подстановка, то разрешимы также обе задачи* $\mathbf{Sk}(\neg A)\,\Gamma\,?\,C$, $\Gamma\,\mathbf{Sk}(\neg B)?C$ *и* θ *— их допустимая подстановка.*

Доказательство. Утверждение докажем индукцией по высоте деривационного дерева разрешимой задачи $\mathbf{Sk}(\neg(A \,\&\, B))\,\Gamma\,?\,C$. Рассмотрим лишь два случая строения этого деривационного дерева. Остальные подобны рассмотренным.

1. Если $\mathbf{Sk}(\neg(A \,\&\, B))\,\Gamma\,?\,C$ примитивизируется подстановкой θ, то и $\mathbf{Sk}(\neg A)\,\Gamma\,?\,C$ должна примитивизироваться подстановкой θ.

2. Рассмотрим случай, когда задача $\mathbf{Sk}(\neg(A \,\&\, B))\,\Gamma\,?\,C$ разбивается на подзадачи в деривационном дереве по правилу $(\supset?)$. Здесь два варианта.

(а) Пусть $C = \bot$ и $\mathbf{Sk}(\neg(A \,\&\, B))\,\Gamma\,?\,C$ сводится к задаче $\mathbf{Sk}((A \,\&\, B) \supset \bot)\,\Gamma\,?\,A \,\&\, B$. Тогда дочерняя для нее задача $\mathbf{Sk}(\neg(A \,\&\, B))\,\Gamma\,?\,A$ также разрешима и θ — ее допустимая подстановка. По индуктивному предположению тогда разрешима задача $\mathbf{Sk}(\neg A)\,\Gamma\,?\,A$. По правилу $(\supset?)$ тогда разрешима задача $\mathbf{Sk}(\neg A)\,\Gamma\,?\,\bot$ и θ — ее допустимая подстановка.

(б) Пусть задача $\mathbf{Sk}(\neg(A \,\&\, B))\,\Gamma\,?\,C$ разбивается в деривационном дереве по правилу $(\supset?)$ на подзадачи

$$\mathbf{Sk}(\neg(A \,\&\, B))\,(\Delta_1\xi)\,\Gamma\,?\,F_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(\neg(A \,\&\, B))\,(\Delta_k\xi)\,\Gamma\,?\,F_{i_k}\xi,$$

где $F_{i_1} \,\&\, F_{i_2} \,\&\, \ldots,\, F_{i_k}$ — импликанта нормальной формы псевдоформулы $F \in \Gamma$, $\xi = \eta \circ \sigma$ и θ — комбинация подстановок σ, β_1, $\ldots$, β_k. По индуктивному предположению тогда разрешимы задачи

$$\mathbf{Sk}(\neg A)\,(\Delta_1\xi)\,\Gamma\,?\,F_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(\neg A)\,(\Delta_k\xi)\,\Gamma\,?\,F_{i_k}\xi$$

и соответственно $\beta_1, \ldots, \beta_k$ являются их допустимыми подстановками. По правилу $(\supset?)$ тогда разрешима задача $\mathbf{Sk}(\neg A)\,\Gamma\,?\,C$ и θ является ее допустимой подстановкой. $\square$

Предложение 5.8. *Если задача* $\mathbf{Sk}(\neg(A \supset B))\,\Gamma\,?\,C$ *разрешима в* $\mathcal{F}_3$ *и* θ *— ее допустимая подстановка, то разрешима также задача* $\mathbf{Sk}(A)\,\mathbf{Sk}(\neg B)\,\Gamma\,?\,C$ *и* θ *— ее допустимая подстановка.*

Доказательство. Утверждение докажем индукцией по высоте деривационного дерева разрешимой задачи $\mathbf{Sk}(\neg(A \supset B))\,\Gamma\,?\,C$. Рассмотрим два случая строения этого деривационного дерева. Остальные случаи рассматриваются аналогично.

1. Если задача $\mathbf{Sk}(\neg(A \supset B))\,\Gamma\,?\,C$ примитивизируется подстановкой θ, то и задача $\mathbf{Sk}(A)\,\mathbf{Sk}(\neg B)\,\Gamma\,?\,C$ должна примитивизироваться подстановкой θ.

2. Рассмотрим случай, когда задача $\mathbf{Sk}(\neg(A \supset B))\,\Gamma\,?\,C$ разбивается на подзадачи в деривационном дереве по правилу $(\supset?)$. Рассмотрим два возможных подслучая.

(а) Пусть $C = \bot$ и $\mathbf{Sk}(\neg(A \supset B))\,\Gamma\,?\,C$ сводится к задаче $\mathbf{Sk}((A \supset B) \supset \bot)\,\Gamma\,?\,A \supset B$. Тогда должна быть разрешима дочерняя для нее задача $\mathbf{Sk}(A)\,\mathbf{Sk}(\neg(A \supset B))\,\Gamma\,?\,B$ и θ должна быть ее допустимой подстановкой. По индуктивному предположению тогда также разрешима задача $\mathbf{Sk}(A)\,\mathbf{Sk}(A)\,\mathbf{Sk}(B \supset \bot)\,\Gamma\,?\,B$. По правилу $(\supset?)$ тогда разрешима задача $\mathbf{Sk}(A)\,\mathbf{Sk}(\neg B)\,\Gamma\,?\,\bot$ и θ — ее допустимая подстановка.

(б) Пусть задача $\mathbf{Sk}(\neg(A \supset B))\,\Gamma\,?\,C$ разбивается в деривационном дереве по правилу $(\supset?)$ на подзадачи

$$\mathbf{Sk}(\neg(A \supset B))\,(\Delta_1\xi)\,\Gamma\,?\,F_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(\neg(A \supset B))\,(\Delta_k\xi)\,\Gamma\,?\,F_{i_k}\xi,$$

где $F_{i_1} \,\&\, F_{i_2} \,\&\, \ldots, F_{i_k}$ — импликанта нормальной формы псевдоформулы $F \in \Gamma$, $\xi = \eta \circ \sigma$ и θ — комбинация подстановок σ, $\beta_1, \ldots, \beta_k$. По индуктивному предположению тогда разрешимы задачи

$$\mathbf{Sk}(A)\,\mathbf{Sk}(\neg B)\,(\Delta_1\xi)\,\Gamma\,?\,F_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(A)\,\mathbf{Sk}(\neg B)\,(\Delta_k\xi)\,\Gamma\,?\,F_{i_k}\xi$$

и соответственно $\beta_1, \ldots, \beta_k$ являются их допустимыми подстановками. По правилу $(\supset?)$ тогда разрешима задача $\mathbf{Sk}(A)\,\mathbf{Sk}(\neg B)\,\Gamma\,?\,C$ и θ является ее допустимой подстановкой. $\square$

Предложение 5.9. *Если задача* $\mathbf{Sk}(\neg\forall x\,A(x))\,\Gamma\,?\,C$ *разрешима в* $\mathcal{F}_3$ *и* θ *— ее допустимая подстановка, где псевдоформулы* $\Gamma \cup \{(\forall x\,A(x))\}$

не содержат свободно переменной y, то задача $\mathbf{Sk}(\neg\, A(y))\,\Gamma\,?\,C$ *также разрешима и* θ *— ее допустимая подстановка.*

Доказательство. Утверждение докажем индукцией по высоте деривационного дерева разрешимой задачи $\mathbf{Sk}(\neg\forall x\, A(x))\,\Gamma\,?\,C$. Рассмотрим два случая строения деривационного дерева. Остальные случаи рассматриваются подобным образом.

1. Если задача $\mathbf{Sk}(\neg\forall x\, A(x))\,\Gamma\,?\,C$ примитивизируется подстановкой θ, то и задача $\mathbf{Sk}(\neg\, A(y))\,\Gamma\,?\,C$ должна примитивизироваться подстановкой θ.

2. Рассмотрим случай, когда задача $\mathbf{Sk}(\neg\forall x\, A(x))\,\Gamma\,?\,C$ разбивается на подзадачи в деривационном дереве по правилу $(\supset?)$. Здесь могут быть два варианта.

(а) Пусть $C = \bot$ и $\mathbf{Sk}(\neg\forall x\, A(x))\,\Gamma\,?\,C$ в деривационном дереве сводится к задаче $\mathbf{Sk}((\forall x\, A(x)) \supset \bot)\,\Gamma\,?\,\forall x\, A(x)$. Тогда разрешима дочерняя для нее задача $\mathbf{Sk}(\neg\forall x\, A(x))\,\Gamma\,?\,A(y)$ и θ — ее допустимая подстановка. По индуктивному предположению тогда должна быть разрешима задача $\mathbf{Sk}(\neg\, A(y))\,\Gamma\,?\,A(y)$. По правилу $(\supset?)$ тогда разрешима задача $\mathbf{Sk}(\neg\, A(y))\,\Gamma\,?\,\bot$ и θ — ее допустимая подстановка.

(б) Пусть задача $\mathbf{Sk}(\neg\forall x\, A(x))\,\Gamma\,?\,C$ разбивается в деривационном дереве по правилу $(\supset?)$ на подзадачи

$$\mathbf{Sk}(\neg\forall x\, A(x))\,(\Delta_1\xi)\,\Gamma\,?\,F_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(\neg\forall x\, A(x))\,(\Delta_k\xi)\,\Gamma\,?\,F_{i_k}\xi,$$

где $F_{i_1}\,\&\,F_{i_2}\,\&\,\ldots,\,F_{i_k}$ — импликанта нормальной формы псевдоформулы $F \in \Gamma$, $\xi = \eta\circ\sigma$ и θ — комбинация подстановок σ, β_1, $\ldots$, β_k. По индуктивному предположению тогда разрешимы задачи

$$\mathbf{Sk}(\neg\, A(y))\,(\Delta_1\xi)\,\Gamma\,?\,F_{i_1}\xi,\ \ldots,\ \mathbf{Sk}(\neg\, A(y))\,(\Delta_k\xi)\,\Gamma\,?\,F_{i_k}\xi$$

и соответственно β_1, $\ldots$, β_k являются их допустимыми подстановками. По правилу $(\supset?)$ тогда разрешима задача $\mathbf{Sk}(\neg\, A(y))\,\Gamma\,?\,C$ и θ является ее допустимой подстановкой. □

Предложение 5.10. *Если в* $\mathcal{F}_3$ *разрешима задача* $\mathbf{Sk}(\neg C)\,\Gamma\,?\,\bot$ *и* θ *— ее допустимая подстановка, то разрешима задача* $\Gamma?C$ *и* θ *— ее допустимая подстановка.*

Доказательство. Утверждение докажем индукцией по количеству логических связок и кванторов в C.

1. Если C — атом, дизъюнкция или псевдоформула вида $\exists x\, D(x)$, то утверждение просто совпадает с формулировкой правила декомпозиции $(\bot_{\mathbf{c}})$. Рассмотрим остальные случаи индукции.

2. Пусть $C = D \,\&\, E$ и разрешима задача $\mathbf{Sk}(\neg(D \,\&\, E))\,\Gamma\,?\,\bot$. По предложению 5.7 тогда должны быть разрешимы задачи $\mathbf{Sk}(\neg D)\,\Gamma\,?\,\bot$ и $\mathbf{Sk}(\neg E)\,\Gamma\,?\,\bot$. По предположению индукции отсюда должны быть разрешимы задачи $\Gamma\,?\,D$ и $\Gamma\,?\,E$. Наконец по правилу $(?\&)$ тогда разрешима задача $\Gamma\,?\,C$.

3. Пусть $C = D \supset E$ и разрешима задача $\mathbf{Sk}(\neg(D \supset E))\,\Gamma\,?\,\bot$. По предложению 5.8 тогда разрешима задача $\mathbf{Sk}(D)\,\mathbf{Sk}(\neg E)\,\Gamma\,?\,\bot$. По индуктивному предположению отсюда разрешима задача $\mathbf{Sk}(D)\,\Gamma\,?\,E$. В результате по правилу $(?\supset)$ тогда разрешима задача $\Gamma\,?\,C$.

4. Пусть $C = \forall x\, D(x)$ и разрешима задача $\mathbf{Sk}(\neg\forall x\, D(x))\,\Gamma\,?\,\bot$. По предложению 5.9 тогда должна быть разрешима задача $\mathbf{Sk}(\neg\, D(y))\,\Gamma\,?\,\bot$. По предположению индукции отсюда должна быть разрешима задача $\Gamma\,?\,D(y)$. Наконец по правилу $(?\forall)$ тогда разрешима задача $\Gamma\,?\,C$. □

Будем обозначать через $\overline{\mathcal{F}}_3$ продукционную систему, которая получается из $\mathcal{F}_3$ в результате добавления правил декомпозиции, сформулированных в предложениях 5.2, 5.3, 5.4, 5.5, 5.6, 5.10. Очевидно, что $\overline{\mathcal{F}}_3$ и $\mathcal{F}_3$ эквивалентны с точки зрения объема решаемых задач.

Предложение 5.11. *Если задача* $(\Gamma,\, A)$ *разрешима в продукционной системе* $\overline{\mathcal{P}}_3$*, то задача* $(\mathbf{Sk}(\Gamma)\,?\,A)$ *разрешима в продукционной системе* $\overline{\mathcal{F}}_3$*.*

Доказательство. Пусть θ — подстановка из набора допустимых подстановок задачи $\pi_0 = (\Gamma\,?\,A)$. Мы предполагаем, что подстановка θ является основной. В противном случае мы можем заменить "нерешенные" метапеременные на константы из H_0. Пусть Θ — соответствующее этой подстановке θ деривационное дерево. Пусть процесс расширения языка при построении дерева Θ имеет вид

$$\Omega \subseteq \begin{array}{l} \Theta_0 \subseteq \Theta_1 \subseteq \ldots \subseteq \Theta_n \subseteq \Theta_{n+1} \subseteq \ldots \subseteq \Theta_h = \Theta, \\ \Omega_0 \subseteq \Omega_1 \subseteq \ldots \subseteq \Omega_n \subseteq \Omega_{n+1} \subseteq \ldots \subseteq \Omega_h, \\ H_0 \subseteq H_1 \subseteq \ldots \subseteq H_n \subseteq H_{n+1} \subseteq \ldots \subseteq H_h. \end{array} \tag{16}$$

Индукцией по построению деривационного дерева Θ покажем, как для каждого дерева Θ_n можно построить дерево Θ'_n такое, что корнем дерева Θ'_n является задача $\pi'_0 = (\mathbf{Sk}(\Gamma)\,?\,A)$, дочерние вершины получаются из родительских по одному из правил декомпозиции системы $\overline{\mathcal{F}}_3$,

а листья дерева Θ'_n либо примитивизируются подстановкой θ, либо соответствуют нераскрытым листьям $\pi_n = (\Gamma_n,\, A_n)$ дерева Θ_n и являются задачами π'_n вида $\pi'_n = (\Gamma'_n \,?\, A'_n)$. При этом обозначаемое штрихом формируемое соответствие будет обладать следующим свойством: $A'_n = A_n$ и $\Gamma'_n = \mathbf{Sk}(\Gamma_n)$.

В начале процесса построения дерева Θ' положим $\Gamma'_0 = \mathbf{Sk}(\Gamma)$, $A'_0 = A$, $\pi'_0 = (\Gamma'_0 \,?\, A'_0)$, $\Theta'_0 = \{\pi'_0\}$. Очевидно, что дерево Θ'_0 обладает требуемым свойством.

Покажем как получить дерево Θ'_{n+1} из дерева Θ'_n в зависимости от правила декомпозиции, которое используется при формировании дерева Θ_{n+1} из дерева Θ_n.

1. Если $A_n = \top$, то $\Theta_{n+1} = \Theta_n$. В этом случае листу $\pi_n = (\Gamma_n,\, A_n)$ соответствует лист $\pi'_n = (\Gamma'_n \,?\, \top)$, где $\Gamma'_n = \mathbf{Sk}(\Gamma_n)$, который также является примитивной задачей. При этом задача $\pi'_n = (\Gamma'_n \,?\, \top)$ примитивизируется любой подстановкой, в том числе θ. Положим $\Theta'_{n+1} = \Theta'_n$. При этом дерево Θ'_{n+1} вновь обладает требуемым свойством и имеет вид:

$$\begin{array}{c} (\Gamma'_n \,?\, \top) \\ \vdots \\ (\Gamma'_0 \,?\, A'_0) \end{array}$$

2. Пусть A_n — атомарная псевдоформула или $\bot$, лист $\pi_n = (\Gamma_n,\, A_n)$ примитивизируется подстановкой θ, $A_n\theta = E\theta$, $E \in \Gamma_n$ и лист $\pi'_n = (\Gamma'_n \,?\, A'_n)$ соответствует листу π_n, где $A'_n = A_n$ и $\Gamma'_n = \mathbf{Sk}(\Gamma_n)$. Тогда $E' = \mathbf{Sk}(E) = E$. Отсюда задача $\pi'_n = (\Gamma'_n \,?\, A'_n)$ примитивизируется подстановкой θ. Положим $\Theta'_{n+1} = \Theta'_n$. При этом дерево Θ'_{n+1} вновь обладает требуемым свойством и имеет вид:

$$\begin{array}{c} (\Gamma'_n \,?\, A_n) \\ \vdots \\ (\Gamma'_0 \,?\, A'_0) \end{array}$$

3. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции $(\to \vee_i)$. Следовательно имеем $A_n = (B_1 \vee B_2)$, $A'_n = A_n$. Положим $A'_{n+1} = A_{n+1} = B_i$ и $\Gamma'_{n+1} = \Gamma'_n$. Дерево Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения правила $(?\vee_i)$:

$$\begin{array}{c} \dfrac{(\Gamma'_n \,?\, B_i)}{(\Gamma'_n \,?\, B_1 \vee B_2)} \\ \vdots \\ (\Gamma'_0 \,?\, A'_0) \end{array}$$

При этом если θ является допустимой подстановкой для $(\Gamma'_n \,?\, B_i)$, то θ является допустимой подстановкой для задачи $(\Gamma'_n \,?\, B_1 \vee B_2)$.

4. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции $(\to \exists)$. Это значит, что $A'_n = A_n = (\exists x\, B(x))$. Положим $A'_{n+1} = A_{n+1} = B(X)$ и $\Gamma'_{n+1} = \Gamma'_n$. Дерево Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения правила $(?\exists)$:

$$\dfrac{(\Gamma'_n \,?\, B(X))}{(\Gamma'_n \,?\, \exists x\, B(x))}$$
$$\vdots$$
$$(\Gamma'_0 \,?\, A'_0)$$

Очевидно, что если θ является допустимой подстановкой для задачи $(\Gamma'_n \,?\, B(X))$, то θ является допустимой подстановкой для $(\Gamma'_n \,?\, \exists x\, B(x))$.

5. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции $(\to \&)$. Это значит, что $A'_n = A_n = (B_1 \,\&\, B_2)$. Дерево Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения правила декомпозиции $(?\&)$:

$$\dfrac{(\Gamma'_n \,?\, B_1) \qquad (\Gamma'_n \,?\, B_2)}{(\Gamma'_n \,?\, B_1 \,\&\, B_2)}$$
$$\vdots$$
$$(\Gamma'_0 \,?\, A'_0)$$

При этом если θ является допустимой подстановкой для обеих задач $(\Gamma'_n \,?\, B_1)$ и $(\Gamma'_n \,?\, B_2)$, то θ — допустимая подстановка для $(\Gamma'_n \,?\, B_1 \,\&\, B_2)$.

6. Пусть Θ_{n+1} получается из Θ_n в результате применения продукции $(\to\supset)$. Это значит, что $A'_n = A_n = (B_1 \supset B_2)$. Положим $A'_{n+1} = A_{n+1} = B_2$, $B'_1 = \mathbf{Sk}(B_1)$ и $\Gamma'_{n+1} = \Gamma'_n \cup \{B'_1\} = \mathbf{Sk}(\Gamma_{n+1})$. Дерево Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения правила декомпозиции $(? \supset)$:

$$\dfrac{(\mathbf{Sk}(B_1)\, \Gamma'_n \,?\, B_2)}{(\Gamma'_n \,?\, B_1 \supset B_2)}$$
$$\vdots$$
$$(\Gamma'_0 \,?\, A'_0)$$

Очевидно, что если θ является допустимой подстановкой для задачи $(B'_1\, \Gamma'_n \,?\, B_2)$, то θ является допустимой подстановкой для $(\Gamma'_n \,?\, B_1 \supset B_2)$.

7. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции $(\to \forall)$. Это значит, что $A'_n = A_n = (\forall x\, B(x))$. Положим

$A'_{n+1} = A_{n+1} = B(y)$ и $\Gamma'_{n+1} = \Gamma'_n$. Дерево Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения правила (?$\forall$):

$$\dfrac{(\Gamma'_n \,?\, B(y))}{(\Gamma'_n \,?\, \forall x\, B(x))}$$
$$\vdots$$
$$(\Gamma'_0 \,?\, A'_0)$$

Очевидно, что если θ является допустимой подстановкой для задачи $(\Gamma'_n \,?\, B(y))$, то θ является допустимой подстановкой для $(\Gamma'_n \,?\, \forall x\, B(x))$.

8. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции $(\&{\to})$. Это значит, что $A'_n = A_n$, $C = (B_1 \,\&\, B_2) \in \Gamma_n$, $\Gamma'_n = \mathbf{Sk}(\Gamma_n)$, $B'_i = \mathbf{Sk}(B_i)$, $C' = \mathbf{Sk}(C) = \mathbf{Sk}(B_1 \,\&\, B_2) = \mathbf{Sk}(B_1) \,\&\, \mathbf{Sk}(B_2) = (B'_1 \,\&\, B'_2) \in \Gamma'_n$, $\Gamma_{n+1} = \Gamma_n \cup \{B_1,\, B_2\}$. Заметим, что результатом применения к задаче $(\mathbf{Sk}(B_1 \,\&\, B_2)\, \Gamma'_n \,?\, A'_n)$ сформулированного в предложении 5.2 правила декомпозиции будет $(\mathbf{Sk}(B_1)\, \mathbf{Sk}(B_2)\, \mathbf{Sk}(B_1 \,\&\, B_2)\, \Gamma'_n \,?\, A'_n)$. Положим $\Gamma'_{n+1} = \Gamma'_n \cup \{B'_1,\, B'_2\}$, $A'_{n+1} = A'_n = A_n$. Дерево Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения этого правила декомпозиции:

$$\dfrac{(\mathbf{Sk}(B_1)\, \mathbf{Sk}(B_2)\, \mathbf{Sk}(B_1 \,\&\, B_2)\, \Gamma'_n \,?\, A_n)}{(\mathbf{Sk}(B_1 \,\&\, B_2)\, \Gamma'_n \,?\, A_n)}$$
$$\vdots$$
$$(\Gamma'_0 \,?\, A'_0)$$

По предложению 5.2 заключаем, что если θ является допустимой подстановкой для задачи $(B'_1\, B'_2\, \Gamma'_n \,?\, A_n)$, то θ является допустимой подстановкой также и для задачи $(\Gamma'_n \,?\, A_n)$.

9. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции $(\supset{\to})$. Это значит, что $A'_n = A_n$, $C = (B_1 \supset B_2) \in \Gamma_n$, $\Gamma'_n = \mathbf{Sk}(\Gamma_n)$, $B'_1 = B_1$, $B'_2 = \mathbf{Sk}(B_2)$, $C' = \mathbf{Sk}(C) = \mathbf{Sk}(B_1 \supset B_2) = B_1 \supset \mathbf{Sk}(B_2) = (B'_1 \supset B'_2) \in \Gamma'_n$. Тогда задачи $(\mathbf{Sk}(B_1 \supset B_2)\, \Gamma'_n \,?\, B_1)$ и $(\mathbf{Sk}(B_2)\, \mathbf{Sk}(B_1 \supset B_2)\, \Gamma'_n \,?\, A'_n)$ являются дочерними для родительской задачи $(\mathbf{Sk}(B_1 \supset B_2)\, \Gamma'_n \,?\, A'_n)$, если к последней применить правило декомпозиции, формулировка которого содержится в предложении 5.3. Дерево Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения этого правила декомпозиции:

$$\dfrac{(\mathbf{Sk}(B_1 \supset B_2)\, \Gamma'_n \,?\, B_1) \qquad (\mathbf{Sk}(B_2)\, \mathbf{Sk}(B_1 \supset B_2)\, \Gamma'_n \,?\, A_n)}{(\mathbf{Sk}(B_1 \supset B_2)\, \Gamma'_n \,?\, A_n)}$$
$$\vdots$$
$$(\Gamma'_0 \,?\, A'_0)$$

Из предложения 5.3 следует, что если θ является допустимой подстановкой для задач $(B'_2\,\Gamma'_n\,?\,A_n)$ и $(\Gamma'_n\,?\,B_1)$, то θ является допустимой подстановкой также и для задачи $(\Gamma'_n\,?\,A_n)$.

10. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции $(\forall\ \to)$. Это значит, что $A'_n = A_n$, $C = (\forall x\,B(x)) \in \Gamma_n$, $\Gamma'_n = \mathbf{Sk}(\Gamma_n)$, $C' = \mathbf{Sk}(C) = \mathbf{Sk}(\forall x\,B(x)) \in \Gamma'_n$, $\Gamma_{n+1} = \Gamma_n \cup \{B(X)\}$. Тогда задача $(\mathbf{Sk}(B(X))\,\mathbf{Sk}(\forall x\,B(x))\,\Gamma'_n\,?\,A'_n)$ является результатом применения к задаче $(\mathbf{Sk}(\forall x\,B(x))\,\Gamma'_n\,?\,A'_n)$ сформулированного в предложении 5.4 правила декомпозиции. Положим $\Gamma'_{n+1} = \Gamma'_n \cup \{\mathbf{Sk}(B(X))\}$. Дерево Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения этого правила декомпозиции:

$$\dfrac{(\mathbf{Sk}(B(X))\,\mathbf{Sk}(\forall x\,B(x))\,\Gamma'_n\,?\,A_n)}{(\mathbf{Sk}(\forall x\,B(x))\,\Gamma'_n\,?\,A_n)}$$
$$\vdots$$
$$(\Gamma'_0\,?\,A'_0)$$

По предложению 5.4 заключаем, что если θ является допустимой подстановкой для задачи $(\mathbf{Sk}(B(X))\,\Gamma'_n\,?\,A_n)$, то θ является допустимой подстановкой также и для задачи $(\Gamma'_n\,?\,A_n)$.

11. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции $(\exists\ \to)$. Это значит, что $A'_n = A_n$, $C = (\exists x\,B(x)) \in \Gamma_n$, $\Gamma'_n = \mathbf{Sk}(\Gamma_n)$, $C' = \mathbf{Sk}(C) = \mathbf{Sk}(\exists x\,B(x)) \in \Gamma'_n$, $\Gamma_{n+1} = \Gamma_n \cup \{B(y)\}$. Тогда задача $(\mathbf{Sk}(B(y))\,\mathbf{Sk}(\exists x\,B(x))\,\Gamma'_n\,?\,A'_n)$ является результатом применения к задаче $(\mathbf{Sk}(\exists x\,B(x))\,\Gamma'_n\,?\,A'_n)$ сформулированного в предложении 5.5 правила декомпозиции. Положим $\Gamma'_{n+1} = \Gamma'_n \cup \{\mathbf{Sk}(B(y))\}$. Дерево Θ'_{n+1} построим из дерева Θ'_n, применив указанное правило декомпозиции:

$$\dfrac{(\mathbf{Sk}(B(y))\,\mathbf{Sk}(\exists x\,B(x))\,\Gamma'_n\,?\,A_n)}{(\mathbf{Sk}(\exists x\,B(x))\,\Gamma'_n\,?\,A_n)}$$
$$\vdots$$
$$(\Gamma'_0\,?\,A'_0)$$

Из предложения 5.5 следует, что если θ является допустимой подстановкой для задачи $(\mathbf{Sk}(B(y))\,\Gamma'_n\,?\,A_n)$, то θ является допустимой подстановкой также и для задачи $(\Gamma'_n\,?\,A_n)$.

12. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции $(\vee\ \to)$. Это значит, что $A'_n = A_n$, $C = (B_1 \vee B_2) \in \Gamma_n$, $\Gamma'_n = \mathbf{Sk}(\Gamma_n)$, $B'_i = \mathbf{Sk}(B_i)$, $C' = \mathbf{Sk}(C) = \mathbf{Sk}(B_1 \vee B_2) = (B_1 \vee B_2) = C \in \Gamma'_n$. Тогда задачи $(\mathbf{Sk}(B_1)\,\mathbf{Sk}(B_1 \vee B_2)\,\Gamma'_n\,?\,A'_n)$ и $(\mathbf{Sk}(B_2)\,\mathbf{Sk}(B_1 \vee B_2)\,\Gamma'_n\,?\,A'_n)$ являются дочерними для задачи $(\mathbf{Sk}(B_1 \vee B_2)\,\Gamma'_n\,?\,A'_n)$ после применения сформулированного в предложении 5.6 правила декомпозиции. Дерево

Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения указанного правила декомпозиции:

$$\dfrac{(\mathbf{Sk}(B_1)\,\mathbf{Sk}(B_1 \vee B_2)\,\Gamma'_n\,?\,A_n) \qquad (\mathbf{Sk}(B_2)\,\mathbf{Sk}(B_1 \vee B_2)\,\Gamma'_n\,?\,A_n)}{(\mathbf{Sk}(B_1 \vee B_2)\,\Gamma'_n\,?\,A_n)}$$
$$\vdots$$
$$(\Gamma'_0\,?\,A'_0)$$

Из предложения 5.6 следует, что если θ является допустимой подстановкой для задач $(B'_1\,\Gamma'_n\,?\,A_n)$ и $(B'_2\,\Gamma'_n\,?\,A_n)$, то θ является допустимой подстановкой также и для задачи $(\Gamma'_n\,?\,A_n)$.

13. Пусть Θ_{n+1} получается из Θ_n в результате применения правила декомпозиции $(\bot_{\mathbf{c}})$. Это значит, что $A'_n = A_n$, $B = (\neg\, A_n) \in \Gamma_n$, $\Gamma'_n = \mathbf{Sk}(\Gamma_n)$, $B' = \mathbf{Sk}(B) = \mathbf{Sk}(\neg\, A_n) = (\neg\, A_n) \in \Gamma'_n$, $\Gamma_{n+1} = \Gamma_n \cup \{(\neg\, A_n)\}$, $A_{n+1} = \bot$. Тогда задача $((\neg\, A_n)\,\Gamma'_n\,?\,\bot)$ является результатом применения к задаче $(\Gamma'_n\,?\,A'_n)$ сформулированного в предложении 5.10 правила декомпозиции. Положим $\Gamma'_{n+1} = \Gamma'_n \cup \{(\neg\, A_n)\}$, $A'_{n+1} = \bot$. Дерево Θ'_{n+1} сформируем из дерева Θ'_n с помощью применения этого правила декомпозиции:

$$\dfrac{((\neg\, A_n)\,\Gamma'_n\,?\,\bot)}{(\Gamma'_n\,?\,A_n)}$$
$$\vdots$$
$$(\Gamma'_0\,?\,A'_0)$$

По предложению 5.10 заключаем, что если θ является допустимой подстановкой для задачи $((\neg\, A_n)\,\Gamma'_n\,?\,\bot)$, то θ является допустимой подстановкой также и для задачи $(\Gamma'_n\,?\,A_n)$. Тем самым рассмотрение случая $(\bot_{\mathbf{c}})$ завершено.

Для построения Θ_{n+1}, а затем Θ'_{n+1}, в соответствии с применяемой стратегией мы выбирали лист $\pi_n = (\Gamma_n,\, A_n) \in \Theta_n$, у которого порождали дочерние разрешимые вершины, соответствующие подстановке θ и обеспечивающие разрешимость вершины π_n. В результате было сформировано дерево Θ'_{n+1}, которое получается из дерева Θ'_n в результате замены листа $\pi'_n = (\Gamma'_n\,?\,A'_n)$, соответствующего листу π_n дерева Θ_n, на некоторое новое поддерево с корнем π'_n. При этом каждая вершина ρ полученного дерева Θ'_{n+1} обладает следующим свойством: во-первых, если ρ является листом дерева Θ'_{n+1}, который не соответствует нераскрытому листу дерева Θ_{n+1}, то $\rho\theta$ является примитивной задачей; во-вторых, если θ является допустимой для вершин $\rho_1, \ldots, \rho_k$, всех дочерних у вершины ρ в дереве Θ'_{n+1}, то θ является также допустимой и для ρ. На заключительном шаге этого процесса мы получим дерево $\Theta' = \Theta'_h$ такое,

что θ является допустимой подстановкой задачи $(\mathbf{Sk}(\Gamma)\,?\,A)$ и Θ' является соответствующим этой подстановке θ деривационным деревом в $\overline{\mathcal{F}}_3$. Предложение 5.11 доказано. □

Теорема о полноте. *Если секвенция* $\Gamma \to A$ *выводима в исчислении* $\mathcal{S}_3$*, то задача* $(\mathbf{Sk}(\Gamma)\,?\,A)$ *разрешима в* $\mathcal{F}_3$*.*

Доказательство. Пусть секвенция $\Gamma \to A$ выводима в исчислении $\mathcal{S}_3$. По теореме о полноте $\mathcal{P}_3$ тогда задача $(\Gamma,\, A)$ разрешима в $\mathcal{P}_3$. Продукционные системы $\overline{\mathcal{P}}_3$ и $\overline{\mathcal{F}}_3$ получаются соответственно из $\mathcal{P}_3$ и $\mathcal{F}_3$ за счет добавления некоторых, причем допустимых, правил декомпозиции. Это значит, что задача $(\Gamma,\, A)$ разрешима и в $\overline{\mathcal{P}}_3$. По предложению 5.11 тогда задача $(\mathbf{Sk}(\Gamma)\,?\,A)$ разрешима в $\overline{\mathcal{F}}_3$, а значит и в $\mathcal{F}_3$. Что и требовалось доказать. □

Автор выражает искреннюю признательность проф. В.Ю. Попову за советы, замечания и обсуждение вопросов, которые возникали при построении этой теории.

Список литературы

[1] Вторушин Ю.И., “О поиске вывода в системе натуральной дедукции логики предикатов”, *Интеллектуальные системы*, **13**:3 (2009), 263–288.

[2] Чень Ч., Ли Р., *Математическая логика и автоматическое доказательство теорем*, «Наука», Москва, 1983, 360 с.

[3] Конев Б.Ю., Жебелеан Т., “Метод подъема решений для работы с метапеременными в системе TH∃OREM∀”, *Записки научных семинаров ПОМИ*, **293** (2002), 94–117

[4] Sieg W., Byrnes J., “Normal natural deduction proofs (in classical logic)”, *Studia Logica*, **60**:1 (1998), 67–106

[5] Okhotnikov O., “A new sequent calculus for automated proof search”, *Applied Mathematical Sciences*, **8**:100 (2014), 4977–4984

[6] Большакова Е.И., Мальковский М.Г., Пильщиков В.Н., *Искусственный интеллект: методы и алгоритмы эвристического поиска*, МГУ, Москва, 2002, 81 с.

[7] Маслов С.Ю., *Теория дедуктивных систем и ее применения*, «Наука», Москва, 1986, 136 с.

[8] Lyaletski A., “Admissibility, compatibility, and deducibility in first-order sequent logics”, *Computer Science Journal of Moldova*, **23**:3 (2015), 289–303

[9] Degtyarev A., Voronkov A., “Kanger’s choices in automated reasoning”, *Collected Papers of Stig Kanger with Essays on His Life and Work*, 2001, 53–68

[10] Минц Г.Е., "Теорема Эрбрана", *Математическая теория логического вывода*, 1967, 311–350.

About proof-search in classical natural deduction calculus using partial skolemization

Okhotnikov O.A.

Automated proof search in single-conclusion sequential variant of classical predicate calculus is considered. In this algorithm, metavariables and partial skolemization are used. Theorems of soundness and completeness for the considered algorithm are proved.

Keywords: automated theorem proving, mechanical proof search, first order language, predicate calculus, sequent calculus, production system, artificial intelligence.

Часть 3.
Математические модели

О функциональной системе полиномов с рациональными коэффициентами

Алексиадис Н.Ф.

[1]В настоящей статье исследуется проблема полноты для полиномиальных функций с рациональными коэффициентами, а также задачи функционального характера, порожденные ее решением, а именно: изучение структуры замкнутых и предполных классов, задача о базисах полных систем[2]. В частности, доказано, что

1) система функций является полной тогда и только тогда, когда она целиком не содержится ни в одном предполном классе;

2) мощность множества всех предполных классов равна континууму.

3) существует полная система, не имеющая базиса.

Ключевые слова: полином, функциональная система, проблема полноты, рациональная функция, замкнутые классы.

1. Предварительные сведения

С целью полноты изложения и независимости материала приведем некоторые сведения из теории функциональных систем, необходимые для дальнейшего изложения[3]. При изложении материала в основном используется терминология книг [4] и [5].

Пусть G – некоторое (конечное или бесконечное) множество, содержащее не менее двух элементов.

