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

Интеллектуальные Системы.

Теория и приложения

ТОМ **24** ВЫПУСК 1 * **2020**

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

О верификации формализованных математических доказательств

Вторушин Ю.И.[1]

Рассматривается алгоритм верификации формализованных математических доказательств. Основная его часть формулируется в виде продукционной системы с метапеременными. Доказываются теоремы о его корректности и полноте.

Ключевые слова: автоматическое доказательство теорем, система автоматизации дедукции, верификация доказательств, язык первого порядка, исчисление предикатов, продукционная система, искусственный интеллект.

Введение

Автоматическое доказательство теорем является весьма важной областью современных исследований [1]. В этой области, кроме задачи поиска доказательств теорем, другой важной проблемой является верификация полученных каким-либо способом формализованных доказательств [2]. В данной статье рассматривается вторая из указанных проблем. Основные определения и обозначения понятий, связанных с автоматическим доказательством теорем, можно найти в статьях [3, 4]. В настоящей работе мы будем использовать принятые там понятия и обозначения. В статьях [3, 4] рассматривается алгоритм поиска натурального вывода. В общем случае такие алгоритмы имеют экспоненциальную сложность, которые либо заканчивают свою работу и выдают искомый результат, либо заканчивают работу и выдают слово No, либо не заканчивают работу. В отличие от алгоритма поиска, алгоритм верификации формализованного математического текста должен быть эффективным разрешающим алгоритмом. Такой алгоритм должен всегда заканчивать свою работу

[1] *Вторушин Юрий Игоревич* — фронтенд-разработчик в DSS Lab, e-mail: urchick@mail.ru.

Vtorushin Yurii Igorevich — front-end developer in DSS Lab.

и выдавать слово Yes, если текст является доказательством, и No — в противном случае.

В статье [3] рассмотрены системы натуральной дедукции $\mathfrak{D}_i$, где $i = 1, 2, 3$. При этом случай $i = 1$ соответствует случаю классической логики, $i = 2$ — интуиционистской, а $i = 3$ — минимальной. Через $\mathcal{D}_i$ будем обозначать систему натуральной дедукции, которая получается из $\mathfrak{D}_i$ в результате удаления аксиом и правил вывода, описывающих предикат равенства. В этой статье мы ограничимся рассмотрением только исчислений $\mathcal{D}_i$ для $i = 1, 2, 3$.

Во многих системах автоматизации дедукции (САД) пользователь имеет возможность записывать математический текст в виде, близком к тому, как текст выглядит в обычной математической статье. Пользователь может не дробить доказательство до уровня правил вывода системы $\mathcal{D}_i$. Такие алгоритмы верификации в соответствии с [5] мы будем называть *алгоритмами очевидности.*

В соответствии с синтаксисом языка САД обоснование формулы Φ в тексте формальной статьи может быть сделано двумя способами:

$$\text{“}\Phi \text{ by } \alpha_1, \ldots, \alpha_m\text{”} \quad \text{и} \quad \text{“}\Phi \text{ proof } \ldots \text{ qed”}.$$

В первом случае речь идет о непосредственном обосновании, а во втором — о более сложном доказательстве или локальном подвыводе. При этом метки $\alpha_1, \ldots, \alpha_m$ должны идентифицировать некоторые формулы $A_1, \ldots, A_m$ в тексте статьи, которые должны быть в области видимости формулы Φ. В статье [3] все обоснования должны соответствовать аксиомам и правилам вывода системы натуральной дедукции $\mathcal{D}_i$. Здесь мы описываем более общий алгоритм очевидности, соответствующий требованиям [5].

В настоящее время разработано много алгоритмов очевидности, которые лежат в основе существующих САД. В качестве примера можно привести широко известные системы Mizar [6] и SAD [7]. Проект SAD выполняется в рамках работ по реализации программы [5]. Верификатор системы SAD может использовать уже разработанные различные алгоритмы очевидности для классической логики. Также алгоритм верификатора системы Mizar описан в литературе и реализован только для классической логики.

По нашему мнению, недостаток используемых алгоритмов очевидности в существующих САД заключается в том, что в них непосредственное обоснование формулы Φ из формул $A_1, \ldots, A_n$ не тождественно

с тем, что обычно математик понимает под этим. Кроме того, используемые методы верификации для разных логик описываются на основе отличающихся концепций. В этой работе описываются единообразно для всех трех логических систем $\mathcal{D}_i$ соответствующие им алгоритмы очевидности $\mathcal{E}_i$, которые лишены отмеченных недостатков.

Алгоритм очевидности формулируется в разделе 1. При этом основная часть алгоритма формулируется в виде продукционной системы с метапеременными. Тем самым на теоретическом уровне алгоритм задается с точностью до стратегии построения дерева поиска типа И/ИЛИ в рамках этой продукционной системы. Теоремы о корректности и полноте рассматриваемого в статье алгоритма очевидности формулируются и доказываются соответственно в разделах 2 и 3.

1. Алгоритм верификации доказательств

В этом разделе для каждой из трех систем натуральной дедукции $\mathcal{D}_i$ определяется алгоритм очевидности $\mathcal{E}_i$. Для этого, прежде всего, описывается алгоритм верификации непосредственных обоснований "Φ by $\alpha_1, \ldots, \alpha_m$". Этот алгоритм формулируется в виде продукционной системы $\mathcal{V}_i$. Алгоритм очевидности $\mathcal{E}_i$ включает $\mathcal{V}_i$ как часть. Тем самым алгоритм $\mathcal{E}_i$ задается с точностью до стратегии построения дерева поиска типа И/ИЛИ в рамках $\mathcal{V}_i$.

При построении данной теории мы следуем [4]. Задачи продукционной системы формулируются с использованием псевдоформул и псевдотермов некоторого фиксированного языка Ω. При этом свободные предметные переменные псевдоформул считаются константами языка Ω. Мы не будем различать псевдоформулы, отличающиеся лишь переименованием связанных переменных. Будем всегда предполагать, что рассматриваемые псевдоформулы обладают свойством чистоты переменных [4]. Для унифицируемых псевдоформул A и B через $\mathbf{MGU}(A, B)$ ниже обозначается их наиболее общий унификатор.

Предложениями будем называть пары вида $\langle \alpha, \Phi \rangle$, где α — метка и Φ — псевдоформула.

После сделанных определений теперь точным образом опишем продукционные системы $\mathcal{V}_i$ для $i = 1, 2, 3$. Задачами этих систем являются упорядоченные пары вида $(\Gamma \rhd \Psi)$, где Γ есть конечное множество предложений и Ψ есть псевдоформула. Примитивными задачами для $\mathcal{V}_1$ и $\mathcal{V}_3$

являются задачи следующих двух видов:

$$\langle \alpha_1,\, \Phi_1 \rangle \,\ldots\, \langle \alpha_i,\, \Phi_i \rangle \,\ldots\, \langle \alpha_m,\, \Phi_m \rangle \,\rhd\, \Phi_i,$$

$$\langle \alpha_1,\, \Phi_1 \rangle \,\ldots\, \langle \alpha_m,\, \Phi_m \rangle \,\rhd\, \top.$$

Продукционная система $\mathcal{V}_2$, кроме указанных примитивных задач, включает также примитивные задачи следующего вида:

$$\langle \alpha_1,\, \Phi_1 \rangle \,\ldots\, \langle \alpha_i,\, \bot \rangle \,\ldots\, \langle \alpha_m,\, \Phi_m \rangle \,\rhd\, \Psi.$$

Будем говорить, что содержащая метапеременные $X_1,\, \ldots,\, X_k$ дедуктивная задача P *примитивизируема*, если существует подстановка $\theta = \begin{pmatrix} X_1 & \ldots & X_k \\ t_1 & \ldots & t_k \end{pmatrix}$ вместо метапеременных $X_1,\, \ldots,\, X_k$ такая, что дедуктивная задача $P\theta$ является примитивной.

Прежде всего, в нашем случае задачу

$$\langle \alpha_1,\, \Phi_1 \rangle \,\ldots\, \langle \alpha_i,\, \Phi_i \rangle \,\ldots\, \langle \alpha_m,\, \Phi_m \rangle \,\rhd\, \Psi$$

примитивизируют подстановки θ вида $\mathbf{MGU}(\Psi,\, \Phi_i)$ в случае, когда псевдоформулы Ψ и Φ_i унифицируемы. Кроме того, если $\Psi = \top$ или $\Phi_i = \bot$, то задачу примитивизирует любая подстановка, в частности пустая ε.

Формулируемые ниже правила 1 – 6 определяют общие для систем $\mathcal{V}_i$ правила декомпозиции.

1. Если $\Gamma \,\rhd\, \Psi_1$ и $\Gamma \,\rhd\, \Psi_2$ суть две разрешимые задачи, у которых существуют совместимые допустимые подстановки θ_1 и θ_2, то тогда задача $\Gamma \,\rhd\, \Psi_1 \,\&\, \Psi_2$ также разрешима и ее допустимой подстановкой считается комбинация θ_1 и θ_2. Символически сформулированное правило декомпозиции запишем в виде

$$\frac{\Gamma \,\rhd\, \Psi_1; \qquad \Gamma \,\rhd\, \Psi_2}{\Gamma \,\rhd\, \Psi_1 \,\&\, \Psi_2}.$$

Это удобная запись того, что задача вида $\Gamma \,\rhd\, \Psi_1 \,\&\, \Psi_2$ сводится к двум задачам $\Gamma \,\rhd\, \Psi_1$ и $\Gamma \,\rhd\, \Psi_2$.

2. Если для $j = 1$ или $j = 2$ разрешима задача $\Gamma \,\rhd\, \Psi_j$ и θ — ее допустимая подстановка, то задача $\Gamma \,\rhd\, \Psi_1 \vee \Psi_2$ также считается разрешимой и θ считается ее допустимой подстановкой:

$$\frac{\Gamma \,\rhd\, \Psi_j}{\Gamma \,\rhd\, \Psi_1 \vee \Psi_2}.$$

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

$$\frac{\Gamma \rhd \Psi(X)}{\Gamma \rhd \exists x\, \Psi(x)}.$$

4. Если разрешимы две задачи $\Gamma \rhd \Phi_1 \supset \Psi$ и $\Gamma \rhd \Phi_2 \supset \Psi$, которые имеют совместимые допустимые подстановки θ_1 и θ_2, то разрешима также задача $\langle \alpha,\, \Phi_1 \vee \Phi_2 \rangle\, \Gamma \rhd \Psi$ и ее допустимой подстановкой считается комбинация θ_1 и θ_2:

$$\frac{\Gamma \rhd \Phi_1 \supset \Psi; \qquad \Gamma \rhd \Phi_2 \supset \Psi}{\langle \alpha,\, \Phi_1 \vee \Phi_2 \rangle\, \Gamma \rhd \Psi}.$$

5. Если для $j = 1$ или $j = 2$ разрешима задача $\langle \alpha,\, \Phi_j \rangle\, \Gamma \rhd \Psi$ и θ — ее допустимая подстановка, то задача $\langle \alpha,\, \Phi_1 \,\&\, \Phi_2 \rangle\, \Gamma \rhd \Psi$ также считается разрешимой и θ считается ее допустимой подстановкой:

$$\frac{\langle \alpha,\, \Phi_j \rangle\, \Gamma \rhd \Psi}{\langle \alpha,\, \Phi_1 \,\&\, \Phi_2 \rangle\, \Gamma \rhd \Psi}.$$

6. Если разрешимы две задачи $\Gamma \rhd \Phi_1$ и $\langle \alpha,\, \Phi_2 \rangle\, \Gamma \rhd \Psi$, у которых существуют совместимые допустимые подстановки θ_1 и θ_2, то разрешима также задача $\langle \alpha,\, \Phi_1 \supset \Phi_2 \rangle\, \Gamma \rhd \Psi$ и ее допустимой подстановкой считается комбинация θ_1 и θ_2:

$$\frac{\Gamma \rhd \Phi_1; \qquad \langle \alpha,\, \Phi_2 \rangle\, \Gamma \rhd \Psi}{\langle \alpha,\, \Phi_1 \supset \Phi_2 \rangle\, \Gamma \rhd \Psi}.$$

7. Если разрешима задача $\langle \alpha,\, \Phi(X) \rangle\, \Gamma \rhd \Psi$, где X — новая метапеременная в поисковом дереве, а θ — ее допустимая подстановка, то считается разрешимой задача $\langle \alpha,\, \forall x\, \Phi(x) \rangle\, \Gamma \rhd \Psi$ и θ считается ее допустимой подстановкой:

$$\frac{\langle \alpha,\, \Phi(X) \rangle\, \Gamma \rhd \Psi}{\langle \alpha,\, \forall x\, \Phi(x) \rangle\, \Gamma \rhd \Psi}.$$

Формулировка продукционных систем $\mathcal{V}_i$ завершена. Каждая продукционная система $\mathcal{V}_i$ 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 заключается в задании способа 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 соединяющей лист с корнем ветви с помощью операции комбинации [8]. Тем самым в ходе процесса построения дерева поиска у его вершин формируются наборы допустимых подстановок. В качестве символических названий сформулированных правил декомпозиции 1 – 7 будем соответственно использовать следующие обозначения: $(\triangleright\&)$, $(\triangleright\vee)$, $(\triangleright\exists)$, $(\vee\triangleright)$, $(\&\triangleright)$, $(\supset\triangleright)$, $(\forall\triangleright)$.

Если корень дерева поиска имеет хотя бы одну допустимую подстановку θ, то *соответствующее* этой подстановке *деривационное поддерево* Θ вычисляется согласно следующей процедуре. Прежде всего, включаем в Θ исходную задачу P. В случае, когда подстановка θ примитивизирует задачу P, поддерево Θ содержит вершину P в качестве листа. В противном случае мы перебираем у родительской вершины P все дочерние связки вершин. Пусть r — количество таких связок и $P_1^j,\ \ldots,\ P_{q_j}^j$ — одна из них, где $j = 1, \ldots, r$. Рассмотрим всевозможные комбинации подстановок $\theta_1,\ \ldots,\ \theta_{q_j}$, где каждая θ_s берется из набора допустимых подстановок задачи P_s^j. Среди связок дочерних вершин задачи P обязательно найдется хотя бы одна такая, что указанная комбинация совпадет с θ. Такую связку $P_1^j, \ldots, P_{q_j}^j$ добавим в текущее поддерево Θ. Затем этот процесс приме́ним для всех вершин $P_1^j,\ \ldots,\ P_{q_j}^j$ и соответствующих подстановок $\theta_1,\ \ldots,\ \theta_{q_j}$. Последнее из полученных поддеревьев и есть искомое деривационное поддерево Θ.

Теорема. *Алгоритм проверки непосредственных обоснований всегда завершает свою работу.*

Доказательство. Пусть n есть общее число вхождений всех подформул в исходную задачу

$$\langle\alpha_1,\, \Phi_1\rangle\ \ldots\ \langle\alpha_m,\, \Phi_m\rangle\ \triangleright\ \Psi. \tag{1}$$

Введем обозначение $h = 2 \cdot (m + 1)$. Чтобы установить применимость алгоритма к любой задаче (1) покажем, что дерево поиска конечно и содержит не более, чем $p_n(h) = 1 + h + h^2 + \ldots + h^n$ вершин.

Действительно, применение правил декомпозиции $(\triangleright\&)$, $(\triangleright\vee)$, $(\triangleright\exists)$, $(\vee\triangleright)$, $(\&\triangleright)$, $(\supset\triangleright)$, $(\forall\triangleright)$ всегда приводит к тому, что значение параметра n у каждой дочерней задачи становится меньше значения этого

параметра у родительской задачи. Отсюда следует, что, во-первых, высота дерева поиска T задачи (1) не превосходит n; во-вторых, у каждой родительской вершины может быть не более, чем h дочерних вершин. В то же время конечное дерево высоты n, у которого все отличные от листьев вершины имеют ровно h дочерних вершин, содержит $p_n(h)$ вершин. Таким образом, дерево поиска T задачи (1) конечно и содержит не более, чем $p_n(h)$ вершин. □

Сформулированный в [3] для $\mathcal{D}_i$ алгоритм очевидности теперь распространим на более сложный вариант непосредственных обоснований "Φ by $\alpha_1, \ldots, \alpha_m$", который верифицируется продукционной системой $\mathcal{V}_i$. Получаемый таким способом алгоритм очевидности будем обозначать $\mathcal{E}_i$. Новый алгоритм $\mathcal{E}_i$ теперь зависит от выбора стратегии построения дерева поиска типа И/ИЛИ в рамках $\mathcal{V}_i$.

В соответствии с [3] новый алгоритм очевидности $\mathcal{E}_i$ обрабатывает исходный текст следующим образом. Алгоритм $\mathcal{E}_i$ последовательно проверяет, во-первых, правильность непосредственных умозаключений, которые соответствует обоснованиям вида "Φ by $\alpha_1, \ldots, \alpha_m$"; во-вторых, правильность структуры доказательств, которые соответствуют обоснованиям вида "Φ proof ... qed". В последнем случае проверяется соответствие тезиса доказательства доказываемой формуле.

Таким образом, изменения касаются только процедуры обработки непосредственных обоснований "Φ by $\alpha_1, \ldots, \alpha_m$". Процедура обработки обоснований вида "Φ proof ... qed" не меняется. А именно, рассуждение "proof ... qed", кроме предложений и их обоснований вида "$\alpha : \Psi$ ...", может содержать также специфические фразы четырех видов: "let $\xi_1, \ldots \xi_k$", "consider $\xi_1, \ldots \xi_k$ such that $\alpha : \Psi$", "assume $\alpha : \Psi$" и "thus Ψ", где $\xi_1, \ldots \xi_k$ суть фиксируемые предметные переменные и Ψ есть формула. Алгоритм $\mathcal{E}_i$ составляет список всех допущений и заключений $\Psi_1, \Psi_2, \ldots, \Psi_n, \Psi_{n+1}$ рассуждения "proof ... qed", соответствующий фразам "assume $\alpha : \Psi_j$" и "thus Ψ_j". Затем $\mathcal{E}_i$ сопоставляет с формулой Φ прямой тезис

$$\forall\bar{\xi}_1(\Psi_1\,\lambda_1\,\forall\bar{\xi}_2(\Psi_2\,\lambda_2\,(\ldots\forall\bar{\xi}_n(\Psi_n\,\lambda_n\,\forall\bar{\xi}_{n+1}\Psi_{n+1}))\ldots)$$

рассуждения "proof ... qed", где $\bar{\xi}_j$ — список всех переменных из фраз "let ...", находящихся между пунктами Ψ_{j-1} и Ψ_j, а для связки λ_j имеет место

$$\lambda_j = \begin{cases} \&, & \text{если } \Psi_j \text{ — заключение,} \\ \supset, & \text{если } \Psi_j \text{ — допущение.} \end{cases}$$

Кроме того, для доказательств в рамках классической логики $\mathcal{D}_1$ в случае, когда последним заключением Ψ_{n+1} рассуждения "proof ... qed" является противоречие $\bot$, а также Ψ_n имеет вид $\neg\,\Theta$ и λ_n есть $\supset$, помимо прямого тезиза, рассматривается для сравнения с Φ также косвенный тезис

$$\forall\bar{\xi}_1(\Psi_1\,\lambda_1\,\forall\bar{\xi}_2(\Psi_2\,\lambda_2\,(\ldots\forall\bar{\xi}_{n-1}(\Psi_{n-1}\,\lambda_{n-1}\,\forall\bar{\xi}_n\Theta))\ldots).$$

Очевидным следствием теоремы о конечности поисковых деревьев у продукционной системы $\mathcal{V}_i$ и так сформулированного алгоритма очевидности $\mathcal{E}_i$ является следующее утверждение.

Следствие. *Алгоритм очевидности $\mathcal{E}_i$ всегда завершает свою работу.*

Замечание. Будем называть пару $\langle m,\, n\rangle$ *сложностью* дедуктивной задачи (1). С практической точки зрения можно считать, что параметр n ограничен некоторым достаточно большим натуральным значением N. Что фактически наблюдается в реальных математических статьях. Тогда, при проверке непосредственных обоснований, число вершин в дереве поиска не превосходит $p_N(h)$, а это значит, что алгоритм очевидности $\mathcal{E}_i$ можно считать полиномиальным.