Обозначим через F_G множество всех функций $f(x_1, x_2, ..., x_n)$, переменные которых определены на множестве G и сами функции принимают значения из этого же множества G. Предполагается, что множество

[1]aleksiadis@yandex.ru

[2]Ранее автором было исследовано функциональные системы полиномов с натуральными и целыми коэффициентами (см. [1] - [2])

[3]Читатели, знакомые с основными понятиями теории функциональных систем, могут пропустить этот раздел.

F_G содержит и все функции от нулевого числа переменных, т.е. функции, являющиеся просто элементами (константами) множества G.

Пусть $F_G^{(n)}$ множество всех n-местных функций из F_G $(n = 0, 1, 2, ...)$.

Очевидно, что $F_G^{(0)} = G$ и $F_G = \cup_{n=0}^{\infty} F_G^{(n)}$.

Говорят, что *функция* $f(x_1, ..., x_{i-1}, x_i, x_{i+1}, ..., x_n)$ *существенно зависит от переменной* x_i, если существуют такие два набора

$$(c_1, ..., c_{i-1}, a, c_{i+1}, ..., c_n) \text{ и } (c_1, ..., c_{i-1}, b, c_{i+1}, ..., c_n)$$

значений переменных $x_1, ..., x_{i-1}, x_i, x_{i+1}, ..., x_n$, что

$$f(c_1, ..., c_{i-1}, a, c_{i+1}, ..., c_n) \neq f(c_1, ..., c_{i-1}, a, b_{i+1}, ..., c_n).$$

В этом случае мы говорим, что x_i является *существенной переменной функции* $f(x_1, ..., x_{i-1}, x_i, x_{i+1}, ..., x_n)$.

Если x_i не является существенной переменной $f(x_1, ..., x_i, ..., x_n)$, то она называется *фиктивной переменной функции* $f(x_1, ..., x_i, ..., x_n)$.

Пусть $f(x_1, ..., x_{i-1}, x_i, x_{i+1}, ..., x_n)$ и $g(x_1, ..., x_{i-1}, x_{i+1}, ..., x_n)$ две произвольные функции из F_G и пусть x_i фиктивная переменная функции $f(x_1, ..., x_{i-1}, x_i, x_{i+1}, ..., x_n)$. Если для любых $c_1, ..., c_{i-1}, c_i, c_{i+1}, ..., c_n$ значений переменных $x_1, ..., x_{i-1}, x_i, x_{i+1}, ..., x_n$ имеем

$$f(c_1, ..., c_{i-1}, c_i, c_{i+1}, ..., c_n) = g(c_1, ..., c_{i-1}, c_{i+1}, ..., c_n),$$

то говорят, что функция $g(x_1, ..., x_{i-1}, x_{i+1}, ..., x_n)$ получается из функции $f(x_1, ..., x_{i-1}, x_i, x_{i+1}, ..., x_n)$ *удалением (изъятием) фиктивной переменной* x_i и, наоборот, функция $f(x_1, ..., x_{i-1}, x_i, x_{i+1}, ..., x_n)$ получена из функции $g(x_1, ..., x_{i-1}, x_{i+1}, ..., x_n)$ *добавлением фиктивной переменной* x_i.

Две функции называются *равными*, если одна из них может быть получена из другой путем добавления или изъятия некоторых фиктивных переменных.

Замечание 1. *В дальнейшем будем считать, что вместе с функцией* f *заданы и все равные ей функции, т.е. функции рассматриваем с точностью до фиктивных переменных.*

Замечание 2. *Если дана конечная система функций* $f_1, f_2, ..., f_m$ *(где* $m \geq 1$*), то можно считать, что все они зависят от одних и тех же переменных* $x_1, x_2, ..., x_n$*, т.е. имеют вид*

$$f_1(x_1, x_2, ..., x_n), f_2(x_1, x_2, ..., x_n), ..., f_m(x_1, x_2, ..., x_n).$$

Замечание 3. *Если дана функция, отличная от константы, то путем отождествления переменных из нее можно получить равную ей функцию, все переменные которой являются существенными.*

Алгебра $\mathbf{F}_G = (F_G, O)$, где F_G – некоторое множество функций (не обязательно все) из G в G, а O – некоторое множество операций, при этом эти операции замкнуты относительно множества F_G, называется *функциональной системой* (*ф.с.*) $\mathbf{F}_G$.

Для произвольного подмножества M множества F_G обозначим через $I_O(M)$ множество всех функций из F_G, которые получаются из M с помощью конечного числа применения операций из O. Множество $I_O(M)$ называется *замыканием множества* M.

С замыканием множества естественным образом можно связать оператор замыкания.

Оператор, переводящий произвольное подмножество M множества F_G на множество $I_O(M)$, называется *оператором замыкания* (*относительно операций из* O) и обозначается через I_O (или через $[\,]$, если множество операций O известно и из контекста понятно о каких операциях идет речь, т.е. вместо $I_O(M)$ пишем $[M]$).

Легко заметить, что для любых подмножеств M, M_1, M_2 множества F_G :

- $M \subseteq I_O(M)$;
- $I_O(I_O(M)) = M$;
- Если $M_1 \subseteq M_2$, то $I_O(M_1) \subseteq I_O(M_2)$.

Множество $M (M \subseteq F_G)$ называется (*функционально*) *замкнутым*, если $I_O(M) = M$.

Замкнутое множество принято называть *замкнутым классом*.

Ясно, что

- $I_O(M)$ *является замкнутым классом* $(\forall M \subseteq F_G)$;
- *пустое множество является замкнутым классом*;
- *ножество* F_G *также является замкнутым классом.*

Пусть V – произвольное подмножество множества G. Обозначим через $U(V)$ множество всех таких функций $f(x_1, x_2, ..., x_n)$ из F_G, что

$$f(c_1, c_2, ..., c_n) \in V, \text{ если } c_1, c_2, ..., c_n \in V.$$

В этом случае $f(x_1, x_2, ..., x_n)$ называется *функцией, сохраняющей множество* V, а $U(V)$ – *классом, сохраняющим множество* V.

Замечание 4. *Легко заметить, что для любого непустого собственного подмножества* V *множества* G *(т.е.* $V \neq \varnothing$ *и* $V \neq F_G$*) выполнено:*

- $U(V) \neq \varnothing$;
- $U(V) \neq F_G$;
- $V \subseteq U(V)$;
- $I_O(U(V)) = U(V)$;
- $U^{(0)}(V) = V$, *где* $U^{(0)}(V)$ – *множество всех нульместных функций, т.е. констант из* $U(V)$.

Более того, для любых двух подмножеств V_1 *и* V_2 *множества* G *выполнено:*
$U(V_1) \neq U(V_2)$, *если* $V_1 \neq V_2$.

Подмножество M множества F_G называется (*функционально*) *полным* в $\mathbf{F}_G$, если $I_O(M) = F_G$.

Полное множество принято называть *полной системой*.

Из замечания (4) следует справедливость следующего утверждения. Если множество $M (M \subseteq G)$ содержит полную систему в F_G, то M также является полной системой в F_G.

Система функций $B (B \subseteq F_G)$ называется *базисом ф.с.* $\mathbf{F}_G$, если B полна в $\mathbf{F}_G$, но всякая ее собственная подсистема не является полной в $\mathbf{F}_G$.

Подмножество M множества F_G называется *предполным* (*максимальным*) *классом в* $\mathbf{F}_G$, если $I_O(M) \neq F_G$, но для любой функции f из $F_G \setminus M$ выполнено $I_O(M \cup \{f\}) = F_G$.

Очевидно, что *предполный класс является замкнутым классом.*

Замечание 5. *Отметим, что для доказательства предполноты в* $\mathbf{F}_N$ *множества* $M (M \subseteq F_G)$ *нужно показать, что*

- $M \neq \varnothing$ *и* $M \neq F_G$;
- M *замкнутый класс;*
- *для любого* f *из* $F_G \setminus M$ *система* $M \cup \{f\}$ *полна в* F_G.

Если T произвольный замкнутый класс в $\mathbf{F}_G$, то аналогичным образом определяются замкнутый класс в T, предполный класс в T, полная система в T и базис множества T.

Система K замкнутых подмножеств множества F_G называется *критериальной системой в* $\mathbf{F}_G$, если любое множество $M(M \subseteq F_G)$ является полным в $\mathbf{F}_G$ тогда и только тогда, когда оно целиком не содержится ни в одном классе системы K.

В ф.с. $\mathbf{F}_G$ критериальной системой является, например, множество всех замкнутых классов, отличных от всего F_G. В общем случае последняя критериальная система является "избыточной". Это позволяет перейти к рассмотрению более "экономных" критериальных систем и с этой точки зрения может быть уточнено строение критериальной системы.

Критериальная система называется *приведенной*, если она не содержит собственных подсистем, являющихся критериальными.

Замечание 6. *Следует отметить, что в функциональной системе* $\mathbf{F}_G$ *приведенная система, если она существует, определяется однозначно и состоит из всех предполных классов в* F_G *(см. [4].)*

Замкнутый класс $T(T \subseteq F_G)$ называется *конечно-порожденным*, если в T существует конечная полная система.

Функциональная система $\mathbf{F}_G$ является конечно-порожденной, если F_G конечно-порожденный класс.

$\mathbf{F}_G$ называется *функциональной системой счетной мощности*, если F_G счетное множество.

В ф.с. $\mathbf{F}_G$ множество $M(M \subseteq F_G)$ называется полной системой относительно множества D, если $D \subseteq M$ и M полная система в $\mathbf{F}_G$.

И, наконец, приведем одно (нужное в дальнейшем) утверждение.

Если в функциональной системе каждый замкнутый класс содержится в некотором предполном классе, то множество всех предполных классов образует (приведенную) критериальную систему (см.[3]).

2. Постановка задачи

Одной из основных проблем в теории функциональных систем является *проблема полноты, состоящая в описании всех подмножеств данного множества* F_G, *которые являются полными в* $\mathbf{F}_G$.

В теории функциональных систем выделяют два подхода к решению проблемы полноты: алгоритмический и алгебраический. В первом случае

ставится вопрос о существовании алгоритма, устанавливающего полноту или неполноту заданных систем функций. Во втором случае задача заключается в получении критериев (т.е. необходимых и достаточных условий) полноты.

Изучение проблемы полноты осуществлялось путем исследования конкретных функциональных систем. Во многих этих конкретных системах решение проблемы полноты было сведено к описанию всех предполных классов в ней, суть которого заключается в следующем: данная система функций является полной тогда и только тогда, когда она целиком не содержится ни в одном из предполных классов. Исторически сложилось так, что метод решения проблемы полноты в терминах предполных классов стал одним из основных для функциональных систем.

Целью настоящей работы является исследование проблемы полноты для функциональной системы полиномов с рациональными коэффициентами[4]; более подробно, решение следующего круга задач.

1) *Полнота систем функций*:

   - *является ли множество всех предполных классов критериальной системой*?
   - *найти число предполных классов*;
   - *найти предполные классы (по возможности).*

2) *Задача о базисах*:

   - *имеет ли базис каждая полная система*?
   - *мощность базисов.*

3. Определение функциональной системы полиномов с рациональными коэффициентами

В этом разделе мы приводим определение функциональной системы полиномиальных функций с рациональными коэффициентами. Но сначала несколько стандартных обозначений.

N – множество всех натуральных чисел (включая число 0).

Z – множество всех целых чисел.

Q – множество всех рациональных чисел.

[4]Определение этой системы см. ниже в разделе 1.3.

$|A|$ – мощность множества A.

c_0 – мощность счетного множества.

c – мощность континуума.

$\equiv$ – обозначим, по определению, тождественно равно.

Для удобства изложения полагаем, что $0^0 = 1$.

$\square$ – квадрат будет обозначать "конец доказательства".

Выражение вида $cx_1^{k_1}x_2^{k_2}...x_n^{k_n}$, где $n, k_1, k_2, ..., k_n \in N$, а $c \in Q$ называется *мономом с рациональным коэффициентом*, зависящим от n переменных $x_1, x_2, ..., x_n$; при этом, когда $n = 0$, тогда заданный моном является просто константой c, т.е. мономом с рациональным коэффициентом, зависящим от 0-го числа переменных.

Конечная сумма мономов с рациональным коэффициентами называется *полиномом с рациональным коэффициентом.*

Обозначим через $P_Q^{(n)}(n = 0, 1, 2, ...)$ множество всех функций из Q^n в Q, задаваемых полиномами с рациональными коэффициентами и зависящих от n переменных, а через P_Q множество всех функций, задаваемых полиномами с рациональными коэффициентами, т.е. $P_Q = \cup_{n=0}^{\infty} P_Q^{(n)}$. Эти функции назовем *полиномиальными функциями с рациональными коэффициентами* или просто *рп-функциями*[5].

Замечание 7. *Для простоты вместо фраз "функция задаваемая полиномом" и "полиномиальная функция"мы будем употреблять просто "полином", т.е. мы отождествляем формулу и функцию, задаваемую этой формулой.*

В качестве множества операций O мы рассматриваем операции суперпозиции. Операции суперпозиции включают в себя:

- перестановку переменных;
- отождествления переменных;
- переименования переменных (без отождествления);
- введение фиктивной переменной;
- удаление фиктивной переменной;
- подстановку одной функции в другую.

[5]Символы 'p' и 'n' – это первые буквы соответственно слов "рациональный" и "полином".

Поскольку любая суперпозиция функций из P_Q является опять функцией из P_Q, то мы вправе рассмотреть пару (P_Q, O), где O – множество операций суперпозиции над полиномиальными функциями из P_Q; эта пара является функциональной системой, которую мы назовем *функциональной системой полиномиальных функций с рациональными коэффициентами* и обозначим ее через $\mathbf{F}_Q$.

Теорема 1. $\mathbf{F}_Q$ *является функциональной системой счетной мощности.*

Д о к а з а т е л ь с т в о. Это следует из того, что существует всего счетное число полиномов с рациональными коэффициентами. □

Теорема 2. *В функциональной системе* $\mathbf{F}_Q$ *система функций*

$$B = \{x - y, xy, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\},$$

где p *– любое простое число, является полной системой; более того она является и базисом ф.с.* $\mathbf{F}_Q$.

Д о к а з а т е л ь с т в о. Обозначим через $f(x, y) \equiv x - y$ и $g(x, y) \equiv xy$.

Сначала получим все константы из $P_Q^{(0)}$. Имеем

$$f(x, x) = 0; h_1(x) \equiv f(0, x) = -x;$$

$$h_2(x, y) \equiv f(x, h_1(y)) = x + y; h_2(\frac{1}{2}, \frac{1}{2}) = 1,$$

$$h_2(1, 1) = 2, h_2(1, 2) = 3, h_2(1, 3) = 4, ...$$

$$h_1(1) = -1, h_1(2) = -2, h_1(3) = -3, ...$$

Следовательно, мы имеем все целые числа.

Далее, пусть c – произвольное число из Q, отличное от 0 и 1; тогда c можно представить в виде $c = \frac{m}{n}$, где m – некоторое целое число, отличное от 0, а n – некоторое целое положительное число, больше 1 и пусть $n = p_1^{k_1} p_2^{k_2} ... p_r^{k_r}$, где $p_1, p_2, ..., p_r$ – некоторые простые числа, а $r, k_1, k_2, ..., k_r$ – некоторые целые положительные числа.

Очевидно, что

$$g(\frac{1}{p_i}, \frac{1}{p_i}) = \frac{1}{p_i^2}, g(\frac{1}{p_i}, \frac{1}{p_i^2}) = \frac{1}{p_i^3}, ..., g(\frac{1}{p_i}, \frac{1}{p_1^{k_i - 1}}) = \frac{1}{p_1^{k_i}}, (i = 1, 2, ..., r);$$

$$g(\frac{1}{p_1^{k_1}}, \frac{1}{p_2^{k_2}}) = \frac{1}{p_1^{k_1}p_2^{k_2}}, g(\frac{1}{p_1^{k_1}p_2^{k_2}}, \frac{1}{p_3^{k_3}}) = \frac{1}{p_1^{k_1}p_2^{k_2}p_3^{k_3}}, ..., g(\frac{1}{p_1^{k_1} \cdot ... \cdot p_{r-1}^{k_{r-1}}}, \frac{1}{p_r^{k_r}}) =$$

$$= \frac{1}{p_1^{k_1} \cdot ... \cdot p_{r-1}^{k_{r-1}}p_r^{k_r}} = \frac{1}{n}.$$

Рассмотрим два случая.

1. $m > 0$. Тогда имеем

$$h_2(\frac{1}{n}, \frac{1}{n}) = \frac{2}{n}, h_2(\frac{2}{n}, \frac{1}{n}) = \frac{3}{n}, \cdot ... \cdot h_2(\frac{m-1}{n}, \frac{1}{n}) = \frac{m}{n}.$$

2. $m < 0$. Тогда $-m > 0$ b строим число $\frac{-m}{n}$ так, как это сделано в случае (1). Далее, имеем

$$h_1(\frac{-m}{n}) = -\frac{-m}{n} = \frac{m}{n}.$$

Итак, $[B] \supseteq Q$.

Теперь построим все мономы с рациональными коэффициентами, отличные от константы, т.е. рп-функции вида $cx_1^{k_1}x_2^k...x_n^{k_n}$ где c - произвольное рациональное число, отличное от нуля, n - произвольное положительное целое число, а $k_1, k_2, ..., k_n$ – произвольные натуральные числа, при этом $k_1 + k_2 + ... + k_n \neq 0$ (т.е. имеем хотя бы одну существенную переменную). Очевидно, что из функции $g(x, y)$ и константы 1 с помощью операций суперпозиции можно получить любой моном вида $x_1^{k_1}x_2^k...x_n^{k_n}$ (в том числе и тождественную функцию $f(x) = x$). Далее, подставляя в функцию $g(x, y)$ вместо x константу c, а вместо y – моном $x_1^{k_1}x_2^k...x_n^{k_n}$, мы получим рп-моном $cx_1^{k_1}x_2^k...x_n^{k_n}$.

Ясно, что из функции $f(x, y)$ с помощью операций суперпозиции можно получить любую линейную функцию вида $x_1+x_2+...+x_m (m \geq 2)$.

Далее, поскольку любой рп-полином является конечной суммой рп-мономов, то, если в подходящей линейной функции вида $x_1 + x_2 + ... + x_m$ вместо ее переменных подставим соответствующие рп-мономы, то получим желаемую рп-полином из P_Q.

Таким образом, из функций множества B с помощью операций суперпозиции можно получить все функции из P_Q, поэтому $[B] \supseteq P_Q$. Но, с другой стороны, поскольку $B \subset P_Q$, то в силу свойства оператора замыкания $[B] \subseteq P_Q$. Следовательно, $[B] = P_Q$, т.е. B является полной системой в $\mathbf{F}_Q$.

Далее, докажем, что полная система B является и базисом ф.с. $\mathbf{F}_Q$.