В качестве примера рассмотрим формальное доказательство теоремы о том, что множество является пустым в том и только в том случае, когда оно является подмножеством любого множества. Подробное доказательство в рамках системы натуральной дедукции $\mathcal{D}_1$ выглядит следующим образом:

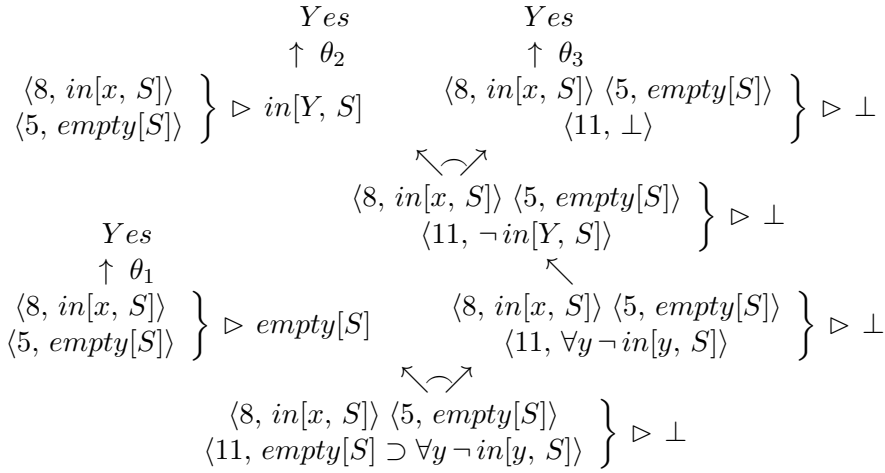
```
environ
  type Set;
  type Element;
  pred in[Element, Set];
  pred subset[Set, Set];
  pred empty[Set];
  reserve A, S, T for Set;
  reserve x, y, z for Element;
  1: for S,A holds subset[A,S] iff for x st in[x,A] holds in[x,S];
  2: for S holds empty[S] iff for x holds not in[x,S];
  3: ex S st empty[S];
text

  for S holds (for T holds subset[S,T]) iff empty[S]
proof
  let S be Set;
  4: (for T holds subset[S,T]) implies empty[S]
  proof
```

```
    assume 4: for T holds subset[S,T];
    5: empty[S] iff (for x holds not in[x,S]) by 2;
    6: (for x holds not in[x,S]) implies empty[S] by 5;
    7: for x holds not in[x,S]
    proof
      let x be Element;
      assume 7: in[x,S];
      consider U be Set such that 8: empty[U] by 3;
      9: empty[U] iff (for y holds not in[y,U]) by 2;
      10: empty[U] implies (for y holds not in[y,U]) by 9;
      11: for y holds not in[y,U] by 10,8;
      12: not in[x,U] by 11;
      13: subset[S,U] iff (for x holds in[x,S] implies in[x,U]) by 1;
      14: subset[S,U] implies (for z holds in[z,S] implies in[z,U]) by 13;
      15: subset[S,U] by 4;
      16: for z holds in[z,S] implies in[z,U] by 14,15;
      17: in[x,S] implies in[x,U] by 16;
      18: in[x,U] by 17,7;
      thus false by 12,18;
    qed;
    thus empty[S] by 6,7;
  qed;
  thus (for T holds subset[S,T]) implies empty[S] by 4;
  assume 5: empty[S];
  let V be Set;
  6: subset[S,V] iff (for x holds in[x,S] implies in[x,V]) by 1;
  7: (for x holds in[x,S] implies in[x,V]) implies subset[S,V] by 6;
  8: for x holds in[x,S] implies in[x,V]
  proof
    let x be Element;
    assume 8: in[x,S];
    9: in[x,V]
    proof
      assume 9: not in[x,V];
      10: empty[S] iff (for x holds not in[x,S]) by 2;
      11: empty[S] implies (for y holds not in[y,S]) by 10;
      12: for y holds not in[y,S] by 11,5;
      13: not in[x,S] by 12;
      thus false by 13,8;
    qed;
    thus in[x,V] by 9;
  qed;
  thus subset[S,V] by 7,8;
qed;
```

Более короткое “человеческое” доказательство, которое успешно верифицируется алгоритмом очевидности $\mathcal{E}_1$, имеет следующий вид:

```
  for S holds (for T holds subset[S,T]) iff empty[S]
proof
  let S be Set;
  thus (for T holds subset[S,T]) implies empty[S]
  proof
    assume 4: for T holds subset[S,T];
    7: for x holds not in[x,S]
    proof
      let x be Element such that 7: in[x,S];
      consider U be Set such that 8: empty[U] by 3;
      11: for y holds not in[y,U] by 2,8;
      12: not in[x,U] by 11;
      14: subset[S,U] implies (for z holds in[z,S] implies in[z,U]) by 1;
      15: subset[S,U] by 4;
      thus contradiction by 12,14,15,7;
    qed;
    thus empty[S] by 2,7;
  qed;
  assume 5: empty[S];
  let V be Set;
  8: for x holds in[x,S] implies in[x,V]
  proof
    let x be Element such that 8: in[x,S];
    thus in[x,V]
    proof
      assume 9: not in[x,V];
      11: empty[S] implies (for y holds not in[y,S]) by 2;
      thus contradiction by 11,5,8;
    qed;
  qed;
  thus subset[S,V] by 1,8;
qed;
```

В этом доказательстве рассмотрим более подробно непосредственное обоснование "$\perp$ by 11,5,8". Алгоритм очевидности $\mathcal{E}_1$ сформирует дерево поиска типа И/ИЛИ, в котором содержится соответствующее допустимой подстановке $\theta = \begin{pmatrix} Y \\ x \end{pmatrix}$ следующее деривационное поддерево:



Здесь три листа этого дерева примитивизируются подстановками $\theta_1 = \varepsilon$, $\theta_2 = \begin{pmatrix} Y \\ x \end{pmatrix}$ и $\theta_3 = \varepsilon$. Комбинация совместимых подстановок θ_1, θ_2 и θ_3 дает допустимую подстановку θ для корня дерева поиска.

2. Теорема о корректности

Теорема о корректности устанавливает допустимость в $\mathcal{D}_i$ верифицируемых в продукционной системе $\mathcal{V}_i$ правил вывода "Φ by $\alpha_1, \ldots, \alpha_m$". Тем самым мы получаем метод, который позволяет правильный с точки зрения алгоритма очевидности $\mathcal{E}_i$ текст доказательства "раздробить" до уровня применений правил вывода системы натуральной дедукции $\mathcal{D}_i$.

Теорема о корректности. *Если задача* $\langle \alpha_1, \Phi_1 \rangle \ldots \langle \alpha_m, \Phi_m \rangle \rhd \Psi$ *разрешима в* $\mathcal{V}_i$*, то существует цепочка непосредственных обоснований, которая устанавливает выводимость формулы* Ψ *из формул* $\Phi_1, \ldots, \Phi_m$ *в* $\mathcal{D}_i$.

Доказательство. Пусть $\theta = \begin{pmatrix} X_1 & \ldots & X_r \\ t_1 & \ldots & t_r \end{pmatrix}$ есть подстановка из набора допустимых подстановок задачи $P = (\langle \alpha_1, \Phi_1 \rangle \ldots \langle \alpha_m, \Phi_m \rangle \rhd \Psi)$. Мы предполагаем, что подстановка θ является основной. В противном случае мы можем заменить "нерешенные" метапеременные на локальные константы. Пусть Θ является соответствующим этой подстановке

θ деривационным деревом. Будем обозначать штрихом следующее соответствие: $\Phi' = \Phi\theta$. Индукцией по высоте дерева Θ покажем, как в рамках $\mathcal{D}_i$ построить цепочку непосредственных обоснований, которая устанавливает выводимость формулы Ψ' из формул $\Phi'_1, \ldots, \Phi'_m$ в $\mathcal{D}_i$. Рассмотрим все возможные варианты получения в рамках $\mathcal{V}_i$ дерева Θ с корнем P.

1. Пусть P примитивизируется подстановкой θ. Тогда либо $\Psi = \top$, либо $\Psi' = \Phi'_i$, либо $\Phi_i = \bot$. В первых двух случаях для всех $i = 1, 2, 3$, а в третьем случае для $i = 2$, считается, что в $\mathcal{D}_i$ формула Ψ' является непосредственным следствием формул $\Phi'_1, \ldots, \Phi'_m$.

2. Пусть $P = (\Gamma \rhd \Psi_1 \,\&\, \Psi_2)$ и получается по правилу $(\rhd\&)$. Пусть дочерние задачи $(\Gamma \rhd \Psi_1)$ и $(\Gamma \rhd \Psi_2)$ являются корнями соответственно поддеревьев Θ_1 и Θ_2. Пусть по индуктивному предположению деривационным поддеревьям Θ_1 и Θ_2 сопоставляются соответственно цепочки непосредственных обоснований

$$\Delta_1\,(\beta_1 : \Psi'_1 \text{ by } \gamma_1, \ldots, \gamma_k) \;\text{ и }\; \Delta_2\,(\beta_2 : \Psi'_2 \text{ by } \delta_1, \ldots, \delta_l).$$

Указанные цепочки устанавливают соответственно выводимость Ψ'_1 и Ψ'_2 из $\Phi'_1, \ldots, \Phi'_m$ в $\mathcal{D}_i$. Тогда дереву Θ мы можем сопоставить цепочку непосредственных обоснований

$$\begin{array}{l}\Delta_1\,(\beta_1 : \Psi'_1 \text{ by } \gamma_1, \ldots, \gamma_k)\\ \Delta_2\,(\beta_2 : \Psi'_2 \text{ by } \delta_1, \ldots, \delta_l)\\ (\Psi'_1 \,\&\, \Psi'_2 \text{ by } \beta_1, \beta_2).\end{array}$$

Полученная цепочка устанавливает выводимость Ψ' из $\Phi'_1, \ldots, \Phi'_m$ в $\mathcal{D}_i$.

3. Пусть $P = (\Gamma \rhd \Psi_1 \vee \Psi_2)$ получается по правилу $(\rhd\vee)$. Пусть для $j = 1$ или 2 по индуктивному предположению деривационному поддереву Θ_j сопоставляется цепочка непосредственных обоснований

$$\Delta\,(\beta : \Psi'_j \text{ by } \gamma_1, \ldots, \gamma_k).$$

Тогда дереву Θ сопоставим цепочку непосредственных обоснований

$$\Delta\,(\beta : \Psi'_j \text{ by } \gamma_1, \ldots, \gamma_k)\,(\Psi'_1 \vee \Psi'_2 \text{ by } \beta).$$

4. Пусть $P = (\Gamma \rhd \exists x\, \Psi_1(x))$ получается по правилу $(\rhd\exists)$. Тогда имеем $\Psi_1(X) = \Psi_1(X_j)$ и $\Psi_1(X)\theta = \Psi'_1(t_j)$ для некоторого $j \in \{1, \ldots, r\}$. Пусть по индуктивному предположению соответствующему задаче $(\Gamma \rhd \Psi_1(X))$ деривационному поддереву в Θ сопоставляется цепочка непосредственных обоснований

$$\Delta\,(\beta : \Psi'_1(t_j) \text{ by } \gamma_1, \ldots, \gamma_k).$$

В таком случае дереву Θ мы можем сопоставить цепочку непосредственных обоснований

$$\Delta\,(\beta : \Psi'_1(t_j)\ \text{by}\ \gamma_1, \ldots, \gamma_k)\,(\exists x\,\Psi'_1(x)\ \text{by}\ \beta).$$

5. Пусть $P = (\langle\alpha_i,\ \Upsilon_1 \vee \Upsilon_2\rangle\,\Pi \rhd \Psi)$ получается по правилу $(\vee\rhd)$. Пусть дочерние задачи $(\Pi \rhd \Upsilon_1 \supset \Psi)$ и $(\Pi \rhd \Upsilon_2 \supset \Psi)$ являются корнями соответственно поддеревьев Θ_1 и Θ_2. Пусть по индуктивному предположению деривационным поддеревьям Θ_1 и Θ_2 сопоставляются соответственно цепочки непосредственных обоснований

$$\Delta_1\,(\Upsilon'_1 \supset \Psi'\ \text{by}\ \gamma_1, \ldots, \gamma_k)\ \ \text{и}\ \ \Delta_2\,(\Upsilon'_2 \supset \Psi'\ \text{by}\ \delta_1, \ldots, \delta_l).$$

Тогда дереву Θ сопоставим цепочку непосредственных обоснований

$$\begin{array}{l}(\alpha_i : \Upsilon'_1 \vee \Upsilon'_2)\\ \Delta_1\,(\beta_1 : \Upsilon'_1 \supset \Psi'\ \text{by}\ \gamma_1, \ldots, \gamma_k)\\ \Delta_2\,(\beta_2 : \Upsilon'_2 \supset \Psi'\ \text{by}\ \delta_1, \ldots, \delta_l)\\ (\Psi'\ \text{by}\ \alpha_i,\ \beta_1,\ \beta_2).\end{array}$$

6. Пусть $P = (\langle\alpha_i,\ \Upsilon_1\,\&\,\Upsilon_2\rangle\,\Pi \rhd \Psi)$ получается по правилу $(\&\rhd)$. Пусть для $j = 1$ или 2 по индуктивному предположению деривационному поддереву Θ_j сопоставляется цепочка непосредственных обоснований

$$(\beta : \Upsilon'_j)\,\Delta\,(\Psi'\ \text{by}\ \gamma_1, \ldots, \gamma_k).$$

Тогда дереву Θ сопоставим цепочку непосредственных обоснований

$$(\alpha_i : \Upsilon'_1\,\&\,\Upsilon'_2)\,(\beta : \Upsilon'_j\ \text{by}\ \alpha_i)\,\Delta\,(\Psi'\ \text{by}\ \gamma_1, \ldots, \gamma_k).$$

7. Пусть $P = (\langle\alpha_i,\ \Upsilon_1 \supset \Upsilon_2\rangle\,\Pi \rhd \Psi)$ получается по правилу $(\supset \rhd)$. Пусть дочерние задачи $(\Pi \rhd \Upsilon_1)$ и $(\langle\alpha_i,\ \Upsilon_2\rangle\,\Pi \rhd \Psi)$ являются корнями соответственно поддеревьев Θ_1 и Θ_2. Пусть по индуктивному предположению деривационным поддеревьям Θ_1 и Θ_2 сопоставляются соответственно цепочки непосредственных обоснований

$$\Delta_1\,(\Upsilon'_1\ \text{by}\ \gamma_1, \ldots, \gamma_k),$$

$$(\beta_2 : \Upsilon'_2)\,\Delta_2\,(\Psi'\ \text{by}\ \delta_1, \ldots, \delta_l).$$

Тогда дереву Θ сопоставим цепочку непосредственных обоснований

$$\begin{array}{l}(\alpha_i : \Upsilon'_1 \supset \Upsilon'_2)\\ \Delta_1\,(\beta_1 : \Upsilon'_1\ \text{by}\ \gamma_1, \ldots, \gamma_k)\\ (\beta_2 : \Upsilon'_2\ \text{by}\ \alpha_i,\ \beta_1)\,\Delta_2\\ (\Psi'\ \text{by}\ \delta_1, \ldots, \delta_l).\end{array}$$

8. Пусть $P = (\langle \alpha_i,\, \forall x\, \Upsilon(x) \rangle\, \Pi \rhd \Psi)$ получается по правилу $(\forall\rhd)$. Тогда имеем $\Upsilon(X) = \Upsilon(X_j)$ и $\Upsilon(X)\theta = \Upsilon'(t_j)$ для некоторого $j \in \{1, \ldots, r\}$. Пусть по индуктивному предположению деривационному поддереву в Θ, соответствующему задаче $(\langle \beta,\, \Upsilon(X) \rangle\, \Pi \rhd \Psi)$, сопоставляется цепочка непосредственных обоснований

$$(\beta : \Upsilon'(t_j))\, \Delta\, (\Psi' \text{ by } \gamma_1, \ldots, \gamma_k).$$

В этом случае дереву Θ мы можем сопоставить цепочку непосредственных обоснований

$$(\alpha_i : \forall x\, \Upsilon'(x))\, (\beta : \Upsilon'(t_j) \text{ by } \alpha_i)\, \Delta\, (\Psi' \text{ by } \gamma_1, \ldots, \gamma_k).$$

□

Следствие. *Если алгоритм очевидности $\mathcal{E}_i$ распознал, что "Δ есть обоснование формулы Ψ с посылками $\Phi_1, \ldots, \Phi_m$", то существует натуральный вывод Δ' в $\mathcal{D}_i$ формулы Ψ из посылок $\Phi_1, \ldots, \Phi_m$.*

Доказательство. Пусть алгоритм очевидности $\mathcal{E}_i$ распознал, что "Δ есть обоснование формулы Ψ с посылками $\Phi_1, \ldots, \Phi_m$". Пусть Δ' получается из Δ в результате замены всех непосредственных обоснований вида "Φ by $\alpha_1, \ldots, \alpha_n$" на цепочки непосредственных обоснований, о которых говорится в теореме о корректности $\mathcal{V}_i$. Тогда тривиальной индукцией по числу подвыводов мы получаем, что Δ' есть натуральный вывод в $\mathcal{D}_i$ формулы Ψ из посылок $\Phi_1, \ldots, \Phi_m$. Что и требовалось доказать. □

3. Теорема о полноте

Формулируемая в этом разделе теорема о полноте обеспечивает верифицируемость в рамках продукционной системы $\mathcal{V}_i$ всех правильных непосредственных умозаключений, получаемых в системе натуральной дедукции $\mathcal{D}_i$. Это влечет полноту алгоритма очевидности $\mathcal{E}_i$ в отношении $\mathcal{D}_i$.

Теорема о полноте. *Если формула Ψ является непосредственным следствием формул $\Phi_1, \ldots, \Phi_m$ в системе натуральной дедукции $\mathcal{D}_i$, то задача $\langle \alpha_1, \Phi_1 \rangle \ldots \langle \alpha_m, \Phi_m \rangle \rhd \Psi$ разрешима в $\mathcal{V}_i$.*

Доказательство. Пусть формула Ψ получает обоснование с помощью одного из прямых правил вывода из формул $\Phi_1, \ldots, \Phi_m$:

$$(\alpha_1 : \Phi_1) \ldots (\alpha_m : \Phi_m)\, \Delta\, (\Psi \text{ by } \alpha_1, \ldots, \alpha_m).$$

Рассмотрим все возможные варианты непосредственных обоснований в $\mathcal{D}_i$ формулы Ψ из формул $\Phi_1, \ldots, \Phi_m$.

1. Пусть формула Ψ является аксиомой $\top$. Тогда задача

$$\langle \alpha_1,\, \Phi_1 \rangle \ldots \langle \alpha_m,\, \Phi_m \rangle \rhd \Psi$$

является примитивной. Следовательно, она разрешима в $\mathcal{V}_i$.

2. Пусть формула $\Psi = (\Phi_1 \,\&\, \Phi_2)$ получает обоснование с помощью правила (&в) из формул Φ_1 и Φ_2:

$$(\alpha_1 : \Phi_1)\,(\alpha_2 : \Phi_2)\,\Delta\,(\Phi_1 \,\&\, \Phi_2 \text{ by } \alpha_1,\, \alpha_2).$$

Тогда задача $(\langle \alpha_1,\, \Phi_1 \rangle \langle \alpha_2,\, \Phi_2 \rangle \Gamma \rhd \Phi_1 \,\&\, \Phi_2)$ разрешима в $\mathcal{V}_i$. Действительно, в этом случае поисковое дерево содержит соответствующее допустимой подстановке ε возникающее в результате применения правила декомпозиции ($\rhd\&$) следующее поддерево решения:

$$\begin{array}{ccc} \langle \alpha_1,\, \Phi_1 \rangle \langle \alpha_2,\, \Phi_2 \rangle \Gamma \rhd \Phi_1 & & \langle \alpha_1,\, \Phi_1 \rangle \langle \alpha_2,\, \Phi_2 \rangle \Gamma \rhd \Phi_2 \\ & \nwarrow \!\frown\! \nearrow & \\ & \langle \alpha_1,\, \Phi_1 \rangle \langle \alpha_2,\, \Phi_2 \rangle \Gamma \rhd \Phi_1 \,\&\, \Phi_2 & \end{array}$$

3. Пусть формула $\Psi = (\Upsilon_1 \vee \Upsilon_2)$ является непосредственным следствием формулы Υ_j по правилу вывода ($\vee$в):

$$(\alpha_1 : \Upsilon_j)\,\Delta\,(\Upsilon_1 \vee \Upsilon_2 \text{ by } \alpha_1).$$

Тогда задача $(\langle \alpha_1,\, \Upsilon_j \rangle \Gamma \rhd \Upsilon_1 \vee \Upsilon_2)$ разрешима в $\mathcal{V}_i$. Действительно, в этом случае поисковое дерево содержит соответствующее допустимой подстановке ε деривационное поддерево, которое получается в результате применения правила декомпозиции ($\rhd\vee$):

$$\begin{array}{c} \langle \alpha_1,\, \Upsilon_j \rangle \Gamma \rhd \Upsilon_j \\ \uparrow \\ \langle \alpha_1,\, \Upsilon_j \rangle \Gamma \rhd \Upsilon_1 \vee \Upsilon_2 \end{array}$$

4. Пусть формула $\Psi = (\exists x\, \Upsilon(x))$ получает обоснование с помощью правила ($\exists$в) из формулы $\Upsilon(t)$:

$$(\alpha_1 : \Upsilon(t))\,\Delta\,(\exists x\, \Upsilon(x) \text{ by } \alpha_1).$$

Тогда задача $(\langle \alpha_1,\, \Upsilon(t) \rangle \Gamma \rhd \exists x\, \Upsilon(x))$ разрешима в $\mathcal{V}_i$. Действительно, в этом случае поисковое дерево содержит соответствующее допустимой

подстановке $\theta = \begin{pmatrix} X \\ t \end{pmatrix}$ возникающее в результате применения правила декомпозиции ($\triangleright\exists$) следующее деривационное поддерево:

$$\begin{array}{c} \langle \alpha_1,\, \Upsilon(t) \rangle\, \Gamma \;\triangleright\; \Upsilon(X) \\ \uparrow \\ \langle \alpha_1,\, \Upsilon(t) \rangle\, \Gamma \;\triangleright\; \exists x\, \Upsilon(x) \end{array}$$

5. Пусть формула $\Psi = \Upsilon_j$ является непосредственным следствием формулы $\Upsilon_1 \,\&\, \Upsilon_2$ по правилу вывода (&и):

$$(\alpha_1 : \Upsilon_1 \,\&\, \Upsilon_2)\, \Delta\, (\Upsilon_j \text{ by } \alpha_1).$$

Тогда задача ($\langle \alpha_1,\, \Upsilon_1 \,\&\, \Upsilon_2 \rangle\, \Gamma \triangleright \Upsilon_j$) разрешима в $\mathcal{V}_i$. В этом случае поисковое дерево содержит соответствующее допустимой подстановке ε поддерево решения, которое получается в результате применения правила декомпозиции (&$\triangleright$):

$$\begin{array}{c} \langle \alpha_1,\, \Upsilon_j \rangle\, \Gamma \;\triangleright\; \Upsilon_j \\ \uparrow \\ \langle \alpha_1,\, \Upsilon_1 \,\&\, \Upsilon_2 \rangle\, \Gamma \;\triangleright\; \Upsilon_j \end{array}$$

6. Пусть формула $\Psi = \Upsilon(t)$ получает обоснование с помощью правила ($\forall$и) из формулы $\forall x\, \Upsilon(x)$:

$$(\alpha_1 : \forall x\, \Upsilon(x))\, \Delta\, (\Upsilon(t) \text{ by } \alpha_1).$$

Тогда задача ($\langle \alpha_1,\, \forall x\, \Upsilon(x) \rangle\, \Gamma \;\triangleright\; \Upsilon(t)$) разрешима в $\mathcal{V}_i$. Действительно, в этом случае поисковое дерево содержит соответствующее допустимой подстановке $\theta = \begin{pmatrix} X \\ t \end{pmatrix}$ возникающее в результате применения правила декомпозиции ($\forall\triangleright$) следующее поддерево решения:

$$\begin{array}{c} \langle \alpha_1,\, \Upsilon(X) \rangle\, \Gamma \;\triangleright\; \Upsilon(t) \\ \uparrow \\ \langle \alpha_1,\, \forall x\, \Upsilon(x) \rangle\, \Gamma \;\triangleright\; \Upsilon(t) \end{array}$$

7. Пусть формула Ψ получает обоснование с помощью правила ($\supset$и) из формул $\Upsilon \supset \Psi$ и Υ:

$$(\alpha_1 : \Upsilon \supset \Psi)\, (\alpha_2 : \Upsilon)\, \Delta\, (\Psi \text{ by } \alpha_1,\, \alpha_2).$$

Тогда задача $(\langle \alpha_1,\, \Upsilon \supset \Psi\rangle\, \langle \alpha_2,\, \Upsilon\rangle\, \Gamma \rhd \Psi)$ разрешима в $\mathcal{V}_i$. В этом случае поисковое дерево содержит соответствующее допустимой подстановке ε поддерево решения, которое получается в результате применения правила декомпозиции $(\supset \rhd)$:

$$\begin{array}{c} \langle \alpha_2,\, \Upsilon\rangle\, \Gamma \rhd \Upsilon \qquad \langle \alpha_1,\, \Psi\rangle\, \langle \alpha_2,\, \Upsilon\rangle\, \Gamma \rhd \Psi \\ \nwarrow\!\!\smile\!\!\nearrow \\ \langle \alpha_1,\, \Upsilon \supset \Psi\rangle\, \langle \alpha_2,\, \Upsilon\rangle\, \Gamma \rhd \Psi \end{array}$$

8. Пусть формула Ψ получает обоснование с помощью правила $(\vee\text{и})$ из формул $\Upsilon_1 \vee \Upsilon_2$, $\Upsilon_1 \supset \Psi$ и $\Upsilon_2 \supset \Psi$:

$$(\alpha_1 : \Upsilon_1 \vee \Upsilon_2)\, (\alpha_2 : \Upsilon_1 \supset \Psi)\, (\alpha_3 : \Upsilon_1 \supset \Psi)\, \Delta\, (\Psi \text{ by } \alpha_1,\, \alpha_2,\, \alpha_3).$$

Тогда задача $(\langle \alpha_1,\, \Upsilon_1 \vee \Upsilon_2\rangle\, \langle \alpha_2,\, \Upsilon \supset \Psi\rangle\, \langle \alpha_3,\, \Upsilon \supset \Psi\rangle\, \Gamma \rhd \Psi)$ разрешима в $\mathcal{V}_i$. Действительно, в этом случае поисковое дерево содержит соответствующее допустимой подстановке ε возникающее в результате применения правила декомпозиции $(\supset \rhd)$ следующее поддерево решения:

$$\begin{array}{c} \begin{array}{l}\langle \alpha_2,\, \Upsilon_1 \supset \Psi\rangle \\ \langle \alpha_3,\, \Upsilon_2 \supset \Psi\rangle\end{array}\ \Gamma \rhd \Upsilon_1 \supset \Psi \qquad \begin{array}{l}\langle \alpha_2,\, \Upsilon_1 \supset \Psi\rangle \\ \langle \alpha_3,\, \Upsilon_2 \supset \Psi\rangle\end{array}\ \Gamma \rhd \Upsilon_2 \supset \Psi \\ \nwarrow\!\!\smile\!\!\nearrow \\ \langle \alpha_1,\, \Upsilon_1 \vee \Upsilon_2\rangle\, \langle \alpha_2,\, \Upsilon_1 \supset \Psi\rangle\, \langle \alpha_3,\, \Upsilon_2 \supset \Psi\rangle\, \Gamma \rhd \Psi \end{array}$$

9. Пусть формула Ψ является непосредственным следствием формулы $\bot$ в $\mathcal{D}_2$ по правилу вывода $(\bot\text{и})$:

$$(\alpha_1 : \bot)\, \Delta\, (\Psi \text{ by } \alpha_1).$$

Тогда задача $(\langle \alpha_1,\, \bot\rangle\, \Gamma \rhd \Psi)$ является примитивной в $\mathcal{V}_2$, а, следовательно, эта задача разрешима в $\mathcal{V}_2$. □

Следствие. *Если Δ есть формальное доказательство формулы Ψ из посылок $\Phi_1,\, \ldots,\, \Phi_m$ в $\mathcal{D}_i$, то алгоритм очевидности $\mathcal{E}_i$ установит, что "Δ есть обоснование формулы Ψ с посылками $\Phi_1,\, \ldots,\, \Phi_m$".*

Доказательство. Пусть Δ есть формальный вывод формулы Ψ из посылок $\Phi_1,\, \ldots,\, \Phi_m$ в системе натуральной дедукции $\mathcal{D}_i$. По теореме о полноте $\mathcal{V}_i$ тогда алгоритм очевидности $\mathcal{E}_i$ успешно верифицирует в Δ все непосредственные обоснования вида "Φ by $\alpha_1,\, \ldots,\, \alpha_n$". При этом $\mathcal{E}_i$ также успешно верифицирует в Δ все обоснования вида "Φ proof $\ldots$ qed". Это значит, что с точки зрения $\mathcal{E}_i$ текст Δ является обоснованием формулы Ψ из посылок $\Phi_1,\, \ldots,\, \Phi_m$. Что и требовалось доказать. □

Автор выражает искреннюю признательность О.А. Охотникову за помощь и поддержку при подготовке данной статьи.

Список литературы

[1] Robinson A., Voronkov A., eds, *Handbook of Automated Reasoning*, Elsevier and MIT Press, 2001, 2122 pp.

[2] Wiedijk F., "The QED Manifesto Revisited", *Studies in Logic, Grammar and Rhetoric*, **10**:23 (2007), 121–133

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

[4] Охотников О.А., "О поиске натурального классического логического вывода с использованием частичной скулемизации", *Интеллектуальные системы*, **23**:4 (2019), 39–90

[5] Глушков В.М., *Машина доказывает*, «Знание», Москва, 1981, 64 с.

[6] Grabowski A., Kornilowicz A., Naumowicz A., "Mizar in a Nutshell", *Journal of Formalized Reasoning*, **3**:2 (2010), 153–245

[7] Verchinine K., Lyaletski A., Paskevich A., "System for Automated Deduction (SAD): A Tool for Proof Verification", *CADE-21. Lecture Notes in Computer Science*, **4603** (2007), 398–403

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

About verification of formalized mathematical proofs

Vtorushin Yu.I.

An algorithm for verification of formalized mathematical proofs is considered. Its main part is described in the framework of some production system with metavariables. Theorems of soundness and completeness are proved.

Keywords: automated theorem proving, system for automated deduction, first order language, predicate calculus, production system, artificial intelligence.

Верификация функциональных программ методом построения диаграмм состояний

Миронов А.М.[1]

В статье вводятся графовые объекты, соответствующие функциональным программам, называемые диаграммами состояний. Показано, как можно использовать диаграммы состояний для решения задач верификации функциональных программ. Предлагаемый подход иллюстрируется примером верификации функциональной программы сортировки.

Ключевые слова: верификация программ, функциональные программы, диаграммы состояний.

1. Введение

Проблема **верификации программ**, которая заключается в доказательстве утверждений о том что анализируемые программы обладают заданными свойствами, является одной из основных проблем теоретической информатики. Для различных классов программ используются различные методы верификации. Например, для верификации последовательных программ используются метод индуктивных утверждений Флойда [2], логика Хоара [3], и т.д. Для верификации параллельных и распределенных программ используются методы, основанные на исчислении взаимодействующих систем (CCS) и π-исчислении Милнера [4], [5], теория взаимодействующих последовательных процессов Хоара (CSP) и ее обобщения [6], [7], темпоральная логика и model checking [8], процессная алгебра [9], сети Петри [10], и т.д.

[1] *Миронов Андрей Михайлович* — к.ф.-м.н., доцент каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ имени М.В.Ломоносова, e-mail: amironov66@gmail.com.

Mironov Andrey Mihailovich — Candidate of Physical and Mathematical Sciences, Associate Professor, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intelligent Systems.

Основными методами верификации функциональных программ (ФП) являются вычислительная индукция и структурная индукция [1]. Главные проблемы при использовании этих методов связаны с высокой сложностью построения формальных доказательств корректности анализируемых ФП. Существуют также другие методы верификации ФП, отметим среди них следующие: методы, основанные на анализе потоков управления [11], методы, основанные на рассуждениях с типами данных, абстрактной интерпретации и выводе типов [12], методы, основанные на использовании подхода model checking [13], [14], методы, основанные на понятии мультипараметрического преобразователя деревьев [15].

В настоящей работе мы рассматриваем ФП как системы алгебраических уравнений над строками. Мы вводим понятие диаграммы состояний для таких ФП, и излагаем метод верификации, основанный на диаграммах состояний. Основные преимущества излагаемого подхода по сравнению со всеми вышеперечисленными подходами к верификации ФП связаны с тем, что данный подход позволяет представить доказательства корректности ФП в виде простых свойств их диаграмм состояний.

Основная идея предлагаемого подхода заключается в следующем:

- предполагается, что спецификация свойств анализируемой ФП Σ определяется другой ФП Σ', входными значениями для которой являются выходные значения ФП Σ,
- ФП Σ считается корректной относительно спецификации, выражаемой ФП Σ', если и только если суперпозиция $f_{\Sigma'}(f_{\Sigma})$ функций, соответствующих ФП Σ и Σ', принимает значение 1 на всех своих аргументах, данное утверждение обозначается записью

$$f_{\Sigma'}(f_{\Sigma}) = 1 \tag{1}$$

- обоснование утверждения (1) сводится к анализу диаграммы состояний для ФП $\Sigma'(\Sigma)$, которой соответствует суперпозиция $f_{\Sigma'}(f_{\Sigma})$.

Предложенный метод верификации ФП иллюстрируется примером верификации ФП сортировки вставкой. Сначала мы представляем полное доказательство корректности этой ФП на основе структурной индукции. Это делается для сравнения сложности верификации ФП на основе метода структурной индукции, и сложности предлагаемого метода верификации ФП на основе диаграмм состояний. Далее, мы представляем доказательство корректности ФП методом, основанным на построении

диаграмм состояний. Доказательство вторым методом может быть порождено автоматически. Это демонстрирует преимущества предлагаемого метода верификации ФП по сравнению с верификацией на основе метода структурной индукции.

2. Основные понятия

2.1. Термы

Мы предполагаем, что задано множество $\mathcal{D}$ **значений**, и каждое значение из $\mathcal{D}$ имеет один из трех типов: **C**, **S** или **B**. Значения типа **C** называются **символами**, значения типа **S** называются **символьными строками** (или просто **строками**), значения типа **B** называются **булевыми значениями**. Существует два булевых значения: $\top$ (истина) и $\bot$ (ложь). Каждое значение типа **S** представляет собой конечную (возможно пустую) последовательность значений типа **C**. Совокупности значений типа **C**, **S** и **B** обозначаются записями $\mathcal{D}_{\mathbf{C}}$, $\mathcal{D}_{\mathbf{S}}$ и $\mathcal{D}_{\mathbf{B}}$ соответственно.

Также предполагаем, что заданы множества

- $\mathcal{X}$ **переменных по данным** (или просто **переменных**),
- $\mathcal{C}$ **констант**,
- $\mathcal{F}$ **функциональных символов (ФС)**, и
- Φ **функциональных переменных**.

Каждый элемент x какого-либо из вышеперечисленных множеств связан с **типом**, обозначаемым знакосочетанием $\tau(x)$, и

- если $x \in \mathcal{X}$ или $x \in \mathcal{C}$, то $\tau(x) \in \{\mathbf{C}, \mathbf{S}, \mathbf{B}\}$,
- если $x \in \mathcal{F}$ или $x \in \Phi$, то $\tau(x)$ – это запись вида $(t_1, \ldots, t_n) \to t$, где $t_1, \ldots, t_n, t \in \{\mathbf{C}, \mathbf{S}, \mathbf{B}\}$.

Каждой константе $c \in \mathcal{C}$ соответствует элемент множества $\mathcal{D}_{\tau(c)}$, называемый **значением** этой константы. Символ ε обозначает константу типа **S**, значением которой является пустая строка. Существуют константы типа **B**, которым соответствуют значения $\top$ и $\bot$, данные константы обозначаются теми же символами $\top$ и $\bot$ соответственно.

Каждому ФС $f \in \mathcal{F}$ соответствует частичная функция, обозначаемая тем же символом f, и имеющая вид

$$f : \mathcal{D}_{t_1} \times \ldots \times \mathcal{D}_{t_n} \to \mathcal{D}_t, \quad \text{где } \tau(f) = (t_1, \ldots, t_n) \to t.$$

Ниже мы перечисляем некоторые ФС из $\mathcal{F}$, рядом с каждым ФС мы указываем (через двоеточие) его тип.

1) $head : \mathbf{S} \to \mathbf{C}$. Функция $head$ определена на непустых строках, она отображает каждую непустую строку в ее первый элемент (т.е. если строка u имеет вид $a_1 \dots a_n$, то $head(u) = a_1$).

2) $tail : \mathbf{S} \to \mathbf{S}$. Функция $tail$ определена на непустых строках, она отображает каждую непустую строку u в строку (называемую **хвостом** строки u), получаемую из u удалением ее первого элемента.

3) $conc : (\mathbf{C}, \mathbf{S}) \to \mathbf{S}$. Для каждой пары $(a, u) \in \mathcal{D}_{\mathbf{C}} \times \mathcal{D}_{\mathbf{S}}$ строка $conc(a, u)$ получается путем приписывания символа a в начало строки u.

4) $=: (t, t) \to \mathbf{B}$, где $t \in \{\mathbf{C}, \mathbf{S}, \mathbf{B}\}$, т.е. символ $=$ обозначает три ФС. Значение функции $=$ на паре (x, y) равно $\top$, если x и y совпадают, и $\bot$, иначе.

5) $\leq: (\mathbf{C}, \mathbf{C}) \to \mathbf{B}$. Мы предполагаем, что $\mathcal{D}_{\mathbf{C}}$ – линейно упорядоченное множество, и значение функции $\leq$ на паре (a, b) равно $\top$, если $a \leq b$, и $\bot$, иначе.

6) Булевы ФС: $\neg : \mathbf{B} \to \mathbf{B}$, $\wedge : (\mathbf{B}, \mathbf{B}) \to \mathbf{B}$, и т.д., соответствующие функции являются стандартными булевыми функциями на аргументах $\top$ и $\bot$ (т.е. $\neg(\top) = \bot$, и т.д.).

7) $if\_then\_else : (\mathbf{B}, t, t) \to t$, где $t \in \{\mathbf{C}, \mathbf{S}, \mathbf{B}\}$, т.е. запись if_then_else обозначает три ФС. Функции, соответствующие этим ФС, определяются одинаково:

$$if\_then\_else\ (a, x, y) \stackrel{\text{def}}{=} \begin{cases} x, & \text{если } a = \top, \\ y, & \text{если } a = \bot. \end{cases}$$

Понятие **терма** определяется индуктивно. Каждый терм e связан с некоторым типом $\tau(e) \in \{\mathbf{C}, \mathbf{S}, \mathbf{B}\}$. Определение терма имеет следующий вид:

- каждая переменная по данным и каждая константа является термом, тип которого равен типу этой переменной или константы,

- если f – ФС или функциональная переменная, $e_1, \ldots, e_n$ термы, и

$$\tau(f) = (\tau(e_1), \ldots, \tau(e_n)) \to t,$$

  то запись $f(e_1, \ldots, e_n)$ – терм типа t.

Будем использовать следующие понятия и обозначения.

- Множество всех термов будем обозначать символом $\mathcal{E}$.
- $\forall\, e, e' \in \mathcal{E}$ e' называется **подтермом** терма e, если либо $e' = e$, либо e имеет вид $f(e_1, \ldots, e_n)$, и $\exists\, i \in \{1, \ldots, n\} : e'$ – подтерм терма e_i.
- $\forall\, e \in \mathcal{E}$ записи X_e, Φ_e обозначают множества всех переменных по данным и функциональных переменных соответственно, входящих в e.
- $\forall\, X \subseteq \mathcal{X}$ запись $\mathcal{E}_X$ обозначает множество $\{e \in \mathcal{E} \mid X_e \subseteq X\}$.
- Будем обозначать термы

$$head(e),\ tail(e),\ conc(e, e'), = (e, e'),\ \leq (e, e'),\ if\_then\_else\ (e, e', e'')$$

  записями e_h, e_t, ee', $e = e'$, $e \leq e'$, $[\![e]\!]\, e' : e''$, соответственно.
- Терм называется **простым**, если он имеет вид $e_1 \ldots e_n$, где $\forall\, i = 1, \ldots, n$ e_i – переменная или константа.
- Термы, содержащие булевы ФС, будут обозначаться так же как в математических текстах (т.е. в виде записей $e \wedge e'$, и т.д.), термы вида $e_1 \wedge \ldots \wedge e_n$ могут также обозначаться записями вида $\left\{ \begin{array}{c} e_1 \\ \vdots \\ e_n \end{array} \right\}$,
- $\forall\, e \in \mathcal{E}$ запись $\mathcal{D}_e$ обозначает множество $\mathcal{D}_{\tau(e)}$.
- Списки термов обозначаются записями вида $\bar{e}$.
- Если $\bar{e}$ – список термов вида $(e_1, \ldots, e_n)$, то
  - $\tau(\bar{e})$ обозначает список $(\tau(e_1), \ldots, \tau(e_n))$,
  - $X_{\bar{e}}$, $\Phi_{\bar{e}}$ обозначают множества $\bigcup_{i=1}^{n} X_{e_i}$, $\bigcup_{i=1}^{n} \Phi_{e_i}$ соответственно,
  - $\mathcal{D}_{\bar{e}}$ обозначает множество $\mathcal{D}_{e_1} \times \ldots \times \mathcal{D}_{e_n}$.

- Если $\bar{e}' = (e'_1, \ldots, e'_n)$, $\bar{e}'' = (e''_1, \ldots, e''_n)$ – списки термов, $\tau(\bar{e}') = \tau(\bar{e}'')$, то
  - запись $\bar{e}' = \bar{e}''$ обозначает терм $(e'_1 = e''_1) \wedge \ldots \wedge (e'_n = e''_n)$,
  - если дополнительно предполагается, что для каждой пары i, j различных индексов из $\{1, \ldots, n\}$ терм e'_i не является подтермом терма e''_j, то
    * $\forall\, e \in \mathcal{E}$ запись
    $$e[e''_1/e'_1, \ldots, e''_n/e'_n] \tag{2}$$
    обозначает терм, получаемый из e заменой $\forall\, i = 1, \ldots, n$ каждого подтерма терма e, совпадающего с e'_i, на терм e''_i, терм (2) обозначается также записью $e[\bar{e}''/\bar{e}']$,
    * для каждого списка термов $\bar{e} = (e_1, \ldots, e_m)$ запись $\bar{e}[\bar{e}''/\bar{e}']$ обозначает терм
    $$(e_1[\bar{e}''/\bar{e}'], \ldots, e_m[\bar{e}''/\bar{e}']).$$

- **Уточнением** называется запись θ вида
$$e_1/x_1, \ldots, e_n/x_n, \tag{3}$$
где $x_1, \ldots, x_n$ – различные переменные, $e_1, \ldots, e_n$ – простые термы, причем $\forall\, i = 1, \ldots, n \;\; \tau(x_i) = \tau(e_i)$. $\forall\, e \in \mathcal{E}$ запись $e[\theta]$ обозначает терм $e[e_1/x_1, \ldots, e_n/x_n]$ (аналогичные записи используются, когда вместо терма e рассматривается список термов). (3) называется **переименованием**, если $e_1, \ldots, e_n$ – различные переменные.