Если из B выбросить функцию $x-y$, то получим собственную подсистему $B_- = \{xy, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\}$, состоящую только из рп-мономов и, следовательно, из ее функций с помощью операций суперпозиции можно получить только рп-мономи, поэтому $[B] \neq P_Q$.

Если из B выбросить функцию xy, то получим собственную подсистему $B_* = \{x - y, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\}$, состоящую только из линейных рп-полиномов и, следовательно, из ее функций с помощью операций суперпозиции можно получить только линейные рп-полиномы, поэтому $[B] \neq P_Q$.

Легко заметить, что если из B выбросить константу $\frac{1}{p}$, где p – любое простое число, то из полученной собственной подсистемы с помощью операций суперпозиции невозможно получить все рп-полиномы, в частности константу $\frac{1}{p}$, поэтому $[B] \neq P_Q$.

Если из B выбросить более одной функции, то ясно, что полученная собственная подсистема системы B не является полной в $\mathbf{F}_Q$.

Итак, ни одна собственная подсистема полной системы B не является полной в $\mathbf{F}_Q$; это означает, что B является базисом ф.с. $\mathbf{F}_Q$. □

Ф.с. $\mathbf{F}_Q$ является конечно-порожденной.

4. Замкнутые и предполные классы

Теорема 3. *В ф.с. $\mathbf{F}_Q$ существуют только следующие конечные замкнутые классы:*

- *C, где C - произвольное конечное подмножество множества $F_Q^{(0)}$;*
- *$I_1 = \{x\}, I_2 = \{x; -x\}$;*
- *$C \cup I_1, \{\pm c_1, \cdots, \pm c_k\} \cup I_2$, где $\pm c_1, \cdots, \pm c_k \in Q$, а C, I_1 и I_2 определяются соответственно в предыдущих пунктах.*

Доказательство. Очевидно, что каждое из перечисленных множеств является конечным замкнутым классом в $\mathbf{F}_Q$. Покажем, что в $\mathbf{F}_Q$ не существует других конечных замкнутых классов, отличных от перечисленных. Для этого достаточно доказать, что если замкнутый класс M содержит рп-функцию $f(x_1, ..., x_n)$, отличную от константы, $g(x) = x$ и $h(x) = -x$, то он содержит бесконечное число попарно различных рп-функций.

Пусть M содержит рп-функцию $f(x_1, ..., x_n)$. Поскольку $f(x_1, ..., x_n)$ отлична от константы, то она имеет существенную переменную; не огра-

ничивая общность, можно считать, что существенными переменными функции $f(x_1, ..., x_n)$ являются переменные $x_1, ..., x_n$ $(n \geq 1)$.

Далее, возможны два случая:

1. $n = 1$, т.е. f имеет одну существенную переменную.

Так как функция f отлична от $g(x)$ и $h(x)$, то очевидно, что последовательность функций

$$f(x), f(f(x)), f(f(f(x))), ...$$

состоит из попарно различных функций и все они принадлежат множеству M (поскольку M - замкнутый класс).

2. $n \geq 2$, т.е. f имеет более одной существенной переменной.

Тогда очевидно, что если в функции $f(x_1, ..., x_n)$ на место переменной x_1 подставим функцию $f(y_1, ..., y_n)$ (где $\{y_1, ..., y_n\} \cap \{x_1, ..., x_n\} = \varnothing$), то получим функцию

$$f(f(y_1, ..., y_n), x_2..., x_n),$$

существенно зависящую от всех своих $2n - 1$ переменных; затем, если в функции $f(f(y_1, ..., y_n), x_2..., x_n)$ на место переменной y_1 подставим функцию $f(z_1, ..., z_n)$ (где $\{z_1, ..., z_n\} \cap \{y_1, ..., y_n, x_1, ..., x_n\} = \varnothing$), то получим функцию

$$f(f(f(z_1, ..., z_n), y_2, ..., y_n), x_2..., x_n),$$

существенно зависящую от всех своих $3n - 2$ переменных и т.д. Итак, имеем бесконечную последовательность функций

$$f(x_1, ..., x_n), f(f(y_1, ..., y_n), x_2..., x_n), f(f(f(z_1, ..., z_n), y_2, ..., y_n), x_2..., x_n), ...,$$

которая состоит из попарно различных функций (поскольку числа существенных переменных этих функций попарно различны) и все они принадлежат множеству M (т.к. M – замкнутый класс). Следовательно, M содержит бесконечное число попарно различных функций. □

Замечание 8. *Заметим, что в функциональной системе* $\mathbf{F}_Q$

- *число всех конечных замкнутых классов равно* c_0;
- *число всех бесконечных замкнутых классов равно* c;
- *число всех замкнутых классов равно* c.

Замечание 9. *Что касается базисов замкнутых классов, то имеют место все три случая:*

- *в ф.с.* $\mathbf{F}_Q$ *существует замкнутый класс, имеющий конечный базис;*
- *в ф.с.* $\mathbf{F}_Q$ *существует замкнутый класс, имеющий бесконечный базис;*
- *в ф.с.* $\mathbf{F}_Q$ *существует замкнутый класс, не имеющий базиса.*

Чтобы убедиться в этом, достаточно привести примеры соответствующих замкнутых классов.

Пусть $M = \{2x, 4x, 8x, ..., 2^n x, ...\}$, где n – любое положительное целое число. Ясно, что множество M является замкнутым классом в $\mathbf{F}_Q$, а система $B = \{2x\}$ – его базисом. Следовательно, M замкнутый класс, имеющий конечный базис.

Пусть $M = \{x, 2x, 3x, ..., mx, ...\}$, где m – любое положительное целое число. Ясно, что множество M является замкнутым классом в $\mathbf{F}_Q$, а система $B = \{x, 2x, 3x, 5x, ..., px, ...\}$, где p – любое простое число, является его базисом. Следовательно, M замкнутый класс, имеющий бесконечный базис.

Пусть $M = I_O(T)$, где $T = \{1, x_1^2, x_1^2 x_2^2, x_1^2 x_2^2 x_3^2, ...\}$. Покажем, что замкнутый класс M не имеет базиса.

Очевидно, что M состоит из всех рп-полиномов вида 1 и $x_1^{2k_1} \cdot ... \cdot x_n^{2k_n}$, где n – любое положительное целое число, а $k_1, ..., k_n$ – любые натуральные числа.

Допустим, что класс M имеет базис; обозначим его через B. Ясно, что $1 \in B$ и B содержит функцию, отличную от константы 1; обозначим ее через $f_1(x_1, ..., x_{n_1})$, где $n_1 > 0$. Пусть число переменных, которые содержит эта функция во 2-ой степени, равно $m_1 (0 \leq m_1 \leq n_1)$; не ограничивая общность, можно считать, что этими переменными являются переменные $x_1, ..., x_{m_1}$. Следовательно, $f_1(x_1, ..., x_{n_1})$ имеет вид

$$f_1(x_1, ..., x_{n_1}) = x_1^2 ... x_{m_1}^2 x_{m_1+1}^{2k_{m_1+1}} ... x_{n_1}^{2k_{n_1}},$$

где $k_{m_1+1}, ..., k_{n_1}$ – некоторые натуральные числа отличные, от 0 и 1.

Очевидно, что из функций 1 и $f_1(x_1, ..., x_{n_1})$ с помощью операций суперпозиции невозможно получить функцию

$$g(x_1, ..., x_{m_1+1}) \equiv x_1^2 ... x_{m_1}^2 x_{m_1+1}^2.$$

Следовательно, B содержит функцию вида

$$f_2(x_1, ..., x_{n_2}) = x_1^2 ... x_{m_1}^2 x_{m_1+1}^2 ... x_{m_2}^2 x_{m_2}^{2l_{m_2+1}} ... x_{n_2}^{2l_{n_2}},$$

где n_2, m_2 – некоторые положительные целые числа, при этом $n_2 > n_1$ и $n_1 < m_2 \leq n_2$, а $l_{m_2+1}, ..., l_{n_2}$ – некоторые натуральные числа, отличные от 0 и 1.

Очевидно, что из функций $1, f_1(x_1, ..., x_{n_1})$ и $f_2(x_1, ..., x_{n_2})$ с помощью операций суперпозиции невозможно получить функцию

$$g(x_1, ..., x_{m_2+1}) \equiv x_1^2 ... x_{m_2}^2 x_{m_2+1}^2.$$

Следовательно, B содержит функцию вида

$$f_3(x_1, ..., x_{n_3}) = x_1^2 ... x_{m_2}^2 x_{m_2+1}^2 ... x_{m_3}^2 x_{m_3+1}^{2s_{m_2+1}} ... x_{n_3}^{2s_{n_3}},$$

где n_3, m_3 – некоторые положительные целые числа, при этом $n_3 > n_2$ и $n_2 < m_3 \leq n_3$, а $s_{m_3+1}, ..., s_{n_3}$ – некоторые натуральные числа, отличные от 0 и 1 и т.д.

Таким образом, B содержит бесконечное число попарно различных функций

$$1, f_1(x_1, ..., x_{n_l}), f_2(x_1, ..., x_{n_2}), f_3(x_1, ..., x_{n_3}), ...$$

Но, с другой стороны, если рассмотрим любую бесконечную подпоследовательность функций этой последовательности, содержащую константу 1, то очевидно, что из функций этой подпоследовательности с помощью операции суперпозиции можно получить все функции множества T, следовательно, и все функции замкнутого класса M. Следовательно, собственная подсистема базиса B является полной в M. Получили противоречие. Итак, замкнутый класс M не имеет базиса.

Замечание 10. *В функциональной системе* $\mathbf{F}_Q$

- *число всех замкнутых классов, имеющих конечный базис, равно* c_0;
- *число всех замкнутых классов, имеющих бесконечный базис, равно* c;
- *число всех замкнутых классов, не имеющих базиса, равно* c.

Пункты (1)–(2) очевидны. Чтобы убедиться в справедливости последнего пункта, достаточно заметить, что множество

$$M_T = [\{1, x_1^2, x_1^2x_2^2, x_1^2x_2^2x_3^2, ...\} \cup T],$$

где T - произвольное бесконечное подмножество множества всех простых чисел, не имеет базиса и если $T_1 \neq T_2$, где T_1, T_2 – любые подмножества множества всех простых чисел, то $M_{T_1} \neq M_{T_2}$.

Рассмотрим некоторые конкретные замкнутые классы в $\mathbf{F}_Q$, которые являются предполными классами и тем самым играют важную роль при решении проблемы полноты.

Теорема 4. *Если W произвольное конечное подмножество множества Q, то класс $U(W)$ является предполным классом в функциональной системе $\mathbf{F}_Q$.*

Доказательство. Пусть $W = \{c_1, ..., c_m\}$, где $c_1, ..., c_m$ – константы из Q.

Ясно, что $U(W) \neq \varnothing, U(W) \neq P_Q$ и $I_Q(U(W)) = U(W)$.

Далее, пусть $f(x_1, ..., x_n) \in P_Q \setminus U(W)$; тогда для некоторых констант $c_{i_1}, ..., c_{i_m}$ из W и c из $Q \setminus W$ будет выполнено $f(c_{i_1}, ..., c_{i_m}) = c$.

Очевидно, что рп-полиномы

$$g_1(t, x, y) = \frac{(t - c_1) \cdot ... \cdot (t - c_m)}{(c - c_1) \cdot ... \cdot (c - c_m)} \cdot (x - y - c_1) + c_1,$$

$$g_2(t, x, y) = \frac{(t - c_1) \cdot ... \cdot (t - c_m)}{(c - c_1) \cdot ... \cdot (c - c_m)} \cdot (xy - c_1) + c_1,$$

$$g_3(t) = \frac{(t - c_1) \cdot ... \cdot (t - c_m)}{(c - c_1) \cdot ... \cdot (c - c_m)} \cdot (\frac{1}{p} - c_1) + c_1,$$

где p – произвольное простое число, принадлежать классу $U(W)$.

Следовательно, рп-полиномы

$$g_1(c, x, y) = x - y, g_2(c, x, y) = xy, g_3 = \frac{1}{p}$$

принадлежат классу $[U(W) \cup \{f(x_1, ..., x_n)\}]$; т.е. $[U(W) \cup \{f(x_1, ..., x_n)\}]$ содержит подсистему

$$B = \{x - y, xy, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\},$$

(p – любое простое число), которая является полной в $\mathbf{F}_Q$ (см. теорему (2)). Значит, $[U(W)\bigcup\{f(x_1,...,x_n)\}] = P_Q$). $\square$

Для любого положительного рационального числа c обозначим через $Qc+ \equiv \{q \in Q : q > c\}$ и $V_{c+} = U(Q_{c+})$.

Теорема 5. *Для любого рационального числа $c \geq 1$ множество V_{c+} является предполным классом в функциональной системе $\mathbf{F}_Q$.*

Д о к а з а т е л ь с т в о. Очевидно, что $V_{c+} \neq \varnothing, [V_{c+}] = V_{c+}$ и $[V_{c+}] \neq P_Q$.

Очевидно также, что $f(x,y) = x+y, g(x,y) = xy, h(x) = 2x - c \in V_{c+}$.

Пусть $f(x_1,...,x_n) \in P_Q \setminus V_{c+}$. Тогда для некоторых констант $c_1,...,c_n$ из Q_{c+} и k из $Q \setminus Q_{c+}$ будет выполнено $f(c_1,...,c_n) = k$.

Так как $k \notin Q_{c+}$, то $k < c$; но тогда число $k_1 \equiv h(k) = 2k - c$, которое принадлежит классу $[V_{c+}\bigcup\{f(x_1,...,x_n)\}]$, меньше числа k; число $k_2 \equiv h(k_1) = 2k_1 - c$, которое также принадлежит классу $[V_{c+}\bigcup\{f(x_1,...,x_n)\}]$, меньше числа k_1; число $k_3 \equiv h(k_2) = 2k_2 - c$, которое принадлежит классу $[V_{c+}\bigcup\{f(x_1,...,x_n)\}]$, меньше числа k_2 и т.д. Через конечное число шагов мы получим некоторое отрицательное число, которое принадлежит классу $[V_{c+}\bigcup\{f(x_1,...,x_n)\}]$. Обозначим его через $-l$, где $l-$ некоторое положительное рациональное число.

Далее, числа

$$f(-l,-l), = -2l, f(-l,-2l) = -3l, f(-l,-3l) = -4l, ...$$

принадлежат классу $[V_{c+}\bigcup\{f(x_1,...,x_n)\}]$. Из этих чисел выбираем одно такое число, пусть этим числом является $-ml$, что ml и $ml-1$ принадлежат множеству W_{c+} (здесь $m-$ некоторое целое число). Тогда число $f(ml, ml-1) = -1$ принадлежит классу $[V_{c+}\bigcup\{f(x_1,...,x_n)\}]$.

Имеем

$$f(-1,-1) = -2, f(-1,-2) = -3, f(-1,-3) = -4, \ldots$$

также принадлежат классу $[V_{c+}\bigcup\{f(x_1,...,x_n)\}]$.

Следовательно, $[V_{c+}\bigcup\{f(x_1,...,x_n)\}]$ содержит все отрицательные числа.

Так как для любого простого числа p число $[c+1]+\frac{1}{p}$ больше c, т.е. $[c+1]+\frac{1}{p}$ принадлежит классу V_{c+} (здесь $[c+1]$ - целая часть числа $c+1$), а отрицательное число $-[c+1]$ принадлежит классу $[V_{c+}\bigcup\{f(x_1,...,x_n)\}]$, то число $f([c+1]+\frac{1}{p}, -[c+1]) = \frac{1}{p}$ принадлежит классу $[V_{c+}\bigcup\{f(x_1,...,x_n)\}]$. Более того, $g(-1,x) = -y$, и $f(x,-y) = x - y$.

Итак, $[V_{c+} \bigcup \{f(x_1, ..., x_n)\}]$ содержит подсистему

$$B = \{x - y, xy, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\},$$

где $p-$ произвольное простое число, к которая является полной (теорема (2)).

Следовательно, множество V_{c+} является предполным классом в функциональной системе $\mathbf{F}_Q$. $\square$

Для любого отрицательного целого числа c обозначим через $Q_{c-} \equiv \{q \in Q : q \leq c\}$ и $V_{c-} \equiv U(Q_{c-})$.

Аналогично теореме (5) доказывается справедливость следующего утверждения.

Теорема 6. *Для любого рационального числа $c \leq -1$ множество V_{c-} является предполным классом в функциональной системе $\mathbf{F}_Q$.*

Для любого простого числа p обозначим через Q_p множество всех несократимых рациональных дробей, знаменатель которых делится на p и пусть

$$V_{\bar{p}} = \{f(x_1, ..., x_n) \in P_Q : f(c_1, ..., c_n) \in Q \backslash Q_p \text{ при } c_1, ..., c_n \in Q \backslash Q_p\}.$$

Теорема 7. *Для любого простого числа p множество $V_{\bar{p}}$ является предполным классом в функциональной системе $\mathbf{F}_Q$.*

Д о к а з а т е л ь с т в о. Очевидно, что $V_{\bar{p}} \neq \varnothing, [V_{\bar{p}}] = V_{\bar{p}}$ и $[V_{\bar{p}}] \neq P_Q$.

Очевидно также, что $f(x, y) = x - y, g(x, y) = xy \in V_{\bar{p}}$ и все числа вида $\frac{1}{q}$, где q любое простое число, отличное от p, принадлежать $V_{\bar{p}}$.

Пусть $f(x_1, ..., x_n) \notin V_p$. Это означает, что существуют такие константы $c_1, ..., c_n \in Q \backslash Q_p$ значение функции $f(c_1, ..., c_n) \in Q_p$, т.е. имеет следующий вид несократимой дроби $\frac{r}{kq}$, где $r-$ некоторое целое число, отличное от нуля, а $k-$ некоторое положительное целое число, при этом $|r|$ иk, а также $|r|$ и p взаимно простые числа. Но тогда $[V_{\bar{p}} \cup f(c_1, ..., c_n)]$ содержит число $\frac{r}{kp} + \frac{r}{kp} + \ldots + \frac{r}{kp}(k \text{ раз}) = \frac{r}{p}$.

Так как $|r|$ и p взаимно простые числа, то уравнение $px + ry = 1$ имеет решение в целых числах (это общеизвестно из курсов алгебры и теории чисел). Пусть $x = l$ и $y = s$ является решением этого уравнения, т.е. $pl + rs = 1$, где l и s некоторые целые числа, отличные от 0.