2.2. Понятие функциональной программы

В настоящем тексте под **функциональной программой (ФП)** понимается конечная совокупность Σ равенств вида

$$\begin{cases} \varphi_1(x_{11}, \ldots, x_{1n_1}) = e_1 \\ \ldots \\ \varphi_m(x_{m1}, \ldots, x_{mn_m}) = e_m \end{cases} \tag{4}$$

где $\varphi_1, \ldots, \varphi_m$ – различные функциональных переменные, и для каждого $i = 1, \ldots, m \;\; \varphi_i(x_{i1}, \ldots, x_{in_i})$ и e_i – термы одинакового типа, причем

$$X_{e_i} = \{x_{i1}, \ldots, x_{in_i}\}, \quad \Phi_{e_i} \subseteq \{\varphi_1, \ldots, \varphi_m\}.$$

Главным термом ФП (4) называется левая часть первого равенства в (4) (т.е. терм $\varphi_1(x_{11}, \ldots, x_{1n_1})$).

Совокупность равенств, входящих в ФП (4), можно рассматривать как систему функциональных уравнений относительно функциональных переменных $\varphi_1, \ldots, \varphi_m$. Данная система определяет список

$$(f_{\varphi_1}, \ldots, f_{\varphi_m}) \tag{5}$$

частичных функций, соответствующих функциональным переменным $\varphi_1, \ldots, \varphi_m$, который является наименьшим (в смысле порядка на списках частичных функций, описанного в [1]) решением системы функциональных уравнений (4). Список (5) называется **наименьшей неподвижной точкой (ННТ)** ФП (4). Все детали, связанные с понятием ННТ ФП, м.б. найдены в книге [1]. Первая фунция в списке (5) (т.е. f_{φ_1}) обозначается записью f_Σ, и называется **функцией, определяемой ФП** Σ.

Пусть задана ФП Σ. Запись $\mathcal{E}_\Sigma$ обозначает множество всех термов, таких, что все входящие в них функциональные переменные входят в Σ.

Будем считать ФП Σ и Σ' одинаковыми, если Σ' получается из Σ переименованием переменных по данным и функциональных переменных, т.е. если $X\Phi_\Sigma$ и $X\Phi_{\Sigma'}$ – множества всех переменных по данным и функциональных переменных, входящих в Σ и Σ' соответственно, то существует взаимно однозначное соответствие $f : X\Phi_\Sigma \to X\Phi_{\Sigma'}$, такое, что Σ' получается из Σ заменой каждой переменной $v \in X\Phi_\Sigma$ на $f(v)$.

3. Пример спецификации и верификации функциональной программы

3.1. Пример функциональной программы

Рассмотрим следующую ФП:

$$\left\{\begin{array}{l} \mathbf{sort}(x) = [\![x = \varepsilon]\!]\ \varepsilon : \mathbf{insert}(x_h, \mathbf{sort}(x_t)) \\ \mathbf{insert}(a, y) = [\![y = \varepsilon]\!] \quad a\varepsilon \\ \qquad\qquad\qquad\qquad : \quad [\![a \le y_h]\!] \quad ay \\ \qquad\qquad\qquad\qquad\qquad\quad : y_h\ \mathbf{insert}(a, y_t) \end{array}\right. \tag{6}$$

Эта ФП определяет функцию сортировки на строках. ФП состоит из двух уравнений, которые определяют следующие функции:

- $\mathbf{sort} : \mathbf{S} \to \mathbf{S}$ – основная функция, и

- **insert** : $(\mathbf{C}, \mathbf{S}) \to \mathbf{S}$ – вспомогательная функция, отображающая пару $(a, y) \in \mathcal{D}_{\mathbf{C}} \times \mathcal{D}_{\mathbf{S}}$ в строку, получаемую вставкой символа a в строку y, причем выполнено следующее условие: если строка y упорядочена, то строка $\mathbf{insert}(a, y)$ тоже упорядочена (строка упорядочена, если ее компоненты образуют неубывающую последовательность).

3.2. Пример спецификации функциональной программы

Одно из свойств корректности ФП (6) имеет следующий вид: $\forall\, x \in \mathcal{D}_{\mathbf{S}}$ строка $\mathbf{sort}(x)$ упорядочена. Это свойство м.б. выражено формально. Рассмотрим ФП, определяющую функцию **ord** проверки упорядоченности строк:

$$\begin{aligned} \mathbf{ord}(x) = [\![x = \varepsilon]\!] \quad & 1 \\ : [\![x_t = \varepsilon]\!] \quad & 1 \\ : [\![x_h \le (x_t)_h]\!] \quad & \mathbf{ord}(x_t) : 0 \end{aligned} \tag{7}$$

Функция **ord** позволяет выразить описанное выше свойство корректности в виде следующего математического утверждения:

$$\forall\, x \in \mathcal{D}_{\mathbf{S}} \quad \mathbf{ord}(\mathbf{sort}(x)) = 1. \tag{8}$$

3.3. Пример верификации функциональной программы

Проблема верификации свойства корректности (8) ФП (6) заключается в построении формального доказательства утверждения (8). Это утверждение может быть доказано как обычная математическая теорема, путем использования метода математической индукции. Например, доказательство этого утверждения может иметь следующий вид.

Если $x = \varepsilon$, то, согласно первому уравнению системы (6), равенство $\mathbf{sort}(x) = \varepsilon$ верно, и поэтому

$$\mathbf{ord}(\mathbf{sort}(x)) = \mathbf{ord}(\varepsilon) = 1.$$

Пусть $x \neq \varepsilon$. Докажем (8) для этого случая по индукции.

Предположим что для каждой строки y, длина которой меньше, чем длина строки x, равенство $\mathbf{ord}(\mathbf{sort}(y)) = 1$ верно. Докажем, что тогда

$$\mathbf{ord}(\mathbf{sort}(x)) = 1. \tag{9}$$

(9) равносильно равенству

$$\mathbf{ord}(\ \mathbf{insert}(x_h, \mathbf{sort}(x_t))) = 1. \tag{10}$$

По индуктивному предположению, $\mathbf{ord}(\mathbf{sort}(x_t)) = 1$, откуда следует (10) на основании следующей леммы.

Лемма.

Имеет место импликация:

$$\mathbf{ord}(y) = 1 \quad \Rightarrow \quad \mathbf{ord}(\mathbf{insert}(a, y)) = 1 \tag{11}$$

Доказательство.

Индукция по структуре y.

Если $y = \varepsilon$, то правая часть (11) имеет вид $\mathbf{ord}(a\varepsilon) = 1$, что верно по определению **ord**.

Пусть $y \neq \varepsilon$, и для каждой строки z, длина которой меньше чем длина y, верна импликация:

$$\mathbf{ord}(z) = 1 \quad \Rightarrow \quad \mathbf{ord}(\mathbf{insert}(a, z)) = 1 \tag{12}$$

Пусть $c \stackrel{\text{def}}{=} y_h$, $d \stackrel{\text{def}}{=} y_t$. Тогда (11) имеет вид

$$\mathbf{ord}(cd) = 1 \quad \Rightarrow \quad \mathbf{ord}(\mathbf{insert}(a, cd)) = 1 \tag{13}$$

Для доказательства импликации (13) необходимо доказать, что если $\mathbf{ord}(cd) = 1$, то верны следующие импликации:

(a) $a \leq c \quad \Rightarrow \quad \mathbf{ord}(a(cd)) = 1$,

(b) $c < a \quad \Rightarrow \quad \mathbf{ord}(c\ \mathbf{insert}(a, d)) = 1$.

(a) верно, т.к. из $a \leq c$ следует, что $\mathbf{ord}(a(cd)) = \mathbf{ord}(cd) = 1$.

Докажем (b).

- $d = \varepsilon$. В этом случае правая часть (b) имеет вид

  $$\mathbf{ord}(c(a\varepsilon)) = 1 \tag{14}$$

  (14) следует из неравенства $c < a$.

- $d \neq \varepsilon$. Пусть $p \stackrel{\text{def}}{=} d_h$, $q \stackrel{\text{def}}{=} d_t$.

  В этом случае необходимо доказать, что если $c < a$, то

  $$\mathbf{ord}(c\ \mathbf{insert}(a, pq)) = 1 \tag{15}$$

1) если $a \leq p$, то (15) имеет вид

$$\mathbf{ord}(c(a(pq))) = 1 \tag{16}$$

Т.к. $c < a \leq p$, то (16) следует из равенств

$$\mathbf{ord}(c(a(pq))) = \mathbf{ord}(a(pq)) = \mathbf{ord}(pq) =$$
$$= \mathbf{ord}(c(pq)) = \mathbf{ord}(cd) = 1$$

2) если $p < a$, то (15) имеет вид

$$\mathbf{ord}(c(p\ \mathbf{insert}(a, q))) = 1 \tag{17}$$

Т.к., по предположению

$$\mathbf{ord}(cd) = \mathbf{ord}(c(pq)) = 1$$

то $c \leq p$, и поэтому (17) можно переписать в виде

$$\mathbf{ord}(p\ \mathbf{insert}(a, q)) = 1 \tag{18}$$

Если $p < a$, то

$$\mathbf{insert}(a, d) = \mathbf{insert}(a, pq) = p\ \mathbf{insert}(a, q)$$

поэтому (18) можно переписать в виде

$$\mathbf{ord}(\mathbf{insert}(a, d)) = 1 \tag{19}$$

(19) следует из индуктивного предположения для леммы (т.е. из импликации (12), где $z \stackrel{\text{def}}{=} d$) из равенства

$$\mathbf{ord}(d) = 1$$

которое обосновывается цепочкой равенств

$$1 = \mathbf{ord}(cd) = \mathbf{ord}(c(pq)) = \quad \text{(т.к. } c \leq p\text{)}$$
$$= \mathbf{ord}(pq) = \mathbf{ord}(d). \quad \blacksquare$$

Из вышеприведенного примера мы видим, что даже для простейшей ФП, которая состоит из нескольких строк, доказательство ее корректности является нетривиальным математическим рассуждением, которое трудно проверить и гораздо более трудно построить.

Ниже мы представляем принципиально другой метод верификации ФП, основанный на построении диаграммы состояний ФП, и иллюстрируем его доказательством соотношения (8) на основе этого метода. Это доказательство может быть порождено автоматически, что является свидетельством преимуществ метода верификации ФП на основе диаграмм состояний.

4. Состояния функциональных программ

4.1. Понятие состояния функциональной программы

Состоянием ФП Σ называется запись s вида $\{\beta\}^{y}_{x_1,\ldots,x_n}$ где

- $\beta \in \mathcal{E}_\Sigma$ – терм типа $\mathbf{B}$, называемый **условием** состояния s,
- y – простой терм, называемый **выходным термом** состояния s, и
- $x_1, \ldots, x_n$ – список простых термов, называемых **входными термами** состояния s.

Множество всех состояний ФП Σ обозначается записью $\mathcal{S}_\Sigma$. $\forall\, s \in \mathcal{S}_\Sigma$ запись X_s обозначает множество всех переменных по данным, входящих в состояние s. Если состояние s имеет вид $\{\beta\}^{y}_{x_1,\ldots,x_n}$, то будем обозначать записями β_s, y_s, и $\bar{x}_s$ термы β, y и список $x_1, \ldots, x_n$, соответственно. Если β_s имеет вид $e_1 \wedge \ldots \wedge e_n$, то s может обозначаться записью $\left\{ \begin{array}{c} e_1 \\ \ldots \\ e_n \end{array} \right\}^{y_s}_{\bar{x}_s}$. Если $\beta_s = \top$, то s может обозначаться записью $\{\}^{y_s}_{\bar{x}_s}$).

Состояние $s \in \mathcal{S}_\Sigma$ называется **начальным** состоянием ФП Σ (и обозначается записью s^0_Σ), если оно имеет вид $\{y = \varphi(\bar{x})\}^{y}_{\bar{x}}$ где

- φ – функциональная переменная, входящая в главный терм ФП Σ,
- $\bar{x}$ – список различных переменных,
- y – переменная, не входящая в $\bar{x}$, и
- $\tau(\varphi) = \tau(\bar{x}) \to \tau(y)$.

Состояние $s \in \mathcal{S}_\Sigma$ называется **терминальным**, если β_s не содержит функциональных переменных.

Если $s \in \mathcal{S}_\Sigma$ и θ – уточнение, то $s[\theta]$ обозначает состояние $\{\beta_s[\theta]\}^{y_s[\theta]}_{\bar{x}_s[\theta]}$. Если θ – переименование, то будем говорить, что $s[\theta]$ получается из s переименованием.

4.2. Равенство термов и состояний

Пусть задано множество переменных $X \subseteq \mathcal{X}$. **Означиванием** переменных из X называется функция $\xi : X \to \mathcal{D}$, сопоставляющая каждой переменной $x \in X$ значение $\xi(x)$ типа $\tau(x)$. Множество всех означиваний переменных из X обозначается записью $X^\bullet$. Для

- каждого означивания $\xi \in X^\bullet$, и
- каждого терма e, такого, что $X_e \subseteq X$ и $\Phi_e = \emptyset$,

запись e^ξ обозначает объект, который либо является значением из $\mathcal{D}$, либо не определен, и вычисляется рекурсивно:

- если $e = x \in X$, то $e^\xi = \xi(x)$,
- если $e = c \in \mathcal{C}$, то e^ξ – значение константы c,
- если $e = f(e_1, \ldots, e_n)$, где $f \in \mathcal{F}$, то
  - e^ξ равно значению $f(e_1^\xi, \ldots, e_n^\xi)$, если это значение определено,
  - e^ξ не определено, в противном случае.

Пусть задана ФП Σ. Для каждого $e \in \mathcal{E}_\Sigma$, и каждого $\xi \in X^\bullet$, где $X \supseteq X_e$, запись $e^{\xi,\Sigma}$ обозначает **значение** e **на** ξ **относительно** Σ, которое определяется так же как выше, со следующим отличием: каждая функциональная переменная из Φ_e рассматривается как ФС, с которым связана частичная функция, являющаяся соответствующей компонентой ННТ ФП Σ.

Термы e_1 и $e_2 \in \mathcal{E}_\Sigma$ считаются **равными** (относительно ФП Σ), если для каждого означивания $\xi \in (X_{e_1} \cup X_{e_2})^\bullet$ объекты $e_1^{\xi,\Sigma}$ и $e_2^{\xi,\Sigma}$ либо оба не определены, либо оба определены и совпадают.

Примеры пар равных термов:

$$e_1 e_1' = e_2 e_2' \text{ и } (e_1 = e_2) \wedge (e_1' = e_2'), \quad e e' = \varepsilon \text{ и } \bot,$$
$$[\![\top]\!] e : e' \text{ и } e, \quad [\![\bot]\!] e : e' \text{ и } e', \quad e = \top \text{ и } e, \quad e = \bot \text{ и } \neg e,$$
$$e \wedge (e' = e'') \text{ и } e[e''/e'] \wedge (e' = e'').$$

Пусть Σ – ФП, и s, s' – пара состояний из $\mathcal{S}_\Sigma$. Будем рассматривать s и s' как равные, если верно какое-либо из следующих условий:

- s' получается из s переименованием, и $\bar{x}_s = \bar{x}_{s'}$,
- $\beta_{s'} = \beta \wedge (x = e)$, где $x \in \mathcal{X}$, $x \notin X_s$, $e \in \mathcal{E}$, β получается из β_s заменой некоторых вхождений e на x, $y_{s'} = y_s$, $\bar{x}_{s'} = \bar{x}_s$,
- $\beta_s = \beta \wedge (x = e)$, где $x \in \mathcal{X}$, e – простой терм, $x \notin X_e$, $s' = s[e/x]$.

4.3. Переходы функциональных программ

В этом пункте определяется понятие перехода ФП. Каждый такой переход выражает связь между двумя состояниями ФП, и имеет метку, которая представляет собой

- либо одну из функциональных переменных,
- либо терм типа **B**, называемый **условием** этого перехода.

Переходом ФП Σ называется тройка $r = (s, s', l)$, где s, s' – состояния из $\mathcal{S}_\Sigma$, называемые **началом** и **концом** перехода r, соответственно, и l – запись, называемая **меткой** перехода r. Переход (s, s', l) называется переходом из s в s'. Он может обозначаться также записью $s \xrightarrow{l} s'$.

$\forall\, s \in \mathcal{S}_\Sigma$ есть следующие переходы с началом s: пусть $s = \{\beta\}^{y}_{\bar{x}}$, тогда

1) если β содержит подтерм вида $\varphi(\bar{e})$, и одно из уравнений в Σ имеет вид $\varphi(\bar{x}) = e$, то имеется переход (называемый **раскрытием**)

$$s \xrightarrow{\varphi} \{\beta[e[\bar{e}/\bar{x}]/\varphi(\bar{e})]\}^{y}_{\bar{x}},$$

2) если β содержит подтерм e типа **B**, то имеется пара переходов

$$s \xrightarrow{e} \{\beta \wedge e\}^{y}_{\bar{x}}, \quad s \xrightarrow{\neg e} \{\beta \wedge \neg e\}^{y}_{\bar{x}}, \tag{20}$$

3) если β содержит подтерм e типа **S**, то имеется пара переходов

$$s \xrightarrow{e=\varepsilon} \{\beta \wedge (e = \varepsilon)\}^{y}_{\bar{x}}, \quad s \xrightarrow{e=xx'} \{\beta \wedge (e = xx')\}^{y}_{\bar{x}}, \tag{21}$$

где x, x' **новые** переменные (т.е. не входящие в X_s).

Каждый из переходов, входящий в какую-либо из пар вида (20) или (21), называется **комплементарным** к другому переходу из этой пары.

Множество всех переходов ФП Σ обозначается записью $\mathcal{R}_\Sigma$.

5. Окрестности состояний

5.1. Развертки состояний

Пусть задана ФП Σ.

Разверткой состояния $s \in \mathcal{S}_\Sigma$ называется конечное дерево V,

- каждой вершине v которого сопоставлено состояние $s_v \in \mathcal{S}_\Sigma$, причем корню этого дерева сопоставлено состояние s, и
- каждому ребру r которого сопоставлен переход из $\mathcal{R}_\Sigma$ вида $s_v \xrightarrow{l} s_{v'}$, где v и v' – начало и конец ребра r, метка l этого перехода является также меткой ребра r.

Вершины и ребра V будут отождествляться с теми состояниями и переходами соответственно, которые им сопоставлены.

5.2. Понятие окрестности состояния

Пусть задана ФП Σ.

Каждому состоянию $s \in \mathcal{S}_\Sigma$ соответствует множество $\mathcal{U}_s$ всех **окрестностей** состояния s. Каждая окрестность $U \in \mathcal{U}_s$ является деревом,

- вершинам которого сопоставлены состояния из $\mathcal{S}_\Sigma$, и
- ребра имеют метки, представляющие собой списки меток, используемых в развёртках состояний.

Множество $\mathcal{U}_s$ определяется следующим образом.

1) Каждая развертка V состояния s, такая, что $\forall\, v \in V$ из вершины v
   - либо не выходит ни одного ребра (такая v называется **листом**),
   - либо выходит одно ребро, являющееся раскрытием,
   - либо выходит пара комплементарных ребер,

   принадлежит множеству $\mathcal{U}_s$.

2) Пусть $U \in \mathcal{U}_s$, и s' – вершина U, не являющаяся ни корнем, ни листом, тогда, если
   - ребро с концом в s', имеет вид $s_0 \xrightarrow{l} s'$, и

- ребра с началом s', имеют вид $s' \stackrel{l_1}{\to} s_1$, …, $s' \stackrel{l_n}{\to} s_n$,

то $\mathcal{U}_s$ содержит дерево U', получаемое из U

- удалением вершины s' и связанных с ней ребер, и
- добавлением ребер $s_0 \stackrel{ll_1}{\to} s_1, \ldots, s_0 \stackrel{ll_n}{\to} s_n$, где $\forall\, i = 1, \ldots, n \;\; ll_i$ – конкатенация списков l и l_i.

3) Пусть $U \in \mathcal{U}_s$, и s' – **противоречивая** вершина U (т.е. $\beta_{s'} = \bot$), не являющаяся корнем, тогда $\mathcal{U}_s$ содержит дерево U', получаемое из U удалением вершин, достижимых из s' (т.е. таких, в которые существует путь из s'), а также ребер, связанных с этими вершинами.

Окрестность U', получаемую из U согласно пунктам 2 и 3 данного определения, будем называть **редукцией** окрестности U.

Нетрудно доказать, что

- вершина s какой-либо окрестности противоречива тогда и только тогда, когда концы всех ребер, выходящих из s, противоречивы, и
- состояние противоречиво тогда и только тогда, когда все листья некоторой его окрестности противоречивы.

Если U – окрестность какого-либо состояния, то $\forall\, v \in U$ существует единственный путь из корня U в вершину v. Все вершины U, лежащие на этом пути и не совпадающие с v, будем называть **предками** v.

При графическом изображении окрестностей мы будем использовать следующее соглашение: если U – окрестность какого-либо состояния, то в графическом изображении окрестности U

- вершины, входящие в U, изображаются овалами, причем корневая вершина изображается двойным овалом, противоречивые вершины могут изображаться черными квадратами (■),
- $\forall\, s \in U$ овал O_s, изображающий s, выглядит следующим образом:
  - конъюнктивные члены, входящие в β_s, изображаются в столбик внутри O_s (а если $\beta_s = \top$, то внутри O_s ничего не рисуется),
  - список $\bar{x}_s$ входных термов и выходной терм y_s состояния s изображаются справа от O_s снизу и сверху соответственно, и

– идентификатор состояния s изображается слева сверху от O_s,

- ребра, входящие в U изображаются стрелками, соединяющими овалы: если U содержит ребро $s \xrightarrow{l} s'$, то
  - данное ребро изображается стрелкой из O_s в $O_{s'}$, и
  - рядом с этой стрелкой м.б. указаны компоненты метки l.

5.3. Примеры окрестностей

В этом пункте мы приводим примеры окрестностей состояний для ФП сортировки (6) и для ФП проверки упорядоченности строк (7).

5.3.1. Примеры окрестностей для программы сортировки

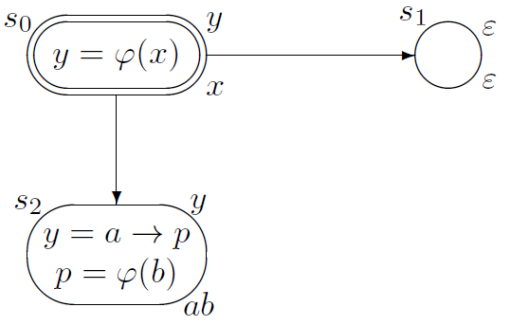
Перепишем ФП сортировки (6), используя более короткие обозначения для входящих в нее функциональных переменных (будем обозначать термы вида $\mathbf{sort}(x)$ и $\mathbf{insert}(a, y)$ записями $\varphi(x)$ и $a \to y$ соответственно):

$$\begin{cases} \varphi(x) = [\![x = \varepsilon]\!]\, \varepsilon : (x_h \to \varphi(x_t)) \\ a \to y = [\![y = \varepsilon]\!] a\varepsilon : \Big([\![a \le y_h]\!]\, ay : y_h(a \to y_t) \Big) \end{cases} \tag{22}$$

Одна из разверток состояния $s_0 \stackrel{\text{def}}{=} \{y = \varphi(x)\}^y_x$ состоит из следующих состояний и ребер:

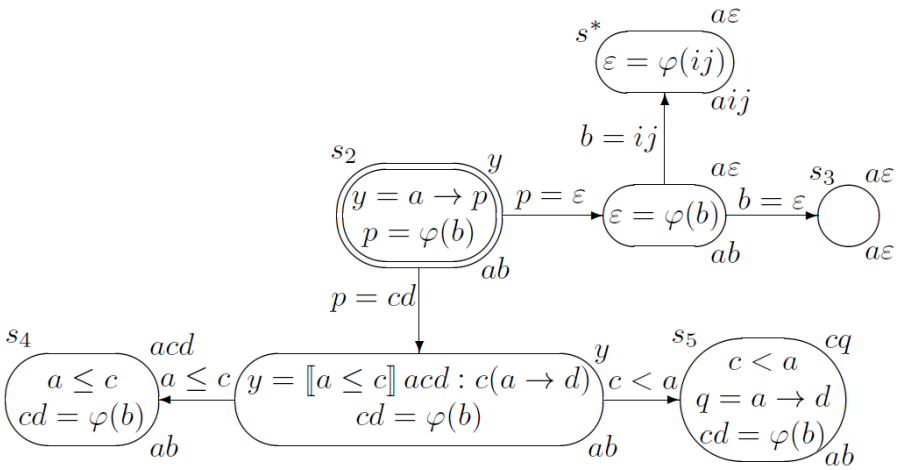
$$\begin{aligned} s_0 &\xrightarrow{\varphi} s \stackrel{\text{def}}{=} \{y = [\![x = \varepsilon]\!]\, \varepsilon : x_h \to \varphi(x_t)\}^y_x, \\ s &\xrightarrow{x=\varepsilon} \left\{ \begin{array}{l} y = [\![x = \varepsilon]\!]\, \varepsilon : x_h \to \varphi(x_t) \\ x = \varepsilon \end{array} \right\}^y_x = \{y = \varepsilon\}^y_\varepsilon = \{\}^\varepsilon_\varepsilon, \\ s &\xrightarrow{x=ab} \left\{ \begin{array}{l} y = [\![x = \varepsilon]\!]\, \varepsilon : x_h \to \varphi(x_t) \\ x = ab \end{array} \right\}^y_x = \\ &= \{y = a \to \varphi(b)\}^y_{ab} = \left\{ \begin{array}{l} y = a \to p \\ p = \varphi(b) \end{array} \right\}^y_{ab}. \end{aligned}$$

Одна из окрестностей, соответствующих данной развертке, имеет вид



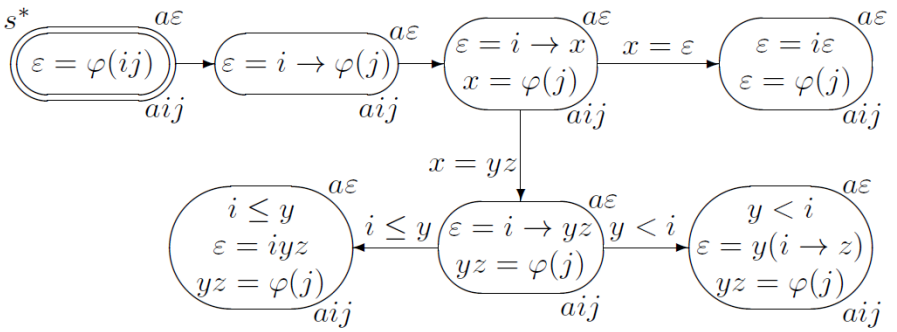
(23)

Одна из окрестностей состояния s_2 в (23) имеет вид



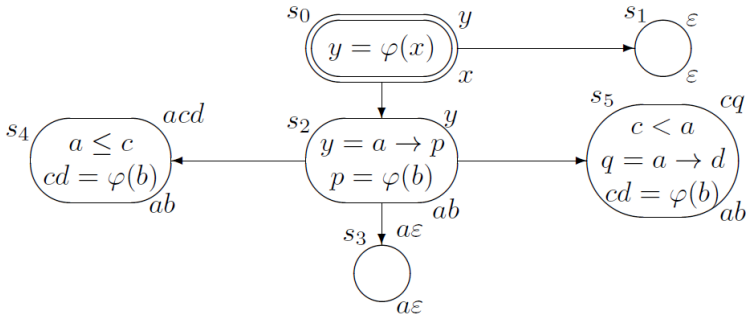
(24)

Одна из окрестностей состояния s^* в (24) имеет вид



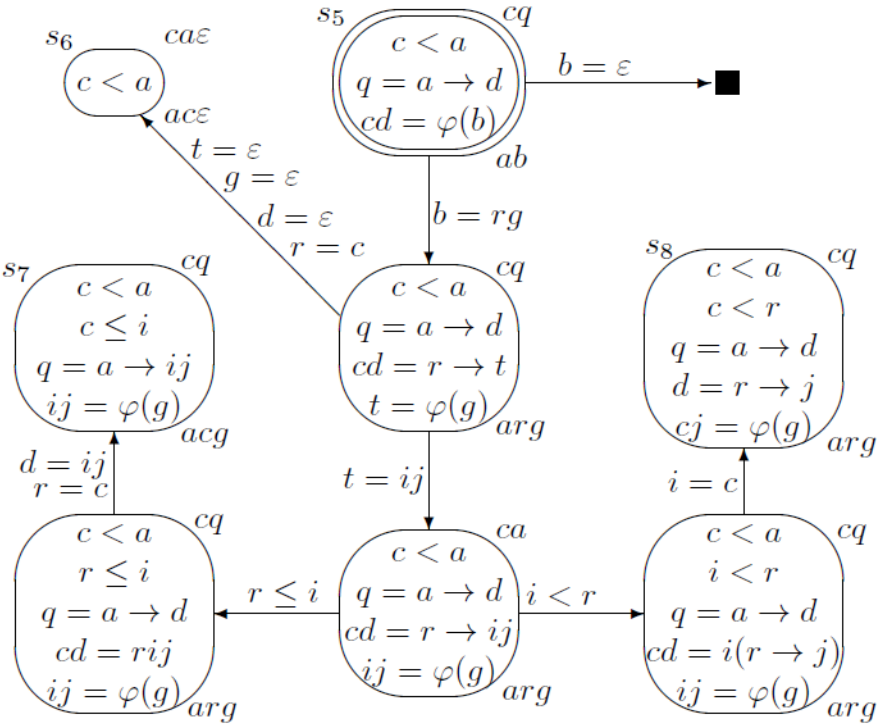
Все листья последней окрестности противоречивы, т.к. среди конъюнктивных членов, входящих в их условия, содержатся равенства вида $\varepsilon = uv$, которые равны терму $\bot$. Следовательно, по сказанному выше, противоречивой является и s^*.

Объединяя окрестности (23) и (24), с учетом вышесказанного заключаем, что одна из окрестностей состояния $s_0 \stackrel{\text{def}}{=} \{y = \varphi(x)\}_x^y$ имеет вид

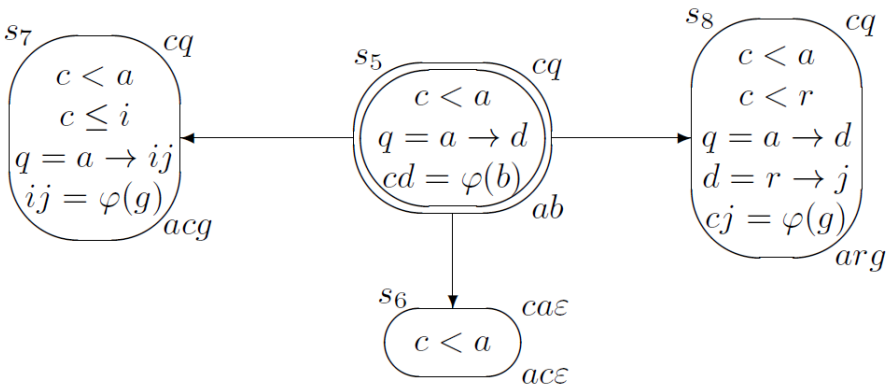


(25)

Другой пример окрестности связан с состоянием s_5. Одна из окрестностей этого состояния имеет вид



Данную окрестность можно редуцировать, и получить окрестность



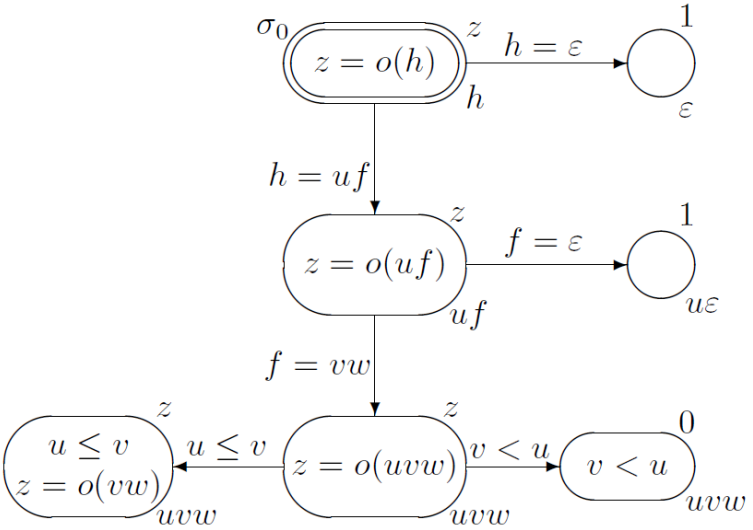
(26)

5.3.2. Примеры окрестностей для программы проверки упорядоченности строк

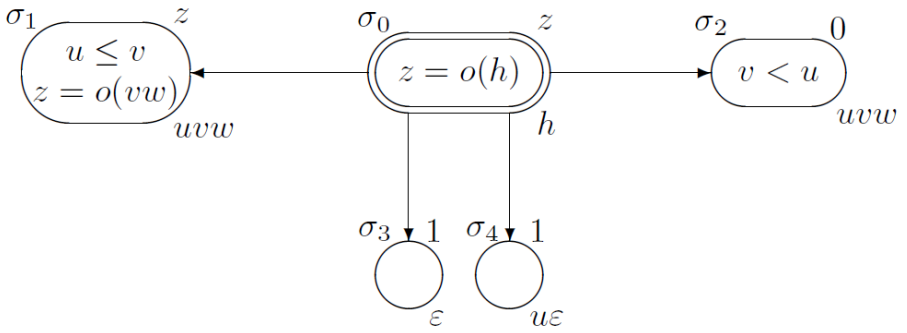
Другие примеры окрестностей состояний связаны с ФП (7) проверки упорядоченности строк. Перепишем эту ФП, используя более короткое обозначение для входящей в нее функциональной переменной:

$$o(x) = [\![x = \varepsilon]\!]1 : \Big([\![x_t = \varepsilon]\!]1 : [\![x_h \le (x_t)_h]\!]o(x_t) : 0\Big) \tag{27}$$

Одна из окрестностей состояния $\sigma_0 \stackrel{\text{def}}{=} \{z = o(h)\}^z_h$ этой ФП имеет вид



Используя определение понятия окрестности состояния ФП, данную окрестность можно преобразовать в окрестность



(28)

6. Вложения состояний

6.1. Явные, условные и обоснованные вложения состояний

Пусть задана пара состояний $s, s' \in \mathcal{S}_\Sigma$.

- **Явное вложение** s в s' – это запись вида

$$\theta : s \hookrightarrow s',$$

где θ – уточнение, и $\beta_s = \beta_{s'}[\theta] \wedge \beta$, причем $\Phi_\beta = \emptyset$.

- **Условное вложение** s в s' – это запись вида

$$\left\{ \begin{array}{l} \eta : r \hookrightarrow r' \\ u[\theta] \hookrightarrow u'[\theta'] \end{array} \right\} : s \hookrightarrow s', \tag{29}$$

где $\eta : r \hookrightarrow r'$ – явное вложение, u, $u' \in \mathcal{S}_\Sigma$, θ и θ' – уточнения, и

$$\beta_s = \beta_{u[\theta]} \wedge \beta_r, \quad \beta_{s'} = \beta_{u'[\theta']} \wedge \beta_{r'}.$$

Посылкой условного вложения (29) называется запись $u \hookrightarrow u'$, где u и u' – соответствующие состояния, входящие в (29).

- **Обоснованное** вложение s в s' – это запись вида

$$s \overset{!}{\hookrightarrow} s' \tag{30}$$

если $\exists U \in \mathcal{U}_s$, $\exists U' \in \mathcal{U}_{s'}$: для каждого нетерминального листа $r \in U$

– либо существует явное вложение r в некоторое $r' \in U'$,

– либо существует условное вложение r в некоторое $r' \in U'$, причем его посылка имеет вид $s \hookrightarrow s'$, где s и s' – состояния в (30).

- Будем говорить, что s **вложено** в s', если существует либо явное вложение s в s', либо условное вложение s в s' с обоснованной посылкой.

  Запись $s \subseteq s'$ обозначает, что s вложено в s'.

Отметим, что каждое обоснованное вложение можно рассматривать как условное вложение с обоснованной посылкой (компонента "явное вложение" в этом условном вложении является тривиальной).

6.2. Примеры вложений состояний

6.2.1. Примеры явных вложений

1) Для состояний $s_4 = \left\{ \begin{array}{l} a \le c \\ cd = \varphi(b) \end{array} \right\}^{acd}_{ab}$ и $s_0 = \{y = \varphi(x)\}^y_x$, входящих в окрестность (25), имеется явное вложение

$$[cd/y, b/x] : s_4 \hookrightarrow s_0. \tag{31}$$

2) Для состояний $s_7 = \left\{ \begin{array}{l} c < a,\ c \le i \\ q = a \to ij \\ ij = \varphi(g) \end{array} \right\}^{cq}_{acg}$ и $s_2 = \left\{ \begin{array}{l} y = a \to p \\ p = \varphi(b) \end{array} \right\}^{y}_{ab}$, входящих в окрестности (26) и (23) соответственно, имеется явное вложение

$$[q/y, ij/p, g/b] : s_7 \hookrightarrow s_2. \tag{32}$$

3) Для состояний $\sigma_1 = \left\{ \begin{array}{l} r \le v \\ z = o(vw) \end{array} \right\}^{z}_{rvw}$ и $\sigma_0 = \{z = o(h)\}^z_h$, входящих в окрестность (28), имеется явное вложение

$$[vw/h] : \sigma_1 \hookrightarrow \sigma_0. \tag{33}$$

6.2.2. Пример условного вложения

Для состояний $s_8 = \left\{ \begin{array}{l} c < a,\ c < r \\ q = a \to d \\ d = r \to j \\ cj = \varphi(g) \end{array} \right\}^{cq}_{arg}$ и $s_2 = \left\{ \begin{array}{l} y = a \to p \\ p = \varphi(b) \end{array} \right\}^{y}_{ab}$, входящих в окрестности (26) и (23) соответственно, имеется условное вложение

с посылкой $s_5 \hookrightarrow s_0$:

$$\left\{ \begin{array}{l} [q/y, d/p] : \left\{ \begin{array}{l} c < a \\ q = a \to d \end{array} \right\}_{acd}^{cq} \hookrightarrow \{y = a \to p\}_{ap}^{y} \\ s_5[\theta] = \left\{ \begin{array}{l} c < r \\ d = r \to j \\ cj = \varphi(g) \end{array} \right\}_{rg}^{cd} \hookrightarrow s_0[\theta'] = \{p = \varphi(b)\}_{b}^{p} \end{array} \right\} : s_8 \hookrightarrow s_2, \quad (34)$$

где $\theta = [r/a, d/q, j/d, g/b]$, $\theta' = [p/y, b/x]$.

6.2.3. Пример обоснованного вложения

Один из примеров обоснованного вложения: $s_5 \overset{!}{\hookrightarrow} s_0$, где s_5 и s_0 – состояния из (25). В данном случае $U =$ (26) и $U' =$ (23). В окрестности (26)

- состояние s_6 является терминальным,
- существует явное вложение (32) состояния s_7 в состояние s_2, и
- существует условное вложение (34) состояния s_8 в состояние s_2 с посылкой $s_5 \hookrightarrow s_0$.

7. Диаграммы состояний

7.1. Понятие диаграммы состояний

Пусть задана ФП Σ.

Диаграмма состояний (ДС) ФП Σ – это тройка

$$D = (U, N, I), \quad (35)$$

компоненты которой имеют следующий смысл:

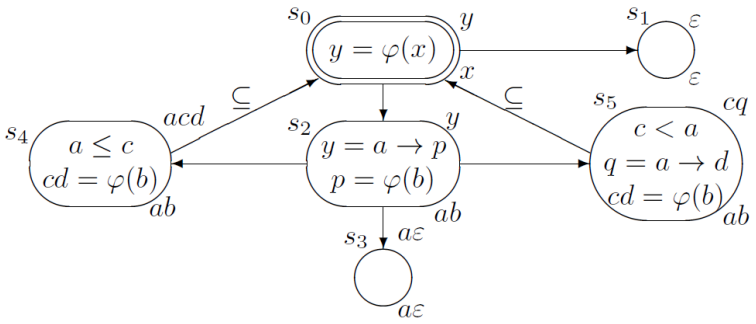
- U – некоторая окрестность начального состояния s^0_Σ,
- N – совокупность всех нетерминальных листьев окрестности U, и
- I состоит из пар вида (s, s'), где $s \in N$, s' – предок s, и $s \subseteq s'$, причем $\forall\, s \in N \;\; \exists\, s' : (s, s') \in I$.

При графическом изображении ДС (35) мы будем обозначать пары из I помеченными стрелками на окрестности U: пара $(s, s') \in I$ будет изображаться стрелкой с началом s, концом s' и меткой $\subseteq$.

7.2. Примеры диаграмм состояний

7.2.1. Диаграмма состояний для программы сортировки

ДС для ФП (22) строится на основе окрестности (25) и имеет вид

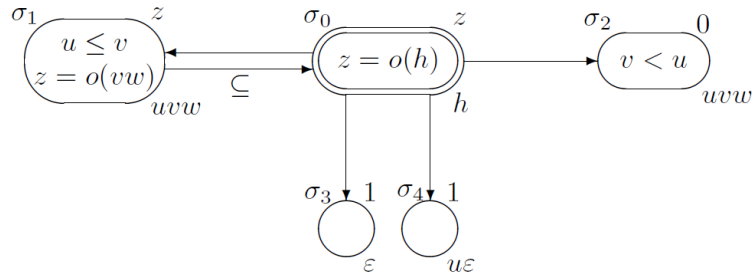


(36)

В данной ДС ребра с меткой $\subseteq$ соответствуют явному вложению (31) и обоснованному вложению $s_5 \overset{!}{\hookrightarrow} s_0$, рассмотренному в пункте 6.2.3.