Далее, возможны два случая:

1. $s > 0$; тогда $[V_{\bar{p}} \cup f(c_1, ..., c_n]$ содержит число $l + \frac{r}{kp} + \frac{r}{kp} + \ldots + \frac{r}{kp}(s \text{ раз}) = \frac{pl+rs}{p} = \frac{1}{p}$;

2. $s < 0$; тогда $[V_{\bar{p}} \cup f(c_1, ..., c_n]$ содержит число $-(\frac{r}{kp} + \frac{r}{kp} + \ldots + \frac{r}{kp} - l)(-s \text{ раз}) = \frac{pl+rs}{p} = \frac{1}{p}$.

Следовательно, $[V_{\bar{p}} \cup f(c_1, ..., c_n]$ содержит подсистему

$$B = \{x - y, xy, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\},$$

которая является полной. Теорема доказана. □

И, наконец, рассмотрим еще одно семейство предполных классов.

Пусть S множество всех простых чисел, а $T-$ произвольное непустое подмножество множества S.

Введем обозначение

$$M_T \equiv (\bigcup_{s \in S} \frac{1}{s}) \cup (\bigcup_{p \in T} \{pz^8\}) \cup (\bigcup_{q \in S \setminus T} \{f_q(t, x, y, z)\}),$$

где $f_q(t, x, y, z) = [(t - qz^8)^4(xz - yz + z + 1)^4 + 1]^2(xz - yz + z + 1)$.

Рассмотрим класс $[M_T]-$ замыкание множества M_T.

Легко доказать, что класс $[M_T]$ не содержит полиномов вида $qz^8 (q \in S \setminus T)$.

С этой целью построим индуктивно последовательность множеств $H_1, H_2, \ldots, H_k, \ldots$ рп-функций.

Б а з и с и н д у к ц и и. Положим $H_1 = [M_T]$.

И н д у к т и в н ы й п е р е х о д. Пусть уже построены множества $H_1, H_2, \ldots, H_k$; тогда H_{k+1} определяется как множество всевозможных суперпозиций вида $g(h_1, \ldots, h_m)$, где $g-$ функция из M_T, а $h_1, \ldots, h_m-$ либо переменные, либо функции из $H_1, H_2, \ldots, H_k$.

Легко заметить, что $\bigcup_{i=1}^{\infty} H_i = [M_T]$.

Далее, нетрудно показать с помощью математической индукции по i, что ни одно множество H_i не содержит полиномов вида qz^8, где $(q \in S \setminus T)$. Отсюда следует, что

1) $[M_T]$ не является полной системой;

2) если T_1 и T_2- произвольные непустые подмножества множества S и $M_{T_1} \neq M_{T_2}$, то $[M_{T_1}] \neq [M_{T_2}]$.

Если к множеству $[M_T]$ добавим любую функцию вида qz^8, где $q \in S \setminus T$ и замкнем полученное множество, то получим функцию

$$f(x, y, z) = f_q(qz^8, x, y, z) = xz - yz + z + 1,$$

из которой с помощью операции суперпозиций можно получить все полиномиальные функции с целыми коэффициентами, и в частности, $x-y$ и xy (см. [2]).

Следовательно, из имеющихся функций с помощью операций суперпозиции можно получить систему рп-функций $B = \{x - y, xy, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\}$, которая является полной. Поэтому $[[M_T] \cup \{qz^8\}]$ является полной в $\mathbf{F}_Q$.

Отсюда следует, что:

1) множество M_T можно расширить до предполного класса (расширение конечное), т.е. M_T содержится в некотором предполном классе;

2) если T_1 и T_2- произвольные непустые подмножества множества S и $T_1 \neq T_2$, то $[M_{T_1}] \cup [M_{T_2}]$ является полной системой. Следовательно, M_{T_1} и M_{T_2} не могут содержаться в одном предполном классе.

Таким образом, мы доказали справедливость следующего утверждения.

Если T_1 и T_2- произвольные непустые подмножества множества S и $T_1 \neq T_2$, то $[M_{T_1}]$ и $[M_{T_2}]$ содержится в разных предполных классах.

5. Полнота систем функций

Рассмотрим алгебраический вариант проблемы полноты: *найти необходимое и достаточное условие полноты систем рп-функций (на языке предполных классов); а, именно: является ли множество всех предполных классов (приведенной) критериальной системой? Найти число предполных классов.*

Лемма 1. *В функциональной системе $\boldsymbol{F}_Q$ каждый замкнутый класс, отличный от P_Q, можно расширить до предполного класса.*

Д о к а з а т е л ь с т в о. Пусть M произвольный замкнутый класс, отличный от P_Q.

Рассмотрим множество $M^* = M \cup \{x+1, x-y, xy\}$ (здесь $x+1$ "добавили"к множеству M из соображения, что замыкание $[M^*]$ содержало все целые числа).

Возможны два случая.

1) $[M^*] = P_Q$. Тогда в силу утверждения (1) класс M можно расширить до предполного класса.

2) $[M^*] \neq P_Q$; тогда существует такое простое число p, что $\frac{1}{p} \notin [M^*]$ (в противном случае $[M^*]$ содержало бы полную подсистему $B = \{x - y, xy, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\}$, и, следовательно, $[M^*] = P_Q$, что противоречиво).

Допустим, что некоторое число $\frac{r}{kp}$ из Q_p содержится в $[M^*]$, где $k-$ некоторое положительное целое число, $r-$ некоторое целое число, при этом $|r|$ и k, а также $|r|$ и $p-$ взаимно простые числа. Как известно из курса алгебры и теории чисел уравнение $px + ry = 1$ имеет решение в целых числах. Пусть $x = l$ и $y = s$ решение этого уравнения, т.е. $pl + rs = 1$, где $l, s-$ некоторые целые числа.

Ясно, что если $\frac{r}{kp} \in [M^*]$, то $\frac{r}{kp} + \ldots + \frac{r}{kp}(k \text{ раз}) = \frac{r}{p} \in [M^*]$;

Следовательно,

$$l + \frac{r}{kp} + \ldots + \frac{r}{kp}(s \text{ раз}) = \frac{pl + rs}{p} = \frac{1}{p} \in [M^*] \text{ при } s > 0$$

и

$$-(\frac{r}{kp} + \ldots + \frac{r}{kp}(-s \text{ раз}) \ - l) = \frac{pl + rs}{p} = \frac{1}{p} \in [M^*] \text{ при } s < 0.$$

Получили противоречие. Значить, $[M^*]$ не содержит ни одного элемента множества Q_p. Более того, $[M^*]$ не содержит ни одной такой рп-функции $f(x_1, \ldots, x_n)$ из P_Q, которая на множестве $Q \backslash Q_p$ принимает значение из Q_p. Поэтому рп-функции из $[M^*]$ сохраняют множество $Q \backslash Q_p$. Следовательно, $[M^*]$ содержится в предполном классе V_p; тем более, M содержится в V_p. Лемма доказана. □

Итак, *каждый замкнутый класс содержится в некотором предполном классе;* отсюда в силу утверждения (1) получаем справедливость следующей теоремы.

Теорема 8. *В функциональной системе* $\mathbf{F}_Q$ *множество всех предполных классов образуют критериальную систему.*

Теорема 9. *В функциональной системе* $\mathbf{F}_Q$ *мощность множества всех предполных классов равна* c.

Доказательство. Рассмотрим множество замкнутых классов $A = \{M_T\}_{T \subset S}$. С одной стороны, в силу утверждения (4) любые два элемента этого множества содержатся в разных предполных классах. Следовательно, число предполных классов не меньше c. Но, с другой стороны, так как F_Q счетная функциональная система, то число всех предполных классов не может быть больше c. Следовательно, искомое число равно континууму. □

6. Базисы полных систем

В функциональной системе $\mathbf{F}_Q$ множество $\{x-y, xy, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\}$, где $p-$ произвольное простое число, является базисом. Следовательно, $\mathbf{F}_Q$ счетно-порожденная функциональная система и поэтому любая полная система имеет счетный базис (если, конечно, он существует). Но всякая ли полная система в $\mathbf{F}_Q$ имеет базис? ответ на этот вопрос дает следующее утверждение.

Теорема 10. *В функциональной системе $\mathbf{F}_Q$ существует полная система, не имеющая базиса.*

Д о к а з а т е л ь с т в о. Обозначим через $q_1 = 0, q_i = \frac{1}{p_{i-1}} (i = 2, 3, \ldots)$ где p_{i-1} есть $(i-1)-$ое простое число в последовательности всех простых чисел $2, 3, 5, 7, 11, \ldots$. Пусть $M = \{x-y, xy, f_1(x), f_2(x), \ldots, f_n(x), \ldots\}$, где $f_n(x)$ $(n = 1, 2, 3, \ldots)-$ рп-функция $n-$ ой степени (т.е. функция, задаваемая полиномом одной переменной x, степень которого равна n, а коэффициенты - рациональные числа), принимающая в точках $q_1, q_2, \ldots, q_{n+1}$ соответственно значения $q_2, q_3, \ldots, q_{n+2}$. Существует такая единственная рп-функция (полином Лагранжа).

Нетрудно заметить, что класс $[M]$ содержит подсистему

$$\{x-y, xy, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\},$$

которая является полной в $\mathbf{F}_Q$. Следовательно, $[M] = P_Q$.

Итак, M полная система.

Допустим, что M имеет базис; обозначим его через B.

Так как $\mathbf{F}_Q$ бесконечно-порожденная ф.с., то B бесконечное множество.

Возможны следующие случаи.

1. $B = \{x-y, xy, f_{i_1}(x), f_{i_2}(x), \ldots, f_{i_n}(x), \ldots\}$, где

$$\{i_1, i_2, \ldots, i_n, \ldots\} \subseteq \{1, 2, \ldots, n, \ldots\}.$$

Но тогда легко заметить, что любая подсистема вида

$$\{x-y, xy, f_{j_1}(x), f_{j_2}(x), \ldots, f_{j_n}(x), \ldots\},$$

где

$$\{j_1, j_2, \ldots, j_n, \ldots\} \subseteq \{i_1, i_2, \ldots, i_n, \ldots\}$$

содержит подсистему

$$\{x-y, xy, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\},$$

которая является полной. Следовательно, $[B] = P_Q$.

2. $B = \{x-y, f_{i_1}(x), f_{i_2}(x), \ldots, f_{i_n}(x), \ldots\}$, где

$$\{i_1, i_2, \ldots, i_n, \ldots\} \subseteq \{1, 2, \ldots, n, \ldots\}.$$

Но тогда из конечного числа функций этого базиса с помощью операции суперпозиции можно получить функцию xy; пусть этими функциями являются $g_1, \ldots, g_k$. Тогда ясно, что любая подсистема вида

$$\{x-y, g_1, \ldots, g_k, f_{j_1}(x), f_{j_2}(x), \ldots, f_{j_n}(x), \ldots\},$$

где

$$\{j_1, j_2, \ldots, j_n, \ldots\} \subseteq \{i_1, i_2, \ldots, i_n, \ldots\}$$

содержит подсистему

$$\{x-y, xy, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\},$$

которая является полной. Следовательно, $[B] = P_Q$.

3. $B = \{xy, f_{i_1}(x), f_{i_2}(x), \ldots, f_{i_n}(x), \ldots\}$, где

$$\{i_1, i_2, \ldots, i_n, \ldots\} \subseteq \{1, 2, \ldots, n, \ldots\}.$$

Но тогда из конечного числа функций этого базиса с помощью операции суперпозиции можно получить функцию $x-y$; пусть этими функциями являются $h_1, \ldots, h_m$. Тогда ясно, что любая подсистема вида

$$\{xy, h_1, \ldots, h_m, f_{j_1}(x), f_{j_2}(x), \ldots, f_{j_n}(x), \ldots\},$$

где

$$\{j_1, j_2, \ldots, j_n, \ldots\} \subseteq \{i_1, i_2, \ldots, i_n, \ldots\}$$

содержит подсистему

$$\{x-y, xy, \frac{1}{2}, \frac{1}{3}, ..., \frac{1}{p}, ...\},$$

которая является полной. Следовательно, $[B] = P_Q$.

4. $B = \{f_{i_1}(x), f_{i_2}(x), \ldots, f_{i_n}(x), \ldots\}$, где

$$\{i_1, i_2, \ldots, i_n, \ldots\} \subseteq \{1, 2, \ldots, n, \ldots\}.$$

Но тогда B состоит только из рп-функций одной переменной, поэтому $[B] \neq P_Q$.

Во всех случаях приходим к противоречию базиса.

Значит, исходная полная система не имеет базиса. □

Автор выражает глубокую благодарность профессору Московского государственного университета им. М.В. Ломоносова, заведующему кафедрой Математической теории интеллектуальных систем механико-математического факультета МГУ В.Б. Кудрявцеву за постановку задачи и постоянную поддержку при выполнении данной работы.

Список литературы

[1] Алексиадис Н.Ф. *Функциональная система полиномов с натуральными коэффициентами*/Вестник МЭИ. 2013. №6. С. 109-111.

[2] Алексиадис Н.Ф. *Алгоритмическая неразрешимость проблемы полноты для полиномов с целыми коэффициентами*/Вестник МЭИ. 2015. №3. С. 110-117.

[3] Гаврилов Г.П. *О функциональной полноте в счетнозначной логике* — в кн.: Проблемы кибернетики, вып 15, М.: "Наука 1965, стр. 5-64.

[4] Кудрявцев В.Б. *Функциональные системы* — М.: Изд–во МГУ, 1982.

[5] Яблонский С.В. *Введение в дискретную математику* — М.: Наука, 1986.

On a functional system of polynomials with rational coefficients

Aleksiadis N.Ph.

This article investigates the completeness problem for polynomial functions with rational coefficients, as well as problems of a functional nature generated by its solution, namely: studying the structure of closed and precomplete classes, the problem of bases of complete systems. In particular, it was proved that

1) the system of functions is complete if and only if it is not wholly contained in any precomplete class;

2) the cardinality of the set of all precomplete classes is equal to the continuum;

3) there is a complete system that does not have a basis.

Keywords: polynomial, functional system, problem completeness, rational function, closed classes.

Запросы на сравнение в задаче параметро-эффективной расшифровки булевых функций

Быстрыгова А.В.

В работе рассматривается задача точной параметро-эффективной расшифровки функций из замкнутых классов Поста при помощи запросов на сравнение. Показано, что сложность расшифровки с помощью этих запросов не хуже, а для некоторых классов по порядку лучше, чем таковая для случая запросов на значение.

Ключевые слова: точная расшифровка, параметро-эффективная расшифровка, запросы на значение, запросы на сравнение, замкнутые классы Поста.

1. Введение

Начиная с двадцатого века, одним из популярных направлений исследований в дискретной математике остается изучение задач по расшифровке функций. Эту задачу можно представить как игру между двумя игроками: “учителем“ и “учеником“, в которой учитель выбирает функцию, а ученик должен как можно скорее узнать загаданную учителем функцию, задавая определенные вопросы. При этом ученику лишь известны некоторые параметры, определяющие функцию: число переменных, от которых зависит функция, количество существенных переменных, какое-то свойство, которым она обладает (например, монотонность, количество единиц в векторе значений функции), и так далее.

Многообразие задач в сфере расшифровки функций достигается за счет варьирования модели, в рамках которой проводится расшифровка, классов функций, которые являются целью расшифровки, типов запросов, с помощью которых осуществляется расшифровка. К примеру, в данной работе рассматривается две версии задачи расшифровки, которые отличаются типом запросов к учителю. В первой версии ученик

просит учителя раскрыть значение выбранной функции на каком-то одном наборе значений переменных (запрос на значение), а во второй версии задачи ученик выбирает два набора значений переменных и просит учителя вернуть знак разности значений загаданной функции на этих наборах (запрос на сравнение). Что касается модели расшифровки, то в обеих используется точная модель, то есть, ученик должен полностью восстановить вектор значений загаданной функции.

В работе [1] проведен обзор результатов точной параметроэффективной расшифровки при помощи запросов на значение для всех замкнутых классов решетки Поста. Насколько целесообразно в этой задаче запросы на значение заменять на запросы на сравнение пока неясно, поскольку в литературе они стали освещаться относительно недавно, когда были впервые введены в [2]. Похожий тип запросов рассматривался в задаче восстановления частичного порядка на множестве [3]. Примеры работ, где изучается сложность расшифровки функций с помощью запросов на сравнение, — [2, 4, 5, 6, 7], но все эти исследования не касались замкнутых классов решетки Поста.

Данная работа посвящена прояснению ситуации, насколько "сильнее" запросы на сравнение относительно запросов на значение в упомянутой выше задаче.

2. Основные понятия и формулировка результатов

Пусть F — некоторый класс функций. Под $F(n,k)$ будем понимать множество функций из F, зависящих от переменных $x_1, x_2, \ldots, x_n$, при этом не более k из которых существенные.

Запросом на сравнение к функции f будем называть пару наборов (a,b), а ответом на него – число $f(a) - f(b)$.

Под *запросом на значение* к функции f будем понимать набор a, а ответом на него будем считать число $f(a)$.

Индекс $_C$ в обозначениях будем приписывать в случае запросов на сравнение и индекс $_V$ — в случае запросов на значение.

Под *алгоритмом расшифровки запросами на сравнение* будем понимать условный эксперимент, который последовательно генерирует запросы на сравнение к функции в зависимости от ответов на предыдущие запросы. Говорим, что *алгоритм расшифровывает функцию* $f \in F(n,k)$,

если ответы на запросы, заданные алгоритмом, однозначно определяют вектор значений функции $f \in F(n,k)$.

Будем говорить, что *класс $F(n,k)$ можно расшифровать запросами на сравнение*, если существует алгоритм расшифровки запросами на сравнение, который для любой функции $f \in F(n,k)$ расшифровывает f. Если про класс $F(n,k)$ нельзя сказать, что его можно расшифровать запросами на сравнение, тогда будем, говорить, что *класс $F(n,k)$ нельзя расшифровать запросами на сравнение.*

Будем говорить, что *класс F можно расшифровать запросами на сравнение*, если для любых целых $n, k(k \leq n)$ можно расшифровать запросами на сравнение класс $F(n,k)$. Если про класс F нельзя сказать, что его можно расшифровать запросами на сравнение, тогда будем, говорить, что *класс F нельзя расшифровать запросами на сравнение.*

Под $\varphi_C(A,f)$ будем понимать число запросов на сравнение, требуемое алгоритму (ученику) A, для расшифровки $f \in F(n,k)$.

$T_C(F,n,k)$ — множество всех алгоритмов расшифровки запросами на сравнение функций из $F(n,k)$.

Определим через $\varphi_C(F,n,k) = min_{A \in T_C(F,n,k)} max_{f \in F(n,k)} \varphi_C(A,f)$ минимальное число запросов на сравнение, которое задаст наилучший алгоритм на самой худшей для себя функции из $F(n,k)$.

Аналогично, определяются *алгоритм расшифровки запросами на значение* и $\varphi_V(A,f), T_V(F,n,k), \varphi_V(F,n,k)$.

Далее будем говорить только про *параметро-эффективную расшифровку*, поэтому считаем, что k мало по сравнению с n, то есть $k = o(n)$.

Под $]a[$ будем понимать наименьшее целое, больше или равное a.

Под $|A|$ будем понимать мощность множества A.

Под *набором, сформированным по множеству A*, будем понимать набор, у которого переменные из множества A установлены в 1, остальные – в 0. Под *запросом на сравнение на паре множеств A, B* будем понимать пару наборов, сформированных по множествам A, B.

Будем писать $f \gtrsim g$, если существует такая положительная константа c, что, начиная с некоторого номера n_0, $g(n) \leq c \cdot f(n)$.

Далее напомним обозначения замкнутых классов Поста, при этом классы из "правой" половины решетки Поста заведомо опустим, в силу того, что они являются двойственными к классам из "левой" половины, следовательно задача расшифровки классов из "правой" половины сводится к задаче расшифровки классов из "левой"половины.

C_1 — все функции двузначной логики.

C_2 — все функции, сохраняющие 1.

C_3 — все функции, сохраняющие 0.

C_4 — пересечение C_2, C_3.

A_1 — все монотонные функции.

A_2 — все монотонные функции, сохраняющие 1.

A_3 — все монотонные функции, сохраняющие 0.

A_4 — пересечение A_2, A_3.

F_4^i — класс функций, таких что для любого j $(2 \leq j \leq i)$ верно, что любые j наборов, на которых f обращается в 0, имеют общую нулевую компоненту.

F_3^i — пересечение класса F_4^i и A_2.

F_1^i — пересечение класса F_4^i и C_4.

F_2^i — пересечение класса F_4^i и A_4.

D_3 — все самодвойственные функции.

D_1 — все самодвойственные функции, сохраняющие 0.

D_2 — пересечение классов D_3, F_2^2.

S_1 — все логические суммы вида $x_{i_1} \vee x_{i_2} \vee \ldots \vee x_{i_k}$.

S_3 — S_1 и константа 1.

S_5 — S_1 и константа 0.

S_6 — объединение классов S_3, S_5.

L_1 — все линейные функции.

L_5 — пересечение L_1, D_3.

L_2 — пересечение L_1, C_2.

L_3 — пересечение L_1, C_3.

L_4 — пересечение L_2, L_3, L_5.

$O_9 = \{1, 0, x, \bar{x}\}$.

$O_6 = \{0, 1, x\}$.

$O_4 = \{x, \bar{x}\}$.

$O_5 = \{1, x\}$.

$O_8 = \{0, x\}$.

$O_1 = \{x\}$.

Теорема 1. *Класс $F(n, k)$ расшифровать запросами на сравнение нельзя тогда и только тогда, когда обе константные функции $0, 1 \in F(n, k)$.*

Непосредственно из этой теоремы получаем следующее следствие.

Следствие 1. *Если $F \in \{C_1, A_1, L_1, S_6, O_9, O_6\}$, то расшифровать запросами на сравнение класс F невозможно. Если $F \in \{C_2, C_3, C_4, A_2, A_3, A_4, F_4^i, F_3^i, F_1^i, F_2^i, D_1, D_2, S_1, S_3, S_5, L_2, L_3, L_4\}$ или $F \in \{O_8, O_5, O_1, O_4, D_3, L_5\}$, то класс F можно расшифровать.*

Теорема 2. *Сложность расшифровки запросами на сравнение функций из классов* $C_2, C_3, C_4, A_2, A_3, A_4, F_4^i, F_3^i, F_1^i, F_2^i, D_1, D_2$ *и классов* $S_1, S_3, S_5, L_2, L_3, L_4, O_8, O_5, D_3, L_5$ *не хуже, чем сложность расшифровки запросами на значение.*

Теорема 3. *Справедливы следующие соотношения:*

1) $\varphi_C(O_1, n, 1) =]\log_3 n[\leq]\log_2 n[= \varphi_V(O_1, n, 1)$,

2) $\varphi_C(O_4, n, 1) \leq]\log_3 n[+ 1 \leq]\log_2 n[+ 1 = \varphi_V(O_4, n, 1)$.

Теорема 4. *Если при расшифровке запросами на сравнение разрешить ровно один запрос на значение, тогда классы* $C_1, A_1, L_1, S_6, O_9, O_6$ *можно расшифровать со сложностью не более, чем на один запрос больше, чем при расшифровке, в которой разрешены только запросы на значение.*

3. Доказательство теорем

Далее приведем доказательство теоремы 1.

Доказательство. Докажем достаточность. Обе константные функции на любом запросе на сравнение возвращают значение 0, поэтому распознать, какая из них загадана невозможно.

Докажем необходимость. От противного. Пусть $\{0, 1\} \not\subseteq F(n, k)$.

Поскольку класс $F(n, k)$ расшифровать нельзя, то какой бы алгоритм расшифровки $A \in T_C(F, n, k)$ не взять, то найдется функция $f \in F(n, k)$, которую этот алгоритм не сможет расшифровать, то есть нельзя отличить ее от какой-то другой функции из класса $F(n, k)$. Рассмотрим следующий алгоритм Q. В качестве запросов на сравнение возьмем множество всех пар $(a, b), a, b \in \{0, 1\}^n$. Зафиксируем любую функцию $f \in F(n, k)$.

1) Если $f \in \{0, 1\}$, то ответы на все запросы будут равны 0, что отличает ее от остальных неконстантных функций, для которых среди ответов на запросы обязательно встретится хотя бы одна 1.

2) Если $f \notin \{0, 1\}$, то среди ответов на запросы встречается хотя бы одна 1, что отличает эту функцию от константной. Рассмотрим любую отличную от f функцию $g \in F(n, k), g \notin \{0, 1\}$. Функции f, g отличаются, значит, существует набор a такой, что $f(a) \neq g(a)$. Возможны два случая:

- существует набор b такой, что $f(b) = g(b)$, тогда на запросе (a, b) функции f, g вернут разные ответы;
- для любого набора b верно неравенство $f(b) \neq g(b)$, тогда на запросе (c, d), где $f(c) = 0, f(d) = 1$, функция f вернет ответ -1, а функция g вернет ответ 1.

Следовательно, алгоритм Q расшифровывает класс $F(n, k)$, получили противоречие.

□

Утверждение 1. *Если для некоторого набора a, некоторого числа $b \in \{0, 1\}$ для любой $f \in F(n, k)$ известно, что $f(a) = b$, и $\{0, 1\} \not\subseteq F(n, k)$, тогда $\varphi_C(F, n, k) \leq \varphi_V(F, n, k)$.*

Доказательство. В силу теоремы 1, класс $F(n, k)$ можно расшифровать запросами на сравнение, поэтому дальнейшие рассуждения имеют смысл.

Рассмотрим алгоритм расшифровки A запросами на значение, на котором достигается минимум выражения $max_{f \in F(n,k)} \varphi_V(A, f)$. По алгоритму A построим алгоритм расшифровки B запросами на сравнение следующим образом: первая компонента каждого запроса на сравнение есть набор из алгоритма A, а вторая компонента запроса на сравнение – это набор a. Тогда для любой $f \in F(n, k)$ верно соотношение $\varphi_C(B, f) = \varphi_V(A, f)$. Отсюда следует равенство $max_{f \in F(n,k)} \varphi_C(A, f) = max_{f \in F(n,k)} \varphi_V(A, f)$. Учитывая, что в $T_C(F, n, k)$ помимо алгоритмов расшифровки, предложенных выше, возможно имеются еще какие-то, получаем неравенство из утверждения. □

Утверждение 2. *Если $F(n, k) \subseteq D_3(n, k)$, тогда $\varphi_C(F, n, k) \leq \varphi_V(F, n, k)$.*

Доказательство. Рассмотрим алгоритм расшифровки A запросами на значение, на котором достигается минимум выражения $max_{f \in F(n,k)} \varphi_V(A, f)$. По алгоритму A построим алгоритм расшифровки B запросами на сравнение следующим образом: первая компонента каждого запроса на сравнение есть набор из алгоритма A, а вторая компонента запроса на сравнение – это противоположный ему набор. Ответ на любой такой запрос на сравнение однозначно восстановит значение функции на обоих наборах в силу того, что наборы противоположные, а функция самодвойственная. Поэтому для любой

$f \in F(n,k)$ верно соотношение $\varphi_C(B,f) = \varphi_V(A,f)$. Отсюда следует $max_{f\in F(n,k)}\varphi_C(A,f) = max_{f\in F(n,k)}\varphi_V(A,f)$. Учитывая, что в $T_C(F,n,k)$ помимо алгоритмов расшифровки, предложенных выше, возможно имеются еще какие-то, получаем неравенство из утверждения. □

Утверждение 3. *(Нижняя мощностная оценка) Если* $\{0,1\} \not\subseteq F(n,k)$, *то* $\varphi_C(F,n,k) \geq]\log_3 |F(n,k)|[$.

Доказательство. В силу теоремы 1, класс $F(n,k)$ можно расшифровать запросами на сравнение, поэтому дальнейшие рассуждения имеют смысл.

Доказательство аналогично случаю расшифровки запросами на значение.

Пусть задано x запросов на сравнение, на каждый запрос возможны 3 варианта ответа, следовательно, всего возможных векторов ответов не более 3^x ("не более" в силу того, что какие-то комбинации ответов на запросы могут быть между собой не согласованы, то есть не определяют функцию из заданного класса).

Необходимо задать такое количество запросов, чтобы однозначно восстановить функцию, поэтому должно выполняться следующее неравенство $3^x \geq |F(n,k)|$. Отсюда следует неравенство утверждения. □

Утверждение 4. *Если известно, что среди множества переменных* $A = \{x_{i_1}, x_{i_2}, \ldots, x_{i_q}\}, 1 \leq i_1 < i_2 < \ldots < i_q \leq n$, *нечетное число существенных переменных функции* $f(x_1, x_2, \ldots, x_n) \in L_3$. *Тогда одну существенную переменную можно найти, задав не более* $]\log_3 q[$ *запросов на сравнение.*

Доказательство. Шаг 1. Если $|A| = 1$, тогда СТОП, переменная из множества и есть существенная. Иначе, разделить множество A на непересекающиеся множества A_1, A_2, A_3, отличающиеся по мощности не более, чем на один элемент. Перейти к шагу 2.

Шаг 2. Запросить значение на сравнение на множествах A_1, A_2. Если значение равно 1, тогда положить $A = A_1$, перейти к шагу 1. Если значение равно -1, тогда положить $A = A_2$, перейти к шагу 1. Иначе, положить $A = A_3$, перейти к шагу 1.

Если на шаге 2 ответ на запрос равен 0, то значит, либо в обоих A_1, A_2 нечетное число существенных из множества A и значение функции на наборах, сформированных по каждому из множеств A_1, A_2 равно 1, либо в обоих A_1, A_2 четное число существенных из множества A и значение

функции на наборах, сформированных по каждому из множеств A_1, A_2 равно 0. В обоих случах, суммарно в A_1, A_2 содержится четное число существенных переменных, значит нечетное число существенных содержится в A_3.

После каждого шага 2 мощность множества A уменьшается как минимум в 3 раза.

□

Далее приведем доказательство теоремы 2.

Доказательство. Если $F \in \{C_2, C_3, C_4, A_2, A_3, A_4, F_4^i, F_3^i, F_1^i, F_2^i\}$ или $F \in \{D_1, D_2, S_1, S_3, S_5, L_2, L_3, L_4, O_8, O_5\}$, то согласно утверждению 1 получаем неравенство $\varphi_C(F, n, k) \leq \varphi_V(F, n, k)$.

Если $F \in \{D_3, L_5\}$, то в силу утверждения 2 получаем неравенство $\varphi_C(F, n, k) \leq \varphi_V(F, n, k)$.

□

Приведем доказательство теоремы 3.

Доказательство. Соотношение $\varphi_C(O_1, n, 1) =]\log_3 n[$ следует из утверждений 3 и 4. Учитывая соотношение $\varphi_V(O_1, n, 1) =]\log_2 n[$ из [8], получаем первый пункт теоремы.

Для получения верхней оценки для $\varphi_C(O_4, n, 1)$ предлагается следующий алгоритм расшифровки: узнаем значение на запросе $(0 \ldots 0, 1 \ldots 1)$, если ответ равен -1, то загадана функция $x_i, 1 \leq i \leq n$, иначе загадана функция $\bar{x_i}, 1 \leq i \leq n$. В случае функции x_i применяем алгоритм расшифровки, описанный в доказательстве утверждения 4, в случае функции $\bar{x_i}$ применяем похожий алгоритм. В результате, получаем неравенство $\varphi_C(O_4, n, 1) \leq 1 +]log_3 n[$. Учитывая соотношение $\varphi_V(O_4, n, 1) =]\log_2 n[+ 1$ с [1], получаем второй пункт теоремы.

□

Наконец, приведем доказательство теоремы 4.

Доказательство. В самом начале расшифровки сделаем один запрос на значение — запросим значение функции на любом наборе q. Далее в силу того, что известно значение загаданной функции на одном наборе, воспользуемся аналогом доказательства утверждения 1: зафиксируем "наилучший алгоритм расшифровки" A соответствующего класса запросами на значение, и вместо каждого его запроса t будем подавать запрос на

сравнение (q, t), тем самым мы узнаем значение на всех наборах алгоритма A, ведь точно известно значение функции на наборе q. Поскольку алгоритм A расшифровывал функцию, то и полученный алгоритм расшифровки запросами на сравнение и одним запросом на значение восстанавливает ее. □

4. Благодарность

Автор выражает благодарность научному руководителю — д.ф-м.н., профессору Э. Э. Гасанову за постановку задачи и помощь в работе.

Список литературы

[1] А. В. Быстрыгова, "Параметро-эффективная расшифровка булевых функций из замкнутых классов Поста", *Дискрет. матем.*, **31**:2 (2019), 34–58.

[2] Гасанов Э.Э., "Расшифровка линейных функций ранжирования", *Материалы XI Международного семинара "Дискретная математика и ее приложения посвященного 80-летию со дня рождения академика О.Б.Лупанова (Москва, 18-23 июня 2012 г.)*, Изд-во мех-мат фак-та МГУ, 2012, 332–334.

[3] А. А. Абдель Маджид, "О сложности восстановления частичного порядка", *Интеллектуальные системы*, **20**:4 (2016), 5–10.

[4] Гасанов Э. Э., Ниязова З. А., "Расшифровка арифметических сумм малого числа монотонных конъюнкций", *Материалы XI Международного семинара Дискретная математика и ее приложения (Москва, 18-23 июня 2012 г.)*, Изд-во механико-математического ф-та МГУ, Москва, 2012, 335–337.

[5] Ниязова З. А., "Расшифровка арифметических сумм монотонных конъюнкций", *Интеллектуальные системы. Теория и приложения*, **19**:4 (2015), 169–195.

[6] Хегай С.И., "Расшифровка полиномиальных функций ранжирования", *Интеллектуальные системы*, **19**:1 (2015), 213–230.

[7] Быстрыгова А.В., "Письмо в редакцию по поводу статьи З.А.Ниязовой "Расшифровка арифметических сумм монотонных конъюнкций"", *Интеллектуальные системы*, **22**:4 (2018), 107–109.

[8] R. Uehara, K. Tsuchida, I. Wegener, "Optimal Attribute-Efficient Learning Of Disjunction, Parity, And Threshold Functions", *EuroCOLT '97 Proceedings of the Third European Conference on Computational Learning Theory*, 1997.

Using comparation queries in attribute-efficient learning of Boolean functions
Bistrigova A.V.

We consider the problem of exact attribute-efficient learning functions of Post's closed classes with the help of comparation queries. Here, we show that the complexity of learning by comparation queries is not worse than by membership queries. Particularly, for some classes, if we use comparation queries, we get better value of complexity function.

Keywords: exact learning, attribute-efficient learning, membership queries, comparation queries, Post's closed classes.

A-полнота систем с добавками в классе линейных автоматов над кольцом двоично-рациональных чисел

Ронжин Д.В.

Представлено краткое изложение результатов, полученных при исследовании проблемы A-полноты систем линейных автоматов, функционирующих над кольцом двоично-рациональных чисел. Описаны условия A-полноты систем, содержащих все одноместные линейные автоматы и конечную добавку, а так же конечных систем, содержащих сумматор.

Ключевые слова: конечные автоматы,линейные автоматы, двоично-рациональные числа, А-полнота, предполный класс.

1. Введение

При исследовании задачи полноты систем конечных автоматов[1], нередко рассматривается вопрос об A-полноте[2] этих систем, а также задача полноты систем, содержащих добавки[3]. Интересным для изучения подклассом конечных автоматов являются линейные автоматы[4, 5], для которого описаны условия полноты конечных систем в терминах предполных классов.

Настоящая работа посвящена исследованию задачи A-полноты линейных автоматов, алфавиты состояний, входные и выходные алфавиты которых являются декартовыми произведениями множества двоично-рациональных чисел. Рассматриваются системы содержащие все одноместные линейные автоматы и конечную добавку, а также конечные системы, содержащие сумматор. Функции переходов и выходов автоматов в настоящей работе считаются линейными над кольцом двоично-рациональных чисел.

В отличие от случая автоматов над полем рациональных чисел[6], исследуемое в настоящей работе множество автоматов имеет конечный базис

по операциям композиции [1], что приводит к задаче A-полноты конечных систем. В настоящей работе сформулированы полученные условия A-полноты в терминах предполных классов.

2. Постановка задачи

Рассмотрим кольцо двоично-рациональных чисел, которое является подкольцом в поле рациональных чисел:

$$\mathbb{Q}_{\frac{m}{2^n}} = \left\{\frac{m}{2^n} | m \in \mathbb{Z}, n \in \mathbb{N}\right\},$$

Для обозначения того, что некоторая переменная i пробегает подмножество натуральных чисел $\{1, 2, ..., k\}$ будем использовать запись $i \in [1, k]$. $\forall l, k \in \mathbb{N}$ будем рассматривать конечные автоматы[1] с входным алфавитом $\mathbb{Q}^l_{\frac{m}{2^n}}$, выходным алфавитом $\mathbb{Q}_{\frac{m}{2^n}}$ и алфавитом состояний $\mathbb{Q}^k_{\frac{m}{2^n}}$, функции переходов и выходов являются линейными[4, 5]. Данное множество будем называть множеством линейных автоматом над кольцом двоично-рациональных чисел, и обозначим $L(\mathbb{Q}_{\frac{m}{2^n}})$[6].
Заметим, что множество $L(\mathbb{Q}_{\frac{m}{2^n}})$ имеет конечный базис по операциям композиции[1], а именно:

$$\mathbf{B} = \{V_{\bigoplus}(x, y), V_{\cdot(-\frac{1}{2})}(x), \xi_1(x)\}, \text{ где}$$

1) $V_{\bigoplus}(x, y)$ - сумматор с двумя входами в каждый такт,

2) $V_{\cdot(-\frac{1}{2})}(x)$ - умножитель на чило $-\frac{1}{2}$ в каждый такт,

3) $\xi_1(x)$ - задержка с единичным начальным состоянием.