7.2.2. Диаграмма состояний для программы проверки упорядоченности строк

ДС для ФП (27) строится на основе окрестности (28) и имеет вид



(37)

В данной ДС ребро с меткой $\subseteq$ соответствует явному вложению (33).

8. Верификация функциональных программ на основе понятия диаграммы состояний

8.1. Суперпозиция функциональных программ

8.1.1. Понятие суперпозиции функциональных программ

Пусть заданы ФП Σ и Σ', причем главные термы в Σ и Σ' имеют вид $\varphi(\bar{x})$ и $\varphi'(u)$ соответственно, где $\tau(\varphi(\bar{x})) = \tau(u)$, и $X\Phi_\Sigma \cap X\Phi_{\Sigma'} = \emptyset$.

В этом случае можно определить ФП $\Sigma'(\Sigma)$, которая называется **суперпозицией** ФП Σ и Σ', и представляет собой совокупность равенств,

- первое из которых имеет вид $\psi(\bar{x}) = \varphi'(\varphi(\bar{x}))$, где ψ – новая функциональная переменная соответствующего типа, и
- другие равенства – это все равенства, входящие в Σ и Σ'.

Нетрудно видеть, что $\forall\, \bar{d} \in \mathcal{D}_{\bar{x}} \quad f_{\Sigma'(\Sigma)}(\bar{d}) = f_{\Sigma'}(f_\Sigma(\bar{d}))$.

8.1.2. Окрестности начального состояния суперпозиции функциональных программ

Пусть заданы

- ФП Σ и Σ', удовлетворяющие условиям в начале пункта 8.1.1, и
- окрестности $U \in \mathcal{U}_{s^0_\Sigma}$, $U' \in \mathcal{U}_{s^0_{\Sigma'}}$, причем выполнено условие

$$\forall\, s \in U,\ \forall\, s' \in U' \quad X_s \cap X_{s'} = \emptyset.$$

$\forall\, s \in U$, $\forall\, s' \in U'$ будем обозначать записью ss' состояние ФП $\Sigma'(\Sigma)$, определяемое следующим образом: пусть $s = \{\beta\}^y_{\bar{x}}$, $s' = \{\beta'\}^z_{y'}$, тогда

$$ss' \stackrel{\text{def}}{=} \{\beta \wedge \beta' \wedge (y = y')\}^z_{\bar{x}}.$$

Нетрудно видеть, что если $s^0_\Sigma = \{y = \varphi(\bar{x})\}^y_{\bar{x}}$, $s^0_{\Sigma'} = \{z = \varphi'(y')\}^z_{y'}$, то

$$s^0_{\Sigma'(\Sigma)} = \{z = \psi(\bar{x})\}^z_{\bar{x}} = \{z = \varphi'(\varphi(\bar{x}))\}^z_{\bar{x}} = \left\{ \begin{array}{l} z = \varphi'(y) \\ y = \varphi(\bar{x}) \end{array} \right\}^z_{\bar{x}} = s^0_\Sigma s^0_{\Sigma'}.$$

Обозначим записью UU' дерево, вершины которого имеют метки вида ss', где $s \in U$, $s' \in U'$, указанием недетерминированного алгоритма его построения. Данный алгоритм состоит из нескольких этапов. Дерево, построенное на каждом из этих этапов, обозначается той же записью UU'.

- На первом этапе UU' определяется как дерево из одной вершины, которая имеет метку $s^0_\Sigma s^0_{\Sigma'}$.

- Каждый последующий этап заключается в том, что если построенное к текущему моменту дерево UU' содержит лист v с меткой ss', где либо s не является листом в U, либо s' не является листом в U' то выполняется одна из двух следующих операций:

  – если s не является листом в U, и список его последователей имеет вид $s_1, \ldots, s_n$, то к построенному дереву UU' добавляются последователи вершины v с метками s_1s', ..., s_ns',

  – если s' не является листом в U', то вместо предыдущей может выполняться аналогичная операция для последователей s'.

Теорема 1

Определенное выше дерево UU' является окрестностью начального состояния ФП $\Sigma'(\Sigma)$. ■

8.1.3. Диаграмма состояний суперпозиции функциональных программ

Теорема 2

Пусть ФП Σ и Σ' имеют ДС, и определена суперпозиция $\Sigma'(\Sigma)$.
Тогда ФП $\Sigma'(\Sigma)$ тоже имеет ДС. ■

8.2. Задача верификации функциональных программ

Задача верификации ФП Σ заключается в построении доказательства утверждения о том, что ФП Σ удовлетворяет свойству, выражаемому некоторой формальной спецификацией *Spec*.

Ниже будет использоваться следующее обозначение: запись вида $f = 1$, где f – функция, обозначает утверждение:

функция f принимает значение 1 на всех своих аргументах.

В ряде случаев

- формальная спецификация *Spec* выражается другой ФП Σ', и

- корректность Σ относительно *Spec* выражается утверждением

$$f_{\Sigma'(\Sigma)} = 1. \tag{38}$$

Например, одно из свойств корректности ФП сортировки в пункте 3.1 выражается утверждением вида (38) (а именно, утверждением (8)).

Теорема 3

Пусть ФП Σ имеет ДС, в которой для каждого терминального состояния s терм y_s является константой 1. Тогда $f_\Sigma = 1$. ■

Приведенные выше теоремы являются теоретической основой метода верификации ФП на основе построения ДС

- для анализируемой ФП Σ, и
- для ФП Σ', представляющей проверяемое свойство.

Если эти ФП имеют ДС, то, согласно теореме 2, $\Sigma'(\Sigma)$ тоже имеет ДС. Если эта ДС обладает свойством, описанным в теореме 3, то будет верно (38).

В следующем пункте излагается пример применения данного метода.

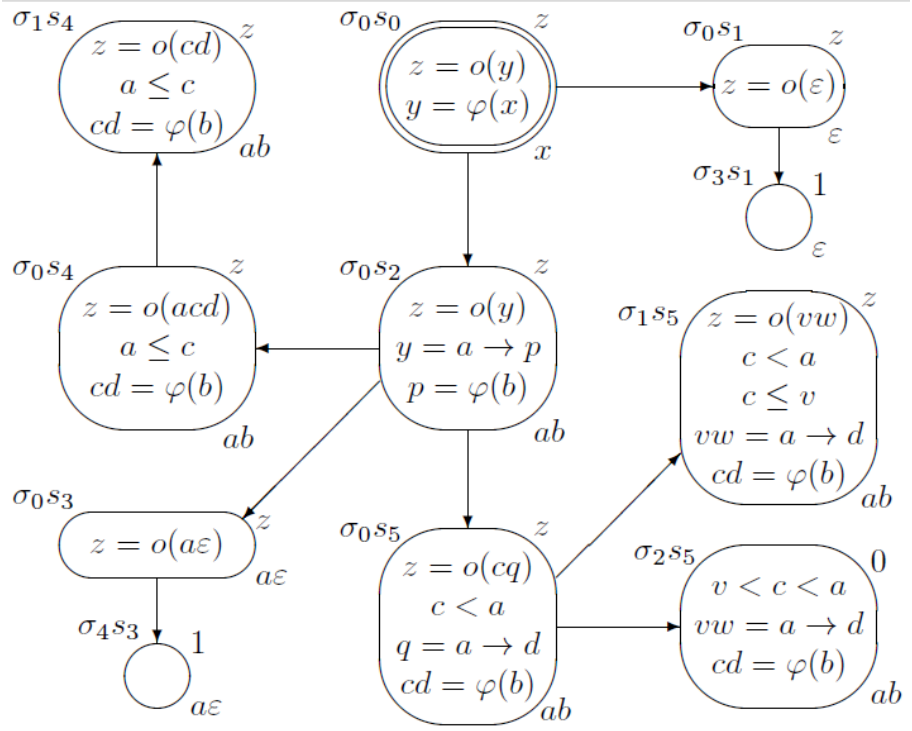
8.3. Пример верификации функциональной программы сортировки при помощи диаграммы состояний

В этом пункте мы иллюстрируем описанный выше метод верификации примером доказательства утверждения (8) для ФП, определенной в пункте 3.1.

Для доказательства равенства (38), где $\Sigma =$ (22) и $\Sigma' =$ (27), построим окрестность начального состояния $s^0_{\Sigma'(\Sigma)}$ ФП $\Sigma'(\Sigma)$ как окрестность вида UU', в соответствии с алгоритмом в пункте (8.1.2) используя

- в качестве U – окрестность (25) состояния s^0_Σ и
- в качестве U' – окрестность (28) состояния $s^0_{\Sigma'}$.

Некоторые состояния получившейся окрестности будут противоречивыми. После их удаления получим следующую окрестность:



(39)

Окрестность (39) содержит 5 листьев. Нетрудно видеть, что

- выходной терм двух из этих листьев ($\sigma_3 s_1$ и $\sigma_4 s_3$) равен 1,
- существует явное вложение листа $\sigma_1 s_4$ в состояние $\sigma_0 s_0$:

$$[b/x, cd/y] : \sigma_1 s_4 \hookrightarrow \sigma_0 s_0,$$

- существует условное вложение $\sigma_1 s_5$ в $\sigma_0 s_0$ с обоснованной посылкой $s_5 \hookrightarrow s_0$:

$$\left\{ \begin{array}{l} [vw/y] : \left\{ \begin{array}{l} c \le v \\ z = o(vw) \end{array} \right\}^z_{vw} \hookrightarrow \{z = o(y)\}^z_y \\ s_5[vw/q] \hookrightarrow s_0[\,] \end{array} \right\} : \sigma_1 s_5 \hookrightarrow \sigma_0 s_0.$$

Построим окрестность листа $\sigma_2 s_5$. Рассмотрим последователей состояния $\sigma_2 s_5$, соответствующих последователям s_6, s_7, s_8 состояния s_5.

- $\sigma_2 s_6 = \{v < u, c < a, uvw = ca\varepsilon\}^0_{ac\varepsilon}$, это состояние противоречиво.
- $\sigma_2 s_7 = \left\{ \begin{array}{l} v < c < a, c \le i \\ vw = a \to ij \\ ij = \varphi(b) \end{array} \right\}^0_{acg}$. Из данного состояния возможны два комплементарных перехода в состояния, в условии одного из которых будет конъюнктивный член $v = a$, а в условии другого – $v = i$. Нетрудно видеть, что оба эти состояния противоречивы.
- $\sigma_2 s_8 = \left\{ \begin{array}{l} v < c < a, c < r \\ vw = a \to d \\ d = r \to j \\ cj = \varphi(g) \end{array} \right\}^0_{arg}$. Из данного состояния возможны два комплементарных перехода в состояния s, s', где
  - в β_s будет конъюнктивный член $d = \varepsilon$, поэтому в β_s возможен также коньюнктивный член $v = a$, откуда нетрудно получить конъюнктивный член $a < c < a$ в β_s, т.е. s противоречиво,
  - в $\beta_{s'}$ будет конъюнктивный член $d = pq$, где p, q – новые переменные. Из s' возможны два комплементарных перехода в состояния $\tilde{s}, \tilde{s}'$, где
    - в $\beta_{\tilde{s}}$ будет конъюнктивный член $a \le p$, откуда следует, что в $\beta_{\tilde{s}}$ возможен также коньюнктивный член $v = a$, откуда нетрудно доказать противоречивость $\tilde{s}$, и
    - в $\beta_{\tilde{s}'}$ будет конъюнктивный член $p < a$, и

$$\tilde{s}' = \left\{ \begin{array}{l} v < c < a, c < r \\ w = a \to q \\ vq = r \to j \\ cj = \varphi(g) \end{array} \right\}^0_{arg} = \left\{ \begin{array}{l} v < c < a, c < r \\ vq = r \to j \\ cj = \varphi(g) \end{array} \right\}^0_{arg}.$$

Таким образом, одна из окрестностей $\sigma_2 s_5$ имеет вид

$$\sigma_2 s_5 \to \tilde{s}'. \tag{40}$$

Нетрудно видеть, что имеется явное вложение

$$[q/w, r/a, j/d, g/b] : \tilde{s}' \hookrightarrow \sigma_2 s_5.$$

Объединяя (39) и (40), получаем окрестность с пятью листьями, причем

- два из этих листьев терминальны и их выходной терм равен 1, и
- остальные три листа нетерминальны, и каждый из них вложен в некоторого своего предка.

Таким образом, объединение окрестностей (39) и (40), вместе с указанными выше вложениями нетерминальных листьев, является ДС ФП $\Sigma'(\Sigma)$.

На основании теоремы 3, заключаем, что верно равенство (38). ■

9. Заключение

В настоящей статье было введено понятие диаграммы состояний функциональной программы, и предложен метод верификации, основанный на понятии диаграммы состояний.

Главным преимуществом предложенного метода верификации является возможность его полной автоматизации: построение ДС может быть выполнено автоматически при помощи достаточно простого алгоритма.

Одна из проблем для дальнейших исследований, связанная с понятием диаграммы состояний, имеет следующий вид: найти достаточное условие (по возможности наиболее сильное) на функциональную программу, такое, что если функциональная программа удовлетворяет этому условию, то она имеет диаграмму состояний.

Список литературы

[1] А.М.Миронов: Теория функциональных программ. Часть 1. Москва, издательство ИПИ РАН, 2013. - 160 с.
`http://is.ifmo.ru/verification/2013/mironov-functional.pdf`

[2] R.W. Floyd: Assigning meanings to programs. In J.T. Schwartz, editor, Proceedings Symposium in Applied Mathematics, Mathematical Aspects of Computer Science, pages 19-32. AMS, 1967.

[3] C. A. R. Hoare: An axiomatic basis for computer programming. Communications of the ACM, 12(10): 576–580, 583, October 1969.

[4] R. Milner: A Calculus of Communicating Systems. Number 92 in Lecture Notes in Computer Science. Springer Verlag, 1980.

[5] R. Milner: Communicating and Mobile Systems: the π-Calculus. Cambridge University Press, 1999.

[6] Hoare, C. A. R.: Communicating sequential processes. Communications of the ACM 21 (8): 666–677, 1978.

[7] Separation Logic: A Logic for Shared Mutable Data Structures. John C. Reynolds. LICS 2002.

[8] Clarke, E.M., Grumberg, O., and Peled, D.: Model Checking. MIT Press, 1999.

[9] J.A. Bergstra, A. Ponse, and S.A. Smolka, editors: Handbook of Process Algebra. North-Holland, Amsterdam, 2001.

[10] C.A. Petri: Introduction to general net theory. In W. Brauer, editor, Proc. Advanced Course on General Net Theory, Processes and Systems, number 84 in LNCS, Springer Verlag, 1980.

[11] N. D. Jones and N. Andersen. Flow analysis of lazy higher-order functional programs. Theoretical Computer Science, 375:120–136, 2007.

[12] Ranjit Jhala, Rupak Majumdar, Andrey Rybalchenko: HMC: Verifying Functional Programs Using Abstract Interpreters, `http://arxiv.org/abs/1004.2884`

[13] N. Kobayashi and C.-H. L. Ong. A type theory equivalent to the modal mu-calculus model checking of higher-order recursion schemes. In Proceedings of LICS 2009. IEEE Computer Society, 2009.

[14] C.-H. L. Ong. On model- checking trees generated by higher order recursion schemes. In Proceedings 21st Annual IEEE Symposium on Logic in Computer Science, Seattle, pages 81–90. Computer Society Press, 2006.

[15] N. Kobayashi, N. Tabuchi, and H. Unno. Higher-order multiparameter tree transducers and recursion schemes for program verification. In POPL, pages 495–508, 2010.

Verification of functional programs by state diagrams

Andrew M. Mironov

In the paper we introduce graphical objects (called **state diagrams**) related to functional programs. It is shown that the state diagrams can be used to solve problems of verification of functional programs. The proposed approach is illustrated by an example of verification of a sorting program.

Keywords: program verification, functional programs, state diagrams

Полнота, устойчивость и интерпретируемость вероятностных тематических моделей

Сухарева А.В.[1]

Интерпретируемость решения, возможность обучения без учителя, масштабируемость сделали тематическое моделирование одним из наиболее популярных инструментов статистического анализа текстов. Тематические модели позволяют снизить размерность пространства данных, так как описывают каждый документ как вероятностную смесь абстрактных тем, каждую тему как распределение над словами словаря коллекции. Переход из пространства слов в пространство тем приводит к естественному решению проблем синонимии и полисемии терминов. Однако есть и ряд недостатков, вызванных зависимостью решения от инициализации. Неустойчивость тематических моделей являются общеизвестным фактом, однако связанная с ней проблема полноты до сих пор в литературе не изучалась. Для решения этой задачи в статье исследуется новый алгоритм нахождения полного набора тем, основанный на построении выпуклой оболочки. Экспериментально подтверждается эффективность данного алгоритма. На практике полный набор тем использовался в качестве инициализации модели ARTM (additive regularization for topic modeling). По сравнению с рандомизированным начальным приближением, базис тем позволяет повысить устойчивость, перплексию на более 10%, когерентность в разы.

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

[1] *Сухарева Анжелика Вячеславовна* — аспирантка каф. математических методов прогнозирования факультета вычислительной математики и кибернетики МГУ имени М.В.Ломоносова, e-mail: ang.sukhareva@gmail.com.

Sukhareva Anjelika Vyacheslavovna — graduate student, Lomonosov Moscow State University, Faculty of Computational Mathematics and Cybernetics, Chair of Mathematical Methods of Forecasting.

1. Введение

Краткая характеристика содержания документов является стандартной задачей, решаемой в области информационного поиска, статистической обработки естественного языка и машинного обучения. Такое представление можно использовать для хранения, классификации или поиска по коллекции документов. Для получения векторного представления документов часто применяются модели «мешок слов», Doc2Vec и Context2Vec. В последнее время возрос интерес к вероятностным тематическим моделям, которые описывают каждый документ как вероятностную смесь абстрактных тем, каждую тему как распределение над словами словаря коллекции. Наиболее популяными тематическими моделями стали модели pLSA (probabilistic latent semantic analysis) [1] и LDA (latent Dirichlet allocation) [2].

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 новых тем и, как следствие, неоднозначному представлению документов. Несмотря на важность требования устойчивости в задачах компьютерной лингвистики и информационного поиска, проблема до сих пор относительно мало изучена. В литературе определено понятие устойчивости [3], предлагаются меры устойчивости [4, 5, 6]. Также в работах рассматриваются модели, повышающие устойчивость.

Модель Clustered LDA [7] основана на неслучайной инициализации LDA. Идея заключается в том, чтобы сначала обучить набор моделей LDA, отличающихся случайной инициализацией. Затем выполнить кластеризацию набора тем из разных моделей. Этот кластерный набор тем формирует основу новой инициализации для LDA, которая запускается для создания модели. Интуиция заключается в том, что кластеризация является естественным способом комбинирования похожих тем. В статье показано, что размер кластера (размер — количество тем в кластере) коррелирует с повторяемостью связной темы. Темы, которые с высокой вероятностью встречаются в документах, как правило, очень повторяемы в разных сериях и, следовательно, образуют большие кластеры. И наоборот, темы, встречающиеся с меньшей вероятностью

в документах в одной модели, менее склонны иметь аналоги в других моделях.

В гранулированной LDA (GLDA) [8] используется локальная регуляризация плотности: слова в локальном контекстном окне данного слова имеют более высокую вероятность получить ту же тему, что и это слово. Предполагается, что слова, характерные для одной и той же темы, часто размещаются внутри некоторого относительно небольшого окна.

Для повышения устойчивости и интерпретируемости моделей в работе [9] применяется новый многокритериальный подход — аддитивная регуляризация тематических моделей (additive regularization for topic modeling, ARTM). Вводятся регуляризаторы для повышения разреженности и различности тем. В экспериментах показывается, что комбинирование этих регуляризаторов улучшает разреженность, чистоту и контрастность тем без значимого ухудшения правдоподобия модели. Регуляризация может существенно влиять на результаты тематического моделирования. Она может как улучшать, так и свести к минимуму устойчивость тематического моделирования [8].

TopicMapping [10] основан на неслучайной инициализации pLSA или LDA. Авторы экспериментально показали, что алгоритмы, разработанные для обнаружения сообществ в сетях, могут улучшать устойчивость тем.

В данной статье также ставится проблема полноты тематических моделей. Экспериментально исследуется взаимосвязь между полнотой, устойчивостью, интерпретируемостью и правдоподобием тематических моделей. Проблема полноты в литературе вообще 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 проблему устойчивости и полноты моделей. В разделе 3 и 4 будут исследованы устойчивость, полнота и интерпретируемость моделей. В разделе 3 вводится понятие устойчивости тем, рассматриваются различные меры устойчивости и интерпретируемости моделей. В разделе 4 вводится понятие полноты моделей. Для построения полного набора тем в разделе 4 предлагается алгоритм на основе выпуклой оболочки. Эмпирические результаты влияния полноты на устойчивость, интерпретируемость и правдоподобие моделей обсуждаются в разделе 5. Наконец, в заключении представлены наши выводы.

2. Постановка задачи

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).