Определим множество формальных степенных рядов над $\mathbb{Q}_{\frac{m}{2^n}}$:

$$\mathbb{Q}^{\infty}_{\frac{m}{2^n}}(\xi) = \left\{\alpha = \sum_{i=0}^{\infty} a_i \cdot \xi^i \mid a_i \in \mathbb{Q}_{\frac{m}{2^n}}\right\}$$

Сложение и умножение элементов из $\mathbb{Q}^{\infty}_{\frac{m}{2^n}}(\xi)$ определется покомпонентно, аналогично операциям над элементами $\mathbb{Q}^{\infty}_{\xi}$ [6].
Аналогично [6] будем называть элемент $\beta^{-1}(\xi) \in \mathbb{Q}^{\infty}_{\frac{m}{2^n}}(\xi)$ обратным к $\beta(\xi) \in \mathbb{Q}^{\infty}_{\frac{m}{2^n}}(\xi)$ если выполняется следующее соотношение:

$$\beta(\xi) \cdot \beta^{-1}(\xi) = 1$$

Несложно видеть, что обратимыми будут являться ряды, свободный коэффициент которых является обратимым элементом в $\mathbb{Q}_{\frac{m}{2^n}}$ (а именно - степенью двойки), то есть:

$$\forall \beta(\xi) \in \mathbb{Q}^{\infty}_{\frac{m}{2^n}}(\xi), \exists \beta^{-1}(\xi) \iff \exists k \in \mathbb{Z} : \beta(0) = 2^k$$

Кольцо многочленов над $\mathbb{Q}_{\frac{m}{2^n}}$ будем обозначать $\mathbb{Q}_{\frac{m}{2^n}}[\xi]$, а для обозначения того, что многочлены $P(\xi), Q(\xi) \in \mathbb{Q}_{\frac{m}{2^n}}[\xi]$ взаимно просты будем использовать запись $\gcd\left(P\left(\xi\right), Q\left(\xi\right)\right) = 1$.

Аналогично $\mathbf{R}(\mathbb{Q})$[6] определим множество дробно-рациональных функций от переменной ξ с коэффициентами из $\mathbb{Q}_{\frac{m}{2^n}}$ следующим образом:

$$\mathbf{R}(\mathbb{Q}_{\frac{m}{2^n}}) = \left\{ \frac{P(\xi)}{Q(\xi)} \mid P(\xi), Q(\xi) \in \mathbb{Q}_{\frac{m}{2^n}}[\xi], Q(0) = 1, \gcd\left(P(\xi),\ Q(\xi)\right) = 1 \right\}$$

Можно заметить, что $\mathbf{R}(\mathbb{Q}_{\frac{m}{2^n}})$ является подкольцом в кольце $\mathbb{Q}_{\frac{m}{2^n}}$.

Приведем формулировки следующих лемм без доказательств, поскольку они аналогичны доказательствам в работе[6]:

Лемма 1. $\forall V(x_1, ..., x_l) \in L(\mathbb{Q}_{\frac{m}{2^n}}), \exists R_0, R_1, ..., R_l \in \boldsymbol{R}(\mathbb{Q}_{\frac{m}{2^n}})$, *такие что:*

$$V(x_1, ..., x_l) = R_0 + \textstyle\sum_{i=1}^{l} R_i \cdot x_i.$$

Лемма 2. $\forall V(x_1, ..., x_l) \colon (\mathbb{Q}^{\infty}_{\frac{m}{2^n}})^l \to \mathbb{Q}^{\infty}_{\frac{m}{2^n}}$, *такого что:*

$$\begin{gathered} V(x_1, ..., x_l) = R_0 + \textstyle\sum_{i=1}^{l} R_i \cdot x_i \\ R_i \in \boldsymbol{R}(\mathbb{Q}_{\frac{m}{2^n}}), i \in [0, l] \end{gathered}$$

верно, что $V(x_1, ..., x_l) \in L(\mathbb{Q}_{\frac{m}{2^n}})$.

Множители R_k будем называть коэффициентами отображения.

В настоящей работе, без ограничения общности, не будем различать автоматы из $L(\mathbb{Q}_{\frac{m}{2^n}})$ и отображения, которые они реализуют.

Рассмотрим задачу A-полноты[2] системы линейных автоматов в $L(\mathbb{Q}_{\frac{m}{2^n}})$. Система линейных автоматов M будет называться A-полной, если $\forall V \in L(\mathbb{Q}_{\frac{m}{2^n}})$ и $\forall \tau \in \mathbb{N}$, в $K(M)$ существует автомат V', совпадающий с автоматом V на словах длины τ. A - замыкание системы M будем обозначать через $A(M)$.

3. Вспомогательные обозначения

Для любого $R \in \mathbf{R}(\mathbb{Q}_{\frac{m}{2^n}})$, через $R(t)$ будем обозначать коэффициент ряда при ξ^t.

$\forall a \in \mathbb{Q}_{\frac{m}{2^n}}, \forall b \in \mathbb{Z}$, будем говорить что $a \vdots b$, в случае если числитель числа a кратен b как целое число.

Определим несколько вспомогательных классов, о которых далее будут сформулированы утверждения.

1) Зафиксируем конечное множество $\mathbf{P}$ простых чисел, отличных от двойки:

$$\mathbf{P} = \{p_i | p_i \neq 2 \text{ - простое число, } i \in [1, k]\}.$$

Будем говорить, что автомат $V \in L(\mathbb{Q}_{\frac{m}{2^n}})$, такой что:

$$V(x_1, .., x_l) = R_0 + \sum_{i=1}^{l} R_i \cdot x_i$$
$$\forall i \in [0, l], R_i \in \mathbf{R}(\mathbb{Q}_{\frac{m}{2^n}})$$

обладает $\mathbf{P}$-свойством, если с точностью до переименования входов автомата V выполнено хотя бы одно из следующих условий:

а) $\exists i \in [1, k] : R_j(0) \vdots p_i, \forall j \in [1, l]$

б) $R_j(0) \vdots p_1 \cdot p_2 \cdot ... \cdot p_k, \forall j \in [2, l]$

Очевидно, что все константные и одноместные автоматы обладают $\mathbf{P}$-свойством.

Определим множество $V_{\mathbf{P}}$ следующим образом:

$$V_{\mathbf{P}} = \{V \ | V \in L(\mathbb{Q}_{\frac{m}{2^n}}), V - \text{ обладает } \mathbf{P} \text{ - свойством } \}$$

2) Будем говорить, что автомат $V(x_1, ..., x_l) \in L(\mathbb{Q}_{\frac{m}{2^n}})$, который реализует отображение:

$$V(x_1, .., x_l) = R_0 + \sum_{i=1}^{l} R_i \cdot x_i$$
$$\forall i \in [0, l], R_i \in \mathbf{R}(\mathbb{Q}_{\frac{m}{2^n}})$$

обладает **D**-свойством, если с точностью до переименования входов автомата V выполнено хотя бы одно из следующих условий:

а) $l \leq 1$

б) $\exists p > 2$ - простое : $R_i(0) \vdots p$;

Определим класс D следующим образом:

$$D = \{V \ | V \in L(\mathbb{Q}_{\frac{m}{2^n}}), V- \text{ обладает } \mathbf{D} \text{ - свойством } \}$$

3) $\forall p > 2$, где p - простое число, определим класс M_p следующим образом:

$$M_p = \Big\{V(x_1,..,x_l) \in L(\mathbb{Q}_{\frac{m}{2^n}}) | V(x_1,..,x_l) = R_0(\xi) + \textstyle\sum_{i=1}^{l} R_i(\xi) \cdot x_i, \\ R_i(1) \vdots p, \forall i \in [1,l]\Big\}$$

4) $\forall p > 2$, где p - простое число, определим класс T_p следующим образом:

$$T_p = \Big\{V(x_1,..,x_l) \in L(\mathbb{Q}_{\frac{m}{2^n}}) | V(x_1,..,x_l) = R_0(\xi) + \textstyle\sum_{i=1}^{l} R_i(\xi) \cdot x_i, \\ R_0(0) \vdots p\Big\}$$

5) Определим класс T_{int} следующим образом:

$$T_{int} = \Big\{V(x_1,..,x_l) \in L(\mathbb{Q}_{\frac{m}{2^n}}) | V(x_1,..,x_l) = R_0(\xi) + \textstyle\sum_{i=1}^{l} R_i(\xi) \cdot x_i, \\ R_i(0) \in \mathbb{Z}, \forall i \in [1,l]\Big\}$$

6) Определим класс $T_{\geq 0}$ следующим образом:

$$T_{\geq 0} = \Big\{V(x_1,..,x_l) \in L(\mathbb{Q}_{\frac{m}{2^n}}) | V(x_1,..,x_l) = R_0(\xi) + \textstyle\sum_{i=1}^{l} R_i(\xi) \cdot x_i, \\ R_i(0) \geq 0, \forall i \in [1,l]\Big\}$$

4. Основные результаты

Множество линейных автоматов из $L(\mathbb{Q}_{\frac{m}{2^n}})$ арности ≤ 1 обозначим V^1. Далее приводятся основные результаты без доказательства.

Лемма 3. $\forall k \in \mathbb{N}, \forall \boldsymbol{P} = \{p_i | p_i \neq 2$ - *простое число,* $i \in [1,k]\}$, $\forall$ *простого* $p > 2$, *верно что:*

$V_{\boldsymbol{P}}$, D, M_p, T_p, T_{int}, $T_{\geq 0}$ — *A-предполные классы.*

Лемма 4. *Пусть* $M \subset L(\mathbb{Q}_{\frac{m}{2^n}})$ - *конечная система.Если*$\forall \boldsymbol{P}$ - *конечного подмножества простых чисел, не содержащего двойку* $M \not\subseteq V_{\boldsymbol{P}}$, *то* $M \not\subseteq D$.

Теорема 1. *Пусть* $M \subset L(\mathbb{Q}_{\frac{m}{2^n}})$ - *конечная система.Если* $\forall \boldsymbol{P}$ - *конечного подмножества простых чисел, не содержащего двойку* $M \not\subseteq V_{\boldsymbol{P}}$, *то* $A(M \cup V^1) = L(\mathbb{Q}_{\frac{m}{2^n}})$.

Теорема 2. *Пусть* $M \subset L(\mathbb{Q}_{\frac{m}{2^n}})$ - *конечная система. Если* $\forall p > 2$ - *простого, выполняется:*

$$M \not\subseteq T_p, M_p, T_{int}, T_{\geq 0},$$

то $A(M \cup \{V_{\bigoplus}(x,y)\}) = L(\mathbb{Q}_{\frac{m}{2^n}})$.

Автор выражает признательность своему научному руководителю, доценту кафедры МаТИС механико-математического факультета МГУ, к.ф.-м.н., Часовских Анатолию Александровичу, за постановку задачи и помощь в проведении исследования.

Список литературы

[1] Кудрявцев В.Б., Алешин С.В., Подколзин А.С., *Введение в теорию автоматов*, НАУКА, Москва, 1985, 320 с.

[2] Буевич В.А., "О полноте, A-полноте и t-полноте в классе автоматных отображений", *Интеллектуальные системы*, **10**:1-4 (2006), 613–638

[3] Бабин Д.Н., Летуновский А.А., "О возможностях суперпозиции, при наличии в базисе автоматов фиксированной добавки из булевых функций и задержки", *Интеллектуальные системы. Теория и приложения*, **19**:3 (2015), 15–22

[4] Часовских А.А., “Проблема полноты для класса линейно-автоматных функций”, *Дискретная математика*, **27**:2 (2015), 134–151

[5] Chasovskikh A.A., “Completeness problem for the class of linear automata functions”, *Discrete Mathematics and Applications*, **26**:2 (2016), 89–104

[6] Ронжин Д.В., “Линейные автоматы над полем рациональных чисел”, *Интеллектуальные системы. Теория и приложения*, **21**:4 (2017), 144–155

A-completeness of systems with additives of linear automata over the ring of dyadic rationals

Ronzhin D.V.

A brief description of results of A-completeness problem for linear finite state automata, functioning over the ring of dyadic rationals is given. Conditions of A-completeness for systems, containing all one-valued linear automata with finite additive and finite systems with a summing automaton are stated.

Keywords: finite state automata, linear automata, dyadic rationals, A-completeness, maximum subclasses.

Часть 4.
Материалы семинаров кафедры МаТИС

Доклады семинара «Теория автоматов»

В третьем и четвёртом кварталах 2019 года на научном семинаре «Теория автоматов» под руководством академика Валерия Борисовича Кудрявцева состоялось 9 докладов.

25 сентября 2019 года

Задача о поиске оптимального положения бруса и её приложения

студент Нерсисян С. А.

Рассматривается задача о поиске положения многомерного прямоугольника, покрывающего максимальное количество точек из данного конечного множества. В докладе будет изложено доказательство NP-трудности задачи и приближенный полиномиальный алгоритм её решения. В качестве приложения будет рассмотрен подход к задаче кластеризации данных. Также будет рассказано о переформулировке полученного метода кластеризации на язык непрерывной статистики, позволяющей существенно уменьшить вычислительную сложность решения при использовании EM-алгоритма.

2 октября 2019 года

Четыре вида подалгебр и сложность задачи удовлетворения ограничениям

с.н.с. Жук Д. Н.

В 2017 году сложность задачи удовлетворения ограничениям на конечном множестве была описана для любого языка допустимых ограничений, а именно было показано, что эта задача решается за полиномиальное время только тогда, когда язык ограничений допускает слабую функцию почти единогласия. Одно из решений, полученное автором, основано на следующей идее. Любая нетривиальная алгебра, содержащая слабую функцию почти единогласия, имеет нетривиальную подалгебру одного из четырех видов: поглощающую, центральную, линейную, или

полиномиально-полную. В докладе мы посмотрим на эти подалгебры с алгебраической точки зрения и рассмотрим удивительные свойства таких подалгебр. Например, мы покажем, что только подалгебры одного типа могут давать в пересечении пустое множество. В конце мы объясним, почему именно линейные подалгебры представляют основную сложность при решении задачи удовлетворения ограничениям.

9 октября 2019 года

О некоторых теоретических и прикладных задачах распознавания образов

с.н.с. Мазуренко И. Л.

В докладе будет просуммирован опыт коллектива прикладных разработчиков в области создания промышленных систем компьютерного зрения с точки зрения составления списка открытых математических задач в данной области.

16 октября 2019 года

О сложности разбора слов в контекстно-свободных грамматиках

доцент Боков Г. В.

Контекстно-свободные грамматики (КСГ) имеют широкое применение в области компьютерных наук. Поскольку грамматические структуры многих языков программирования задаются такими грамматиками, парсеры этих языков зачастую сводятся к разбору слов в КСГ. В докладе будет рассказано о сложности существующих алгоритмов разбора слов в КСГ, о связи проблемы разбора слов в КСГ с другими алгоритмическими проблемами такими, например, как умножение целочисленных и булевых матриц. Кроме того, в докладе будет предложен новый алгоритм разбора слов в КСГ.

23 октября 2019 года

Некоторые свойства латинских квадратов, порожденных правильными семействами функций

с.н.с. Галатенко А. В.

Латинские квадраты и задаваемые с их помощью квазигруппы являются перспективной структурой для реализации различных криптографических примитивов. Для удовлетворения потребностей криптографии необходимо строить большие семейства латинских квадратов высоких порядков, с рядом дополнительных свойств, таких как ограничение на структуру подквазигрупп или на максимальную степень в полиномиальном представлении. Одно из возможных решений – конструкция на основе правильных семейств функций, введенных В. А. Носовым. В докладе будут представлены полученные совместно с В. А. Носовым и А. Е. Панкратьевым результаты о подквазигруппах и квадратичных квазигруппах.

13 ноября 2019 года

Интеллектуальная транспортная система: задачи и методы

в.н.с. Алисейчик П. А.

Доклад представляет собой обзор спектра математических задач, возникающих при анализе информации, полученной с камер наблюдения за дорожным движением, а также информации о движении транспортных средств, полученной из других источников. В первую очередь внимание уделяется задачам, при решении которых применимы методы искусственного интеллекта: распознавание транспортных средств, точное определение их положения и траекторий движения, отслеживание движения на карте местности, анализ поведения групп транспортных средств, хранение больших объемов видео- и другой информации.

20 ноября 2019 года

Компьютерное моделирование логических процессов

профессор Подколзин А. С.

В докладе будет рассказано о последних достижениях в решателе компьютерных задач и представлена общая схема функционирования решателя. Более подробно будет рассказано о логическом программирующем выводе и его роли в решателе.

27 ноября 2019 года

О задаче полноты для автоматов.

профессор Бабин Д. Н.

Сети автоматных функций - это возможная альтернатива нейросетей. Препятствием к широкому применению автоматов, является то, что задача полноты для автоматных функций относительно суперпозиции и обратной связи о общем виде не решена. Имеется два подхода к этой задаче. Подход к полноте абстрактных автоматов подразумевает наличие всех функций без памяти, образующих интерфейсы между соответствующими выходными и входными алфавитами автоматов схемы. А этом случае функции без памяти и "задержка"образуют полную систему. Обобщением этого факта является теорема Летичевского (1961), которая является критерием полноты конечной системы автоматов при наличии всех функций без памяти и других дополнительных условий. Подход к полноте структурных автоматов подразумевает использование произвольных автоматов Мили, входы и выход которых — суть декартовы произведения множества $\{0, 1\}$. Этот случай даёт основные парадоксы теории автоматов: континуум предполных классов (Кудрявцев 1963), алгоритмическая неразрешимость задачи полноты (Кратко 1964). Компромиссным вариантом в этой задаче является понятие A-полноты исследованное Буевичем В.А., когда получаемая схемой автоматная функция совпадает с желаемой лишь до наперёд заданного момента времени. В этом случае число предполных классов — счётно, а в случае фиксированного сколь угодно большого момента времени — даже конечно. Для решения этой задачи В.А. Буевичу удалось перенести язык сохранения предикатов, используемый в задаче полноты булевых функций,

на предикаты, растянутые во времени. Научное наследие учёного чрезвычайно обширно и разнообразно по проблематике. А-полнота и другие результаты Буевича В.А. включены в спецкурсы многих университетов. Его работу продолжили его ученики: Богомолов С.А., Ключников А.В., Лялин И.В., Насыров А.З., Наеф М., Подколзина М.А., Родин А.А. В последствии автору, пользуясь методом Буевича, удалось отделить в задаче полноты алгоритмически разрешимый случай от неразрешимого. Искренне сожалею о потере выдающегося учёного, добрые воспоминания о нём помогут нам двигаться дальше.

4 декабря 2019 года

О восстановлении трехмерного изображения по неполным данным о координатах проекций его точек

с.н.с. Алексеев Д. В.

Рассматривается случай, когда у одной из проекций присутствует информация только об одной из координат. Показано, что при некоторых дополнительных условиях возможно восстановление исходного трехмерного изображения с точностью до аффинной эквивалентности. Такая задача возникает, когда для восстановления формы объекта используется структурная линейно-бахромчатая подсветка. Также в докладе рассматривается задача порождения линейно-бахромчатой подсветки, обладающей заданными свойствами. Показана связь последней задачи с построением графов де Брёйна.

Доклады семинара «Вопросы сложности алгоритмов поиска»

В осеннем семестре 2019 – 2020 учебного года на научном семинаре «Вопросы сложности алгоритмов поиска» под руководством профессора Эльяра Эльдаровича Гасанова состоялось 14 докладов.

4 сентября 2019 года

О прогнозировании сверхслов автоматами

асп. Ведерников И. К.

В докладе рассматривается задача прогнозирования сверхслов. Сперва вводятся основные классы автоматов, после чего доказывается, что наилучшая степень прогнозирования достигается в классе автоматов с размеченными функциями выхода. Затем приводятся критерий частичного прогнозирования общерегулярных сверхсобытий и критерий почти полного прогнозирования одноэлементных сверхсобытий. В конце доклада разбираются некоторые примеры иллюстрирующие перечисленные выше теоремы.

11 сентября 2019 года

Оценки энергопотребления объёмных схем

асп. Ефимов А. А.

В ряде работ исследовалась сложность схем из функциональных элементов, реализующих функции алгебры логики от n аргументов. Автор использует такое понятие сложности схемы, как потенциал. Он равен среднему значению количества единиц на всех внутренних узлах схемы. Неформально говоря, потенциал играет роль средней «энергии» схемы, необходимой для её функционирования. Данная работа посвящена кубическим схемам, которые определяются аналогично плоским схемам, но в манхэттенском пространстве. Цель доклада состоит в получении верхней оценки потенциала объёмных схем, реализующих булевы функции и операторы из различных классов. Также целью является получение нижней оценки потенциала объёмных схем, из некоторого класса булевых операторов.

18 сентября 2019 года

Об одном алгоритме перевода текста правил дорожного движения в формальную модель

асп. Менькин М. И.

Предлагается один из возможных подходов к автоматизированному переводу текста юридического документа в формальную модель на примере Постановления Правительства "О правилах дорожного движения". Перечислены основные сущности юридического документа, вводятся понятия графа дороги, дорожной ситуации, шаблона правила, синтаксического шаблона, разрешимости дорожной ситуации, в совокупности образующие формальную теоретико-графовую модель. Обозначены основные процедуры алгоритма перевода текста правил дорожного движения в данную модель. Приведены примеры разбора предложений, в которых идёт речь о манёвре "Уступить дорогу".

2 октября 2019 года

Комбинаторно-геометрические свойства полиэдров задач комбинаторной оптимизации

д.ф-м.н. Максименко А. Н.

Примерами задач комбинаторной оптимизации с линейной целевой функцией являются задача о кратчайшем пути, задача коммивояжера, задача о рюкзаке и многие другие. Каждая такая задача может быть естественным образом переформулирована в виде задачи линейного программирования, область допустимых решений которой представляет собой выпуклый многогранник. С одной стороны, такая интерпретация позволяет использовать для решения этих задач геометрические методы. С другой стороны, и именно об этом и идет речь в докладе, этот подход позволяет выявить комбинаторно-геометрические свойства задач, характеризующие их сложность в различных классах алгоритмов. В докладе также рассмотрен метод аффинной сводимости, позволяющий сравнивать эти свойства.

9 октября 2019 года

Расшифровка булевых функций из замкнутых классов Поста запросами на значение и запросами на сравнение

асп. Быстрыгова А. В.

В докладе рассматривается задача точной расшифровки функций из замкнутых классов Поста при помощи запросов на значение и запросов на сравнение. Приводится критерий возможности расшифровки запросами на сравнение. Показано, что для классов, которые возможно расшифровать запросами на сравнение, сложность расшифровки не хуже, чем таковая при использовании запросов на значение. Также приводятся примеры классов, для которых сложность расшифровки запросами на сравнение строго меньше сложности расшифровки запросами на значение.

16 октября 2019 года

О конструировании клеточными автоматами двунаправленного движения на луче

студ. Кузнецова Е. В.

В докладе рассматривается вопрос числа состояний клеточного автомата, моделирующего движение точки на одномерном бесконечном экране. Приводится верхняя и нижняя оценка количества состояний данного клеточного автомата.

23 октября 2019 года

Об одном алгоритме достижения консенсуса в криптовалютах

д.ф.-м.н., проф. Гасанов Э.Э, студ. Суюнбекова М.Б.

Предлагается новый алгоритм достижения консенсуса в криптовалютах. Алгоритм основан на выработке игроками совместного случайного числа, на основе которого определяется выигравший игрок. Вероятность выигрыша каждого игрока зависит от размера вложенных средств. При выработке совместного случайного числа каждый игрок обменивается информацией с некоторым числом игроков, и это число рано логарифму от общего числа игроков. Каждый обмен информации основывается

на протоколе "Подбрасывания монеты по телефону"Мануэля Блума. В отличии алгоритма достижения консенсуса, принятого в биткоин, предложенный алгоритм не требует существенных вычислительных ресурсов. Предложена схема организации блокчейна при данном алгоритме достижения консенсуса.

30 октября 2019 года

Оценки кодового расстояния для классических и квантовых локальных кодов

к.ф.-м.н. Калачев Г. В.

В докладе рассматриваются семейства локальных кодов, исправляющих ошибки. Семейство локальных кодов — это семейство линейных кодов, заданных проверочной матрицей, у которых биты можно разместить в D-мерном пространстве таким образом, чтобы все биты, участвующие в одном проверочном соотношении, были расположены в шаре фиксированного радиуса. Известна верхняя оценка на кодовое расстояние и размерность d-мерных локальных кодов $d = O((n/k)^D)$.

В докладе рассмотрен пример кодов, где сообщение, заданное последовательностью из 2^t бит кодируется путём применения к нему одномерного линейного клеточного автомата Rule 60 2^t раз. Закодированное слово состоит из всех состояний клеточного автомата в моменты $0, 1, ..., 2^t - 1$. Показано, что кодовое расстояние такого кода по порядку равно 3^t, то есть $n^{(\log_2 3)/2}$, где $n \asymp 4^t$ — длина кодового слова.

Для аналогичных кодов, задаваемых другими линейными клеточными автоматами, порядок роста кодового расстояния неизвестен. Гипотеза заключается в том, что для любого линейного клеточного автомата эволюция одноклеточной конфигурации содержит минимальное по порядку количество клеток в состоянии 1.

6 ноября 2019 года

Оценки кодового расстояния для некоторых семейств локальных квантовых кодов

к.ф.-м.н. Калачев Г. В.

В квантовых компьютерах все операции выполняются с определённой вероятностью ошибки, и поэтому для обеспечения надёжных квантовых вычислений необходимо защищать квантовое состояние. Для этого

предполагается использовать квантовые коды для коррекции ошибок. В докладе рассматриваются стабилизаторные CSS коды, задаваемые двумя проверочными матрицами H_x и H_z такими, что $H_x H_z^T = 0$. Особый интерес представляют локальные коды, у которых кубиты можно так разместить в D-мерном пространстве, чтобы все кубиты, участвующие в одном проверочном соотношении, были расположены в шаре фиксированного радиуса. Для вычисления синдромов локальных кодов требуются лишь локальные взаимодействия кубитов, поэтому такие коды более эффективны с точки зрения физической реализации.

Для квантовых LDPC кодов и, в частности, для локальных кодов лучшая известная оценка на кодовое рассояние $\Theta(\sqrt{n\sqrt{\log n}})$. Известно несколько семейств LDPC кодов, в том числе, локальных, у которых расстояние равно $\Theta(\sqrt{n})$. Для того, чтобы найти коды с более высоким расстоянием, требуется рассматривать другие классы кодов, для которых расстояние гипотетически может быть выше, чем $\sqrt{n}$. Есть гипотеза, что для 3-мерных фрактальных кодов может достигаться кодовое расстояние по порядку выше $\sqrt{n}$, однако для таких кодов лучшие известные нижние оценки $\Omega(\sqrt[3]{n})$. В докладе приведён пример кодов с фрактальной структурой ошибок, для которых удалось доказать нижнюю оценку $\Omega(n^\alpha)$, $\alpha = \log_2(2(\sqrt{5}-1))/3 \approx 0.435$. Также описаны задачи, связанные с клеточными автоматами, решение которых могло бы улучшить эту оценку. Среди них:

1) Оценка снизу числа единиц в эволюции линейного клеточного автомата Rule 150 на протяжении времени 2^t, начиная с произвольного ненулевого состояния.

2) Рассматривается клеточный автомат Rule 150 со входом: у каждой клетки есть вход, на который можно подавать 0 или 1. Значение входа прибавляется к состоянию по модулю 2. Какое наименьшее число единиц (в сумме за всё время эволюции) нужно нужно подать на входы клеточного автомата, чтобы нулевое состояние через некоторое время перешло в состояние, где подряд 2^t клеток в состоянии 1?

13 ноября 2019 года

Верхняя оценка мощности плоских схем, реализующих функции с ограниченным числом единиц

к.ф.-м.н. Калачев Г. В.

Доклад состоялся в рамках первого заседания межфакультетского научного семинара "Актуальные математические задачи, связанные с проектированием СБИС" под руководством зав. кафедрой математической кибернетики факультета ВМК Ложкина С. А. и профессора кафедры МаТИС механико - математического факультета Гасанова Э. Э.

Рассматривается задача синтеза плоских схем, реализующих булевы функции с ограниченным числом единиц плоскими схемами. $\mathcal{F}_n^N$ — класс всех булевых функций от n переменных, принимающих значение 1 не более, чем на N наборах. Из мощностных соображений можно убедиться, что площадь схем для почти всех функций не может быть меньше $N(n-\log_2 N)$ при $N \leq 2^{n-1}$. Кроме площади схем интерес представляет также энергопотребление, которому в модели плоских схем соответствует такая мера сложности, как потенциал. Потенциал схемы на входном наборе x — это количество единиц в узлах схемы, когда на вход подан набор x. В докладе рассматривается максимальный потенциал схемы, равный максимуму потенциала по всевозможным входным наборам.

Приводится метод синтеза плоских схем, реализующих функции из $\mathcal{F}_n^N$ схемами с площадью $O(N(n-\log_2 N))$ и максимальным потенциалом $O(\sqrt{N(n-\log_2 N)n/\log_2 N})$. Сначала рассматривается случай $N \geq 2^{n/3}$, и строится схема с потенциалом $O(\sqrt{N(n-\log_2 N)})$. Затем рассматривается случай $N < 2^{n/3}$, и он сводится к последовательному вычислению $O(n/\log_2 N)$ функций от $3\log_2 N$ переменных, для которых используется схема, построенная на первом шаге, в итоге вся схема имеет требуемый потенциал.

20 ноября 2019 года

О скорости роста одной колонии жуков

студ. Воротников А. С.

Рассмотрим следующую динамическу систему: дано бесконечное поле с ненулевым однородно расположенным запасом еды. На этом поле появляется жук, который перемещается по полю, ест имеющуюся еду и размножается. Поле моделируется целочисленной решёткой на плоскости,

в которой каждому узлу в начальный момент сопоставлено некоторое одинаковое количество еды. Эту целочисленную решётку в дальнейшем будем называть картой. Жук действует по алгоритму, который упрощённо представляется схемой: искать еду -> есть, пока не насытишься (если еды не хватило, снова искать) -> размножаться. На все действия расходуется энергия, которая получается жуком из еды, расположенной на поле. Если запас энергии жука падает ниже ноля, то жук умирает — исчезает с поля. Под размножением подразумевается деление жука на двух одинаковых жуков, обладающих половиной запаса энергии родителя.

Для некоторого класса прямых хочется построить класс колоний, такой, что для произвольной прямой найдётся колония, чей график численности бесконечное число раз пересекает выбранную прямую. Подзадача — описать такой класс прямых.

Указанная система моделируется однородными структурами. Предлагается использовать подход автоматного моделирования.

Сначала выбирается пять условий, которым должна удовлетворять колония, затем показывается, к каким ограничениям приводят эти условия. Далее следует анализ поведения жуков на карте, из которого вытекает необходимость корректировки некоторых условий. В результате непосредственного разбора поведения всех жуков, удаётся выписать явную функцию численности популяции с некоторого достаточно большого момента времени. Оказывается, что она лежит в конусе, ограниченном двумя прямыми, причём достигает границ конуса бесконечное число раз.

Таким образом удалось для произвольной прямой из класса

$$\mathcal{A} = \{y = ax + b | 0 < a \leqslant \frac{40}{11}, b \in \mathbb{R}\}$$

построить такую колонию, что её график численности популяции бесконечное число раз пересечёт выбранную прямую.

Обозначены требующие дальнейшего детального разбора способы расширения класса прямых до класса

$$\mathcal{B} = \{y = ax + b | a > 0, b \in \mathbb{R}\}.$$

27 ноября 2019 года

Моделирование аэродинамики крыла клеточными автоматами

студ. Гордеева А. С.

В работе исследуется задача моделирования движения полета крыла в воздушном потоке. Предлагается для моделирования использовать клеточные автоматы. Причем считается, что имеются клеточные автоматы, моделирующие движение воздуха, и имеется автомат, моделирующий крыло. Крыло имеет некоторую форму. Клеточные автоматы изображают прямолинейное движение частиц, но при столкновении с крылом «обтекают» его, причем скорость частиц, движущихся по более длинной стороне крыла, больше чем у частиц с другой стороны. Из-за этого возникает подъемная сила. Автомат, моделирующий крыло, «видит» клеточные автоматы из некоторой окрестности крыла и высчитывает скорость частиц, их плотность. На основе этого вычисляется вектор подъемной силы. В результате крыло меняет свои координаты. Написана компьютерная программа, которая отображает функционирование предложенных автоматов. Проведены эксперименты с крыльями разных форм. Эксперименты показали адекватность модели.

4 декабря 2019 года

Об асимптотически точных оценках сложности реализации булевых функций клеточными схемами и методах их получения.

д.ф.-м.н., проф. Ложкин С. А.

Доклад состоялся в рамках второго заседания межфакультетского научного семинара "Актуальные математические задачи, связанные с проектированием СБИС" под руководством зав. кафедрой математической кибернетики факультета ВМК Ложкина С. А. и профессора кафедры МаТИС механико - математического факультета Гасанова Э. Э.

В докладе представлен обзор результатов и методов, связанных с получением асимптотически точных оценок сложности реализации булевых функций и операторов в некоторых моделях клеточных схем из функциональных элементов и контактных схем.

11 декабря 2019 года

Построение кратчайших путей клеточными автоматами с эхом.

д.ф.-м.н., проф. Гасанов Э. Э., студ. Пропажин А. А.

В докладе вводится новый объект — клеточный автомат с эхом. Каждый клеточный автомат с эхом в каждый такт может послать в эфир некоторый сигнал. Алфавит эфира является конечной аддитивной коммутативной полугруппой, а сам эфир представляет собой потенциально бесконечный сумматор сигналов клеточных автоматов, где в качестве суммы выступает определяющая операция данной полугруппы. В тот же такт каждый клеточный автомат получает из эфира суммарный сигнал и, учитывая его, изменяет свое состояние. Введение эфира и возможности посылать в эфир сигналы позволяет мгновенно передавать сигналы на любые расстояния, и тем самым позволяет одному клеточному автомату управлять поведением сколь угодно далеко удаленного от него другого клеточного автомата. Рассматривается также модификация клеточных автоматов с эхом, которые могут посылать сигналы в эфир по определенным направлениям и получать сигналы из эфира из определенных направлений. Иными словами, в этом случае у каждого клеточного автомата имеется несколько локаторов, направленных в разные стороны, и он может с помощью этих локаторов получать сигналы с определенных направлений и посылать сигналы по этим направлениям.

Исследуется задача поиска кратчайшего пути между двумя точками в среде с препятствиями. Приводится решение этой задачи с помощью клеточных автоматов с эхом. Проводится сравнение сложности решения этой задачи с помощью клеточных автоматов с эхом и с помощью обычных клеточных автоматов.

К сведению авторов публикаций в журнале «Интеллектуальные системы. Теория и приложения»

В соответствии с требованиями ВАК РФ к изданиям, входящим в перечень ведущих рецензируемых научных журналов и изданий, в которых могут быть опубликованы основные научные результаты диссертаций на соискание ученой степени доктора и кандидата наук, статьи в журнал «Интеллектуальные системы. Теория и приложения» предоставляются авторами в следующей форме:

1. Статьи, набранные в пакете LaTeX, предоставляются к загрузке через WEB-форму http://intsysjournal.org/generator_form .
2. К статье прилагаются файлы, содержащие название статьи на русском и английском языках, аннотацию на русском и английском языках (не более 50 слов), список ключевых слов на русском и английском языках (не более 20 слов), информация об авторах: Ф.И.О. полностью, место работы, должность, ученая степень и/или звание (если имеется), контактные телефоны (с кодом города и страны), e-mail, почтовый адрес с индексом города (домашний или служебный).
3. Список литературы оформляется в едином формате, установленном системой Российского индекса научного цитирования.
4. За публикацию статей в журнале «Интеллектуальные системы. Теория и приложения» с авторов (в том числе аспирантов высших учебных заведений) статей, рекомендованных к публикации, плата не взимается. Оттиски статей авторам не предоставляются. Журнал распространяется по подписке, экземпляры журнала рассылаются подписчикам наложенным платежом. Условия подписки публикуются в каталоге НТИ «Роспечать», индекс журнала 64559.
5. Доступ к электронной версии последнего вышедшего номера осуществляется через НЭБ «Российский индекс научного цитирования». Номера, вышедшие ранее, размещаются на сайте http://intsysjournal.org, и доступ к ним бесплатный. Там же будут размещены аннотации всех публикуемых статей.

Подписано в печать: 09.12.2019
Дата выхода: 20.12.2019
Тираж: 200 зкз.
Цена свободная
Свидетельство о регистрации СМИ: ПИ № ФС77–58444 от 25 июня 2014 г., выдано Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций(Роскомнадзор).