Тематическая модель появления слов в документах выглядит следующим образом:

$$p(d, w) = p(d)p(w|d) = p(d) \sum_{t \in T} p(w|t)p(t|d) = p(d) \sum_{t \in T} \phi_{wt}\theta_{td}, \qquad (1)$$

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

При построение тематической модели (1) требуется оценить по известной коллекции D параметры модели $\Phi = (\phi_{wt})_{W \times T}$ и $\Theta = (\theta_{td})_{T \times D}$. Построение вероятностной тематической модели является некорректно поставленной задачей стохастического матричного разложения. Множество ее решений в общем случае бесконечно:

$$F \approx \Phi\Theta = (\Phi S)(S^{-1}\Theta) = \Phi'\Theta', \qquad (2)$$
$$\Phi \neq \Phi', \Theta \neq \Theta',$$

где $F = \big(p(d, w)\big)_{W \times D}$ — известная матрица частот, S — произвольная невырожденная матрица, при условии, что все матрицы стохастические.

Решение будем искать с помощью максимизации логарифма правдоподобия:

$$L(\Phi, \Theta) = \sum_{d \in D} \sum_{w \in W} n_{dw} \log p(d, w) = \sum_{d \in D} \sum_{w \in W} n_{dw} \log \sum_{t \in T} \phi_{wt}\theta_{td} \to \max_{\Phi, \Theta} \qquad (3)$$

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

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

Каждый раз модель находит локальный максимум правдопобия (3). Локальных экстремумов экспоненциально много и они различаются по глубине. Однако из локальных максимумов можно построить базис векторов тем тематических моделей, состоящий из линейно независимых, хорошо интерпретируемых тем. Базис будем считать *полным набором тем*, описывающим весь корпус текстов.

3. Устойчивость и интерпретируемость модели

Понятие *устойчивости тем* (stability topics) было введено в работе М. Стэйверса и Т. Гриффитса [3] в 2007 году. В статье замечено, что некоторые темы являются устойчивыми и появляются во всех моделях, другие же специфичны и встречаются только в данном решении. Авторы построили две вероятностные тематические модели, отличающиеся инициализацией. Затем измерялось различие между темами t и s с помощью симметричного расстояния Кульбака-Лейблера (Kullback-Leibler):

$$KL(t,s) = \frac{1}{2}\sum_{w\in W}\phi'_{wt}\log_2\frac{\phi'_{wt}}{\phi''_{ws}} + \frac{1}{2}\sum_{w\in W}\phi''_{ws}\log_2\frac{\phi''_{ws}}{\phi'_{wt}}, \qquad (4)$$

где ϕ', ϕ'' — оценки распределений слов по темам для двух моделей, отличающихся только случайным начальным приближением.

После чего векторы тем второй модели сопоставлялись венгерским алгоритмом [11] наиболее близким темам из первой модели. На основании экспериментов авторы сделали вывод, что по-разному проинициализированные модели дают разные решения, но многие темы устойчивы во всех запусках. Стоит отметить, что инициализация матрицы Φ сильно влияет на результат моделирования, в то время как инициализация матрицы Θ при фиксированной Φ — незначительно. Объясняется это тем, что правдоподобие (3) зависит больше от Φ, чем от Θ. На практике обычно матрица Φ инициализируется случайно. Лучшим алгоритмом инициализации матрицы Φ считается алгоритм Арори [12].

Аналогично, в [4] предлагается рассматривать расстояние между векторами распределений тем по документам d и c двух моделей с различной случайной инициализацией:

$$KL(d,c) = \sum_{t\in T}\theta'_{td}\log_2\frac{\theta'_{td}}{\theta''_{tc}}.$$

Несмотря на широкое применение KL-дивергенции (4) как меры близости между двумя распределениями, она имеет ряд недостатков при измерении сходства тем. Во-первых, в величине KL-дивергенции доминирует длинный хвост слов с низкой вероятностью. Во-вторых, KL-дивергенция зависит от размера словаря. Это означает, что различные словари и пары тем будут давать широкий диапазон значений KL-дивергенции для двух разных тем. Чтобы сделать расстояние между двумя различными темами равным 1, авторы статьи [5] вводят нормированную KL-дивергенцию:

$$NKLS(t,s) = 1 - \frac{KL(t,s)}{\max\limits_{t',s'} KL(t',s')}.$$

Под *устойчивостью тематической модели* будем понимать способность модели сохранять построенные темы в разных запусках. Устойчивость модели может применяться как внутренний критерий качества вероятностных тематических моделей [4], наряду с перплексией. В отличие от перплексии устойчивость не зависит от словаря. Она желательна по нескольким причинам [7]:

1) высокая вариативность тем, возникающих в результате различных инициализаций, может означать, что любая отдельная модель может пропустить некоторые полезные темы;

2) существует корреляция между повторяемостью и интерпретируемостью темы, интерпретируемые темы, как правило, повторяются чаще;

3) улучшение устойчивости темы — один из возможных способов устранения зашумленных тем.

Устойчивость модели можно определить, вычислив все попарные KL-дивергенции тем (или расстояние Жаккара, Хеллингера и т.д.) и усреднив их. В табл. 1 сравниваются различные инициализации модели ARTM по основным критериям, в том числе и устойчивости.

В статье [6] предлагается другой подход к измерению устойчивости модели. Здесь темы рассматриваются как множества отранжированных слов. Для определения сходства между двумя темами вычисляется среднее расстояние Жаккара (Jaccard):

$$AJ(\phi', \phi'') = \frac{1}{n}\sum_{d=1}^{n} \gamma_d(\phi', \phi''), \quad \gamma_d = \frac{|\phi'_d \cap \phi''_d|}{|\phi'_d \cup \phi''_d|},$$

где ϕ'_d — первые d топ-слов темы ϕ', n — общее число топ-слов.

Авторы предлагают меру близости тем, основанную на расстоянии Жаккара:

$$agree(\Phi_x, \Phi_y) = \frac{1}{K} \sum_{i=1}^{K} AJ\big(\phi_{xt}, \pi(\phi_{xt})\big),$$

где $\pi(\phi_{xt})$ — отранжированный список тем Φ_y, поставленных в соответствие темам Φ_x перестановкой π. По аналогии с [3] темы сопоставляются венгерским алгоритмом.

Пусть Φ_0 — матрица слова-темы тематической модели с K темами, обученной на всей выборке документов, а $\{\Phi_1, \ldots, \Phi_\tau\}$ — матрицы слова-темы той же модели, полученные на различных подвыборках этих документов. Подвыборки генерируются случайно, без возвращения документов, доля выбранных документов среди всех документов коллекции может быть любой. Тогда устойчивость тематической модели определяется как:

$$stability(K) = \frac{1}{\tau} \sum_{i=1}^{\tau} agree(\Phi_0, \Phi_i). \tag{5}$$

Данный подход может использоваться для выбора числа тем K из некоторого диапазона $[K_{\min}, K_{\max}]$. Выбирается то значение, при котором модель будет наиболее устойчива к возмущениям в данных.

Устойчивость тесно связана с таким важнейшим свойством тематических моделей как интерпретируемость. Требование интерпретируемости тем плохо формализуется. На практике часто привлекаются асессоры, которые оценивают интерпретируемость тем по топу термов с помощью различных методик, например, можно ли по набору терминов понять, о чем речь в теме или добавляется лишнее слово из другой темы и экспертам предлагается его найти и т.д. Автоматической мерой интерпретируемости принято считать когерентность [13, 14], которая оценивает совместную встречаемость главных термов темы в одних и тех же контекстах. В данной статье для оценки интерпретируемости модели дополнительно к когерентности применяются критерии частоты и контрастности тем, предложенные в работе [9]. Они основаны на гипотезе о том, что в интерпретируемой теме должно хорошо выделяться семантическое ядро (множество слов, характерных для предметной области).

Терм w относится к ядру W_t темы t, если

$$p(w|t) > \frac{1}{|W|}. \tag{6}$$

Чистота темы вводится следующим образом:

$$\sum_{w \in W_t} p(w|t). \tag{7}$$

Контрастность темы вычисляется как:

$$\frac{1}{|W_t|} \sum_{w \in W_t} p(t|w). \tag{8}$$

Чем выше чистота и контрастность тем, тем лучше.

4. Алгоритм построения полного набора тем

Из неустойчивости тем (2) 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 [15], 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:*

$$\min_{v \in conv V} \rho(\phi, v) \leq \varepsilon, \tag{9}$$

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

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 $\min\limits_{v \in conv V} \rho(\phi, v) > \varepsilon$, 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 (3). На этом и основан данный 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 (9), 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 (9) использовался алгоритм SLSQP (Sequential Least SQuares Programming) [16], который реализован во многих популярных математических пакетах, в том числе и SciPy.

5. 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

[1] https://postnauka.ru/

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. Они 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%.

5.1. Моделирование документов

Для построения полного набора использовались модели LDA (Gibbs sampling) с 20 темами. LDA[2] с параметрами $\eta = 0.01$, $\alpha = \frac{1}{20}$ обучалась 3500 итераций. Всего было построено около 2000 моделей. В полный набор Φ_0 были отобраны 23 темы.

Затем для изучения полноты, устойчивости и интерпретируемости тематических моделей обучалась модель ARTM с 23 темами. В экспериментах модель была проинициализирована двумя способами: случайно и базисом тем, который был получен, как описано выше. Во втором случае перплексия повышалась при добавлении шумовой компоненты:

$$\phi_\alpha = \alpha\phi_t + (1-\alpha)\phi'_t,$$

где $\phi_t \in \Phi_0$, $\phi'_t \sim \mathrm{Dir}(\beta)$ — шум.

В экспериментах на тегах коллекции ПостНаука брались $\alpha = 0.5$ и $\beta = 10$ для сглаживания Φ_0. Встряхивание весов позволило избежать переобучения.

При построении ARTM применялся регуляризатор разреживания матрицы Θ с различными коэффициентами. Модель ARTM строилась с помощью библиотеки BigARTM[3].

5.2. Влияние полноты на устойчивость и интерпретируемость тематических моделей

Для изучения связи между полнотой, устойчивостью и интерпретируемостью тематических моделей мы провели такой эксперимент:

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

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

- построили базис тем для моделей LDA с 20 темами, алгоритм отобрал 23 темы;
- сравнили качество моделирования модели ARTM при фиксированной матрице Φ, матрица Θ бралась случайной. Рассмотрели два случая. В первом случае матрица Φ была случайной из равномерного распределения, во втором — проинициализирована полным набором тем.

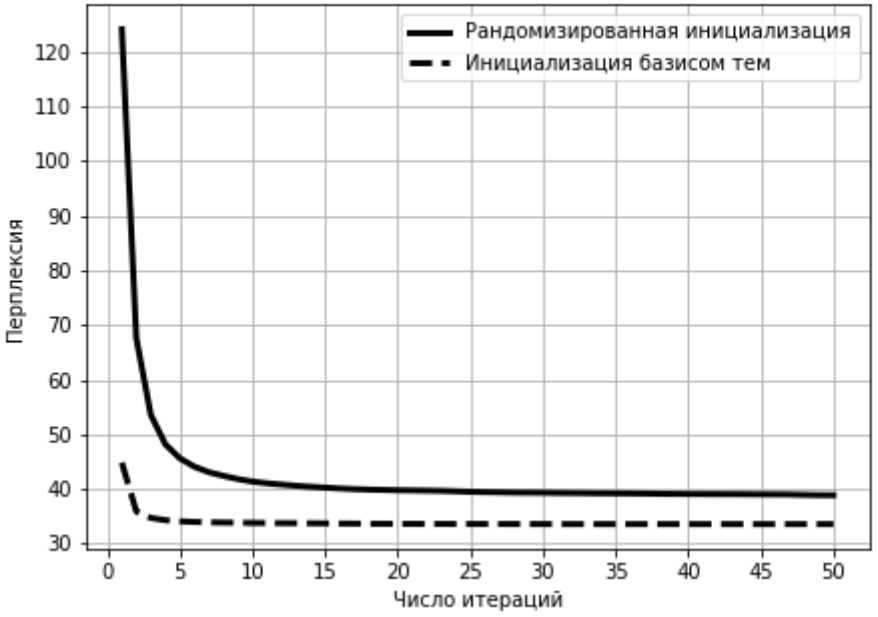


Рис. 1. Обучение модели ARTM, проинициализированной случайно и полным набором тем.

Процесс обучения модели ARTM проиллюстрирован на рис. 1. На графике видно, что модель сошлась по перплексии. Инициализация модели полным набором позволила повысить перплексию на 13%.

Критерий	ARTM (случайно)	ARTM (базис тем)
Перплексия на обучении	38.44	33.53
Перплексия на контроле	15.73	14.20
Устойчивость тем по Θ	$1.59 \cdot 10^{-8}$	$7.77 \cdot 10^{-9}$
Устойчивость тем по Φ	$6.32 \cdot 10^{-8}$	$1.83 \cdot 10^{-8}$
Полнота	0.4	1
Когерентность	-8.21	-3.32
Средний размер ядра	40.47	39.78
Средняя контрастность	0.95	0.93
Средняя чистота	0.9	0.85
Разреженность Φ	0.94	0.94
Разреженность Θ	0.89	0.91

Таблица 1. Сравнение результатов работы тематической модели подхода ARTM, проинициализированной случайно (первый столбец) и полным набором (второй столбец). Качество оценивалось по основным критериям, связанным с полнотой, устойчивостью и интерпретируемостью моделей. В качестве меры близости тем бралось расстояние Хеллингера. Темы считались близкими, если расстояние Хеллингера меньше 0.5.

В табл. 1 приводятся критерии, по которым велось сравнение случайной инициализации с базисом тем. Критерий устойчивости тем по матрице Φ (документов по матрице Θ) определялся как среднее расстояние Хеллингера между соответствующими распределениями тем (документов). Видно, что при фиксированной матрице Φ модель устойчива как по матрице Φ, так и по матрице Θ. Также способствует повышению устойчивости сильная разреженность матриц Φ и Θ. Однако полный набор позволяет строить решение на порядок более устойчивое, чем случайное начальное приближение.

Полнота тематической модели определялась как доля тем близких по расстоянию Хеллингера к темам базиса. Эксперимент показал, что около 40% тем модели похожи на темы базиса, остальные являются их комбинациями. Часто смешиваются абсолютно разные тематики. Это приводит к несогласованности тем, например: биология, эволюция, микробиология, экология, микробы, палеонтология, бактерии, геология,

география, земля. Следует отметить, что все темы базиса удалось найти за 20 моделей ARTM с 23 темами.

Полный набор тем позволяет строить не только более устойчивые модели, но и повышает в разы интерпретируемость (когерентность) тем. При этом наблюдается незначительное уменьшение размера ядра (6), чистоты (7) и контрастности тем (8). Темы полного набора более крупные и общие, чем при случайной инициализации. Более того, содержание тем полного набора не изменилось при моделировании.

6. Заключение

Данная работа посвящена изучению взаимосвязи устойчивости, полноты и интерпретируемости тематических моделей. Под устойчивостью тематической модели будем понимать способность модели сохранять построенные темы в разных запусках. Полнота тематической модели определялась как доля тем близких по расстоянию Хеллингера к темам базиса. Базис строился с помощью алгоритма на основе выпуклой оболочки из тем моделей, отличающихся инициализацией матрицы Ф. Свойство интерпретируемости тематических моделей плохо формализуется. На практике часто привлекаются эксперты для оценки того, насколько понятна тема людям и можно ли дать ей название. Оценки экспертов хорошо коррелируют с основной мерой интерпретируемости — когерентностью.

Неустойчивость тематических моделей являются общеизвестным фактом, однако связанная с ней проблема полноты до сих пор в литературе не изучалась. Для решения этой задачи в статье рассматривается 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 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.

Полученный базис тем использовался для инициализации модели ARTM. В статье был проведен сравнительный анализ инициализации модели ARTM полным набором и случайно по критериям, связанным с исследуемыми проблемами. Инициализация модели полным набором повышает перплексию на 13% и увеличивает скорость сходимости модели. Касательно устойчивости видно, что при фиксированной матрице Ф мо-

дель устойчива как по матрице Φ, так и по матрице Θ. Также способствует повышению устойчивости сильная разреженность матриц Φ и Θ. Однако полный набор позволяет строить решение на порядок более устойчивое, чем случайное начальное приближение.

Эксперимент показал, что около 40% тем модели ARTM похожи на темы базиса, остальные являются их комбинациями. Часто смешиваются разные тематики, что приводит к несогласованности тем.

Полный набор тем позволяет строить не только более устойчивые модели, но и повышает в разы интерпретируемость (когерентность) тем. Темы полного набора более крупные и общие, чем при случайной инициализации. Более того, содержание тем полного набора не изменилось при моделировании.

Таким образом, можно рекомендовать полный набор в качестве хорошего начального приближения матрицы Φ тематической модели. В дальнейшем планируется разработать модель, которая будет строить полный набор тем за меньшее число итераций.

Список литературы

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

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

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

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

[5] 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.

[6] 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.

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

[8] Koltcov S. et al., "Stable topic modeling with local density regularization", *International Conference on Internet Science*, "Springer", 2016, 176–188.

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

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

[11] Kuhn, Harold W., "The Hungarian method for the assignment problem", *Naval research logistics quarterly*, **2**:1-2 (1955), 83–97.

[12] Arora S. et al., “A practical algorithm for topic modeling with provable guarantees”, *International Conference on Machine Learning*, 2013, 280–288.

[13] Newman D. et al., “Automatic evaluation of topic coherence”, *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, "Association for Computational Linguistics", 2010, 100–108.

[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] Сухарева А.В., Воронцов К.В., “Построение полного набора тем вероятностных тематических моделей”, *Интеллектуальные системы. Теория и приложения*, **24**:4 (2019).

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

Completeness, stability and interpretability of probabilistic topic models

Sukhareva A.V.

Interpretability of the solution, the possibility of unsupervised learning, scalability made topic modeling one of the most popular tools for statistical text analysis. Topic models make it possible to reduce the dimension of the data space, since they describe each document as a probabilistic mixture of abstract topics, each topic as distribution over the vocabulary words of a collection. The transition from the space of words into the space of topics leads to a natural solution of the problems of synonymy and polysemy of terms. However, there are a number of disadvantages caused by the dependence of the solution on the initialization. The instability of topic models is a well-known fact, but the problem of completeness related to it is still not studied in the literature. To solve this problem, the article explores a new algorithm for finding a complete set of topics based on the building of the convex hull. Experimentally confirmed the effectiveness of this algorithm. In practice, a complete set of topics was used as the initialization of the ARTM (additive regularization for topic modeling) model. Compared with the randomized initial approximation, the basis topics allows to increase stability, perplexity by more than 10%, coherence by several times.

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

Корректность конструктивной теории множеств без аксиомы объемности относительно семантики арифметической реализуемости, основанной на гиперарифметических видах

Коновалов А.Ю.[1]

Определяется семантика арифметической реализуемости для формул языка теории множеств, основанная на гиперарифметических видах. Доказывается корректность конструктивной теории множеств без аксиомы объемности относительной этой семантики.

Ключевые слова: конструктивная семантика, реализуемость, арифметическая реализуемость, аксиоматическая теория множеств, гиперарифметические виды.

В интуиционистской математике одним из аналогов понятия множества является *вид* как точно сформулированное условие, которому могут удовлетворять некоторые математические объекты (см. [1]), называемые в этом случае *членами* вида. В работе [2] мы определили конструктивную семантику для языка теории множеств, основанную на гиперарифметических видах. В настоящей статье мы рассмотрим модификацию этой семантики, в которой роль вычислимых функций играют арифметические функции.

Будем считать, что язык формальной арифметики LA содержит функциональные символы для всех примитивно-рекурсивных функций,

[1] *Коновалов Александр Юрьевич* — к.ф.-м.н., младший научный сотрудник лаборатории математических проблем искусственного интеллекта мех.-мат. ф-та МГУ имени М.В.Ломоносова, e-mail: alexandr.konoval@gmail.com.

Konovalov Alexander Yurievich — Candidate of Physical and Mathematical Sciences, Junior Researcher, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Laboratory of Mathematical Problems of Artificial Intelligence.

а также константы для всех натуральных чисел. Атомарные формулы (атомы) языка LA суть выражения $t_1 = t_2$, где t_1 и t_2 — термы. Более сложные формулы языка LA строятся обычным образом из атомов при помощи логических связок $\wedge$, $\vee$, $\to$, $\neg$ и кванторов $\exists$, $\forall$.

Пусть фиксировано натуральное число $n \geq 1$. *Униформизацией* формулы $\Phi(x_1, \ldots, x_n, y)$ языка LA, не содержащей параметров, отличных от $x_1, \ldots, x_n, y$, будем называть формулу

$$\Phi(x_1, \ldots, x_n, y) \wedge (\forall z < y)\, \neg\Phi(x_1, \ldots, x_n, z),$$

обозначаемую так: $\Phi^U(x_1, \ldots, x_n, y)$. Каждая такая формула задает частичную функцию $f : \mathbb{N}^n \to \mathbb{N}$, где $f(k_1, \ldots, k_n) = k$, если и только если $\mathbb{N} \models \Phi^U(k_1, \ldots, k_n, k)$, т. е. формула $\Phi^U(k_1, \ldots, k_n, k)$ истинна в стандартной интерпретации. Пусть фиксирована геделева нумерация всех формул языка LA. Формулу с геделевым номером z обозначаем Φ_z. Через I_n обозначаем множество геделевых номеров формул языка L, не содержащих параметров отличных от $x_1, \ldots, x_n, y$. Если $z \in I_n$, то посредством φ^n_z обозначим n-местную частичную функцию, задаваемую формулой Φ^U_z. Отметим, что всякая n-местная частичная функция, определимая в языке LA, имеет вид φ^n_z для некоторого $z \in I_n$. Если φ — n-местная частичная функция, и $k_1, \ldots, k_n$ — натуральные числа, то тот факт, что определено $\varphi(k_1, \ldots, k_n)$, обозначаем так: $!\varphi(k_1, \ldots, k_n)$.

Отношения на множестве натуральных чисел, принадлежащие классу Π^1_1 аналитической иерархии [3, §16.1], назовем Π^1_1-предикатами. Из [3, §16.1, теорема V] следует, что найдется такой Π^1_1-предикат $U(z, x_1, x_2)$, который является универсальным для класса всех 2-местных Π^1_1-предикатов. Натуральное число z назовем Π^1_1-индексом отношения $P(x_1, x_2)$, если имеет место $P(x_1, x_2) \iff U(z, x_1, x_2)$. Будем говорить, что отношение $P(x_1, \ldots, x_n)$ является гиперарифметическим, если $P(x_1, \ldots, x_n)$ и $\neg P(x_1, \ldots, x_n)$ суть Π^1_1-предикаты. Натуральное число z назовем Δ^1_1-индексом отношения $P(x_1, x_2)$, если $z = \mathsf{c}(z_1, z_2)$, где z_1 — Π^1_1-индекс отношения $\neg P(x_1, x_2)$, а z_2 — Π^1_1-индекс отношения $P(x_1, x_2)$. Пусть J — множество всех Δ^1_1-индексов всех 2-местных гиперарифметических отношений, а $D_z(x_1, x_2)$ — гиперарифметическое отношение, Δ^1_1-индекс которого есть z.

Посредством трансфинитной индукции для каждого ординала α определим множество Δ_α следующим образом:

$$\Delta_\alpha \rightleftharpoons \{z \in J \mid \neg\, \exists s\, \exists x\, D_z(s,x)\},\ \text{если } \alpha = 0;$$
$$\Delta_\alpha \rightleftharpoons \{z \in J \mid \forall s, x\, (D_z(s,x) \to x \in \Delta_\beta)\},\ \text{если } \alpha = \beta + 1;$$
$$\Delta_\alpha \rightleftharpoons \bigcup_{\beta<\alpha} \Delta_\beta,\ \text{если } \alpha \text{ — предельный ординал.}$$

Через Δ обозначим объединение всех множеств Δ_α, для которых ординал α конечен либо счетен.

Формулы языка теории множеств строятся из предметных переменных, констант элементов множества Δ, двухместных предикатных символов $=$ и $\in$, логических констант $\bot, \top$, логических связок $\wedge, \vee, \to$, кванторов $\forall, \exists$ и скобок по обычным правилам. При записи формул будем использовать следующие сокращения:

- $\neg\Phi \rightleftharpoons \Phi \to \bot$;
- $\exists x{\in}t\ \Phi(x) \rightleftharpoons \exists x(x{\in}t \wedge \Phi(x))$;
- $\forall x{\in}t\ \Phi(x) \rightleftharpoons \forall x(x{\in}t \to \Phi(x))$;
- $\forall x_1, \ldots, x_n(\Phi \leftrightarrow \Psi) \rightleftharpoons \forall x_1, \ldots, x_n(\Phi \to \Psi) \wedge \forall x_1, \ldots, x_n(\Psi \to \Phi)$.

Формулами с ограниченными кванторами будем называть такие формулы языка языка теории множеств, в которых все вхождения квантора $\forall$ имеют вид $\forall x{\in}t\ \Phi$, а квантора $\exists$ — $\exists x{\in}t\ \Phi$.

Фиксируем примитивно рекурсивную взаимно-однозначную функцию c, кодирующую пары натуральных чисел натуральными числами. Тогда одноместные обратные функции p_1 и p_2, где $\mathsf{p}_1(x)$ и $\mathsf{p}_2(x)$ суть первая и вторая компоненты пары с кодом x, т. е. $\mathsf{c}(\mathsf{p}_1(x), \mathsf{p}_2(x)) = x$, также примитивно рекурсивны.

Для всякого натурального числа e и произвольной замкнутой формулы Φ языка теории множеств определим отношение «e арифметически реализует Φ» (обозначение: $e\ \mathbf{r}\ \Phi$) следующим индуктивным образом:

- $e\ \mathbf{r}\ (a = b) \rightleftharpoons a = b$;
- $e\ \mathbf{r}\ (a \in b) \rightleftharpoons D_b(e, a)$;
- $e\ \mathbf{r}\ (\Phi \wedge \Psi) \rightleftharpoons \mathsf{p}_1 e\ \mathbf{r}\ \Phi$ и $\mathsf{p}_2 e\ \mathbf{r}\ \Psi$;
- $e\ \mathbf{r}\ (\Phi \vee \Psi) \rightleftharpoons (\mathsf{p}_1 e = 0$ и $\mathsf{p}_2 e\ \mathbf{r}\ \Phi)$ или $(\mathsf{p}_1 e = 1$ и $\mathsf{p}_2 e\ \mathbf{r}\ \Psi)$;

- $e\ \mathbf{r}\ \exists x\,\Phi(x) \rightleftharpoons \mathsf{p}_1 e \in \Delta$ и $\mathsf{p}_2 e\ \mathbf{r}\ \Phi(\mathsf{p}_1 e)$;

- $e\ \mathbf{r}\ \forall x_1,\ldots,x_n\,(\Phi(x_1,\ldots,x_n) \to \Psi(x_1,\ldots,x_n)) \rightleftharpoons [e \in I_{n+1}$ и для всех[1] натуральных чисел s и $a_1,\ldots,a_n \in \Delta$, если $s\ \mathbf{r}\ \Phi(a_1,\ldots,a_n)$, то $!\varphi_e^{n+1}(a_1,\ldots,a_n,s)$ и верно $\varphi_e^{n+1}(a_1,\ldots,a_n,s)\ \mathbf{r}\ \Psi(a_1,\ldots,a_n)]$, при этом список переменных $x_1,\ldots,x_n$ может быть пустым;

- $e\ \mathbf{r}\ \forall x_1,\ldots,x_n\,\Phi(x_1,\ldots,x_n) \rightleftharpoons [e\ \mathbf{r}\ \forall x_1,\ldots,x_n\,(\top \to \Phi(x_1,\ldots,x_n))]$, если список переменных $x_1,\ldots,x_n$ непуст, формула $\Phi(x_1,\ldots,x_n)$ не начинается с квантора $\forall$, и логическая связка $\to$ не является главной в $\Phi(x_1,\ldots,x_n)$.

Будем говорить, что замкнутая формула Φ языка теории множеств является *арифметически реализуемой*, если найдется такое натуральное число e, что имеет место $e\ \mathbf{r}\ \Phi$.

В работе [4] П. Ацель определил интуиционистскую теорию CZF. Аксиомы и схемы аксиом теории CZF суть следующие:

$$\forall x,y\,(\forall z\,(z \in x \leftrightarrow z \in y) \to x = y); \tag{Ext}$$

$$\exists z\,\forall x\,(x \in z \to \bot); \tag{$\varnothing$}$$

$$\forall x\,\exists z\,(x \in z \wedge \forall u{\in}z\,\exists u'{\in}z\,\forall y\,(y \in u' \leftrightarrow y = u)); \tag{Inf}$$

$$\forall x,y\,\exists z\,\forall u\,(u \in z \leftrightarrow (u = x \vee u = y)); \tag{Pair}$$

$$\forall x\,\exists z\,\forall u\,(u \in z \leftrightarrow \exists y\,(y \in x \wedge u \in y)); \tag{Un}$$

$$\forall x\,(\forall u{\in}x\,\Phi(u) \to \Phi(x)) \to \forall x\,\Phi(x), \tag{Ind}$$

$$\forall x\,[\forall v{\in}x\,\exists u\,\Phi(v,u) \to \exists y\,(\forall v{\in}x\,\exists u{\in}y\,\Phi(v,u) \wedge \forall u{\in}y\,\exists v{\in}x\,\Phi(v,u))]; \tag{StrColl}$$

$$\forall x,y\,\exists z\,\forall w\,[\forall v{\in}x\,\exists u{\in}y\,\Phi(v,u,w) \to \to \exists y'{\in}z\,(\forall v{\in}x\,\exists u{\in}y'\,\Phi(v,u,w) \wedge \forall u{\in}y'\,\exists v{\in}x\,\Phi(v,u,w))]; \tag{SubsetColl}$$

$$\forall x\,\exists y\,\forall u\,(u \in y \leftrightarrow u \in x \wedge \Phi(u)). \tag{BoundSep}$$

При этом в схеме аксиом (BoundSep) предполагается, что $\Phi(u)$— формула с ограниченными кванторами.

Пусть i — натуральное число, удовлетворяющее соотношению $\varphi_{\mathsf{i}}^2(x,s) \simeq s$, и пусть a и b суть различные Δ_1^1-индексы пустого 2-местного отношения. Тогда $a,\ b \in \Delta_0$ и натуральное число $\mathsf{c}(\mathsf{i},\mathsf{i})$ арифметически

[1] Однако, если в списке $x_1,\ldots,x_n$ на некоторых позициях i и j стоят одинаковые переменные x_i и x_j, то мы не допускаем рассмотрение тех списков $a_1,\ldots,a_n$, в которых $a_i \neq a_j$.

реализует формулу $\forall z\,(z \in a \leftrightarrow z \in b)$. При этом формула $(a = b)$ не является арифметически реализуемой. Таким образом, аксиома (Ext) не является арифметически реализуемой. Обозначим через CZF^- теорию CZF без аксиомы (Ext). Верна следующая теорема.

Теорема 1. *Все аксиомы теории* CZF^- *являются арифметически реализуемыми.*

Список литературы

[1] А. Гейтинг, *Интуиционизм*, Мир, М., 1965.

[2] А. Ю. Коновалов, "Семантика реализуемости для конструктивной теории множеств, основанная на гиперарифметический предикатах", *Вестн. Моск. ун-та. Матем. Механ.*, 2017, № 3, 59–62.

[3] Х. Роджерс, *Теория рекурсивных функций и эффективная вычислимость*, Мир, М., 1972.

[4] P. Aczel, "The type theoretic interpretation of constructive set theory", *Log. Coll.*, **77** (1978), 55–66.

The non-extensional constructive set theory is sound with respect to the sematics of the arithmetic realizability based on hyperarithmetic sorts

Konovalov A.Yu.

A semantics of the arithmetical realizability based on hyperarithmetical sorts for formulas of the language of set theory is introduced. It is proved that constructive set theory without the extensionality axiom is sound with this semantics.

Keywords: constructive semantics, realizability, arithmetical realizability, axiomatic set theory, hyperarithmetical sorts.

О сокращении нелинейной глубины сверточных нейронных схем

Хапкин А.В.[1]

В работе рассматриваются одномерные сверточные схемы в базисе Маккалока-Питтса. Показано, что рассматриваемые схемы могут быть реализованы схемой из априорной и динамической части, в которой вычисления в априорной части не зависят от входных данных. При этом априорная и динамическая части имеют нелинейную глубину, равную 2.

Ключевые слова: сверточная нейронная сеть, нейронная схема, нелинейная сложность, модель Маккалока-Питтса.

1. Введение.

Сверточная нейронная сеть - специальная архитектура нейронных сетей, предложенная Яном Лекуном в 1988 году [1]. Основная идея таких сетей - последовательное использование сверточных слоев и субдискретизирующих слоёв, что от слоя к слою позволяет переходить ко все более и более высоким абстракциям, при этом на последних слоях выделяются признаки очень высокого уровня. Сеть в процессе обучения самонастраивается и учится сама выбирать такие детали путем фильтра маловажных деталей и выделения существенных. Основными преимуществами сверточных нейронных сетей являются не очень большое количество настраиваемых параметров по сравнению с обычными полносвязными сетями, удобное распараллеливание вычислений, что позволяет использовать для вычисления графические процессоры, относительная устойчивость к повороту и сдвигу исходных данных. Сверточные сети активно

[1] *Хапкин Артем Владимирович* — аспирант каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: 9880817@mail.ru.

Khapkin Artyom Vladimirovich — graduate student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

применяются в задачах компютерного зрения - распознавании образов, детекции объектов, также такие сети используются для распознавания речи, обработки аудиосигналов и анализа временных рядов.

Поэтому нужно уметь оптимизировать работу сверточных нейронных сетей, особенно важно уметь обходиться сетями с не очень большой нелинейной глубиной, так как в таком случае вытекает ряд преимуществ - это позволяет ускорять время работы сетей, делать сети более интерпретируемыми.

В данной работе рассматриваются сверточные одномерные нейронные схемы. ([2], [3]) в модели Маккалока-Питтса [4]. Выполняется разложение этих сетей на динамическую и априорную части, при этом вычисления априорной части не зависит от входных данных. Минимизируется нелинейная глубина каждой из этих частей.

2. Основные понятия и определения.

Пусть $X = \{x_1, x_2, \cdots\}$ - алфавит входных переменных. Введем следующие функции:

1) постоянная функция $g_c \equiv c, c \in \mathbb{R}$

2) сумматор $\Sigma_n(x_1, \cdots, x_n) = x_1 + x_2 + \cdots + x_n, x_i \in \mathbb{R}$

3) усилитель $f_\gamma(x) = \gamma x, \gamma, x \in \mathbb{R}$

4) функция Хевисайда, $\theta(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases}$

5) функция $F(x_1, x_2) = \begin{cases} x_1, x_2 \geq 0 \\ 0, x_2 < 0 \end{cases}$

Множество таких функций обозначим за Δ'.

По аналогии с автоматными схемами [5] определим схемы из функциональных элементов.

Каждой функции $f(x_1, \cdots, x_n) : \mathbb{R}^n \to \mathbb{R}$ можно сопоставить графический объект S с n входными стреклами и одной выходной стрелкой

(коротко входы и выход S). Входам объекта S приписаны слева направо переменные $x_1, x_2, \cdots, x_n$.

Элементы из Δ' изображаются следующим образом:

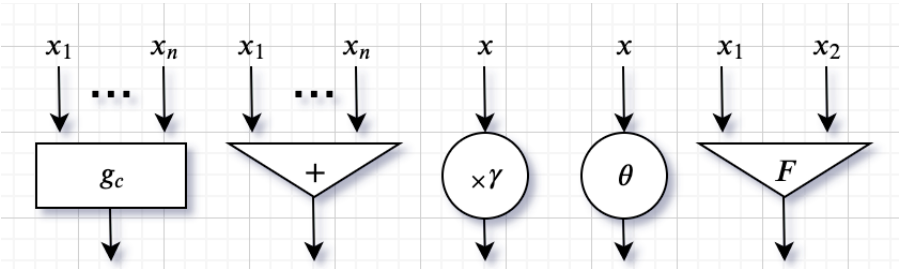


Рис. 1

Графический объект и соответствующая ему функция из Δ' называются функциональными элементами. Теперь индуктивно определим нейронные схемы без памяти. Функциональные элементы из 1-5 - нейронные схемы без памяти. Используя операции суперпозиции [6], а именно операции добавления фиктивного входа, изъятие фиктивного входа, склеивание ходов, переименование ходов без склеивания и последовательное соединение над функциональными элементами, индуктивно определяются нейронные схемы без памяти.

Если функция F реализуется нейронной схемой S, то в этом случае будем использовать запись (S, F). При этом элементы θ и F называются нелинейными, остальные элементы называются линейными. Множество функций $\mathbb{R}^n \to \mathbb{R}$, реализуемых нейронными схемами без памяти, будем обозначать $\mathbb{L}$. В работе [6] показано, что множество $\mathbb{L}$ совпадает с множеством кусочно-линейных функций PL.

Число элементов в схеме называется сложностью нейронной схемы. Число нелинейных элементов в схеме называется нелинейной сложностью нейронной схемы.

Путем в нейронной схеме называется последовательность функциональных элементов
$G_1, G_2, \cdots, G_k$, где вход G_1 является входом схемы, выход G_k является выходом схемы, и для любого $i, i = 1 \cdots k-1$ выход G_i является входом

G_{i+1} . Нелинейной длиной пути называется число нелинейных элементов, содержащихся в нем. Нелинейной глубиной нейронной схемы называется длина самого длинного пути в ней.

Две нейронные схемы эквивалентны, если они реализуют одну и ту же функцию. В работе [6] показано, что для любой нейронной схемы без памяти существует эквивалентная ей схема с нелинейной глубиной 2.

Пусть имеется схема с n входами и одним выходом, в котором последовательно применяются операции свертки и Хевисайда. Приведем ее изображение:

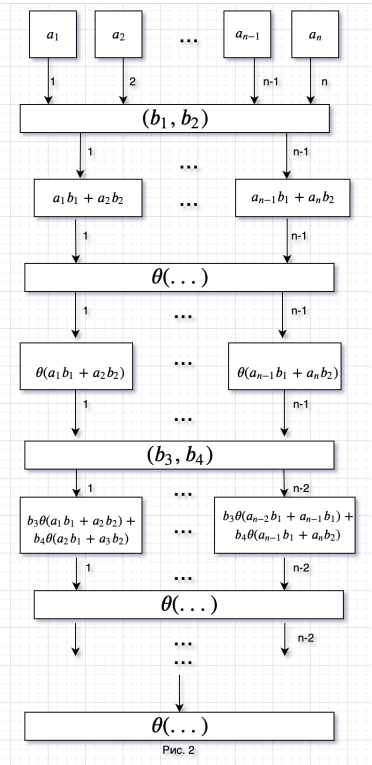


Рис. 2

В этой схеме последовательно применяются операции свертки к каждой соседней паре элементов (применяются две операции усилителя и затем одна операция сумматора между ними) и последующем применении функции Хевисайда к ним. К полученному выходу повторяем такую же последовательность операций. Последовательное применение операций свертки и функции Хевисайда будем называть одним сверточным

слоем нейронной схемы. Приведенная схема содержит $n-1$ сверточных слоев.

В результате таких операций размер входа на каждом последующем слое будет уменьшаться на 1, и в финале у схемы получится только один выход. Полученная нелинейная глубина такой схемы будет равна $n-1$, так как количество блоков с функцией Хевисайда здесь (длина пути) равно $n-1$.

Такую схему назовем одномерной сверточной схемой в базисе Маккалока-Питтса.

Нетрудно видеть, что приведенная схема на рис.2 имеет нелинейную глубину $n-1$. Распишем подробнее, что происходит в процессе вычисления схемы на рис.2.

1. Выходы первого сверточного слоя нейронной схемы (которые передаются на вход второму слою):

$$(\theta(a_1b_1 + a_2b_2), \cdots, \theta(a_{n-1}b_1 + a_nb_2)) \tag{1}$$

2. Выходы второго сверточного слоя нейронной схемы (которые передаются на вход третьему слою):

$$\begin{gathered}(\theta((b_3\theta(a_1b_1 + a_2b_2) + b_4\theta(a_2b_1 + a_3b_2)), \cdots, \\ \theta(b_3\theta(a_{n-2}b_1 + a_{n-1}b_2) + b_4\theta(a_{n-1}b_1 + a_nb_2)))\end{gathered} \tag{2}$$

Далее идет третий сверточный слой с (b_5, b_6) , и так до конца до тех пор, пока у последнего сверточного слоя останется один выход.

3. Преобразование нейронной схемы в явном виде.

3.1. Преобразования схем для сокращения нелинейной глубины.

Введем операцию коньюнкции $\wedge$. В дальнейшем она будет использоваться на элементах, принимающих значения $\{0, 1\}$. Такая операция выражается через базис Маккалока-Питтса. Нетрудно видеть, что верна следующая лемма:

Лемма 1. *Для любого $n \in \mathbb{N}$, имеет место следующее равенство:* $\theta(x_1)\wedge\theta(x_2)\wedge\cdots\wedge\theta(x_n) = \theta(x_1 + x_2 + \cdots + x_n - n)$, *где* $x_i \in 0, 1$.

3.1.1. Случай n=3.

Выпишем в общем виде структуру одномерной сверточный сети в базисе Маккалока-Питтса дли $n = 3$.

1. Значение после первой операции свертки.

$$(a_1b_1 + a_2b_2, a_2b_1 + a_3b_2) \tag{3}$$

2. Значение после первой операции Хевисайда.

$$(\theta(a_1b_1 + a_2b_2), \theta(a_2b_1 + a_3b_2)) \tag{4}$$

3. Значение после второй операции свертки.

$$b_3\theta(a_1b_1 + a_2b_2) + b_4\theta(a_2b_1 + a_3b_2) \tag{5}$$

4. Значение после второй операции Хевисайда.

$$\theta(b_3(\theta(a_1b_1 + a_2b_2) + b_4\theta(a_2b_1 + a_3b_2))) \tag{6}$$

Видно, что нелинейная глубина такой схемы равна 2. Обозначим за $a_1^{'}$ и $a_2^{'}$ результаты первоой операции Хевисайда - $\theta(a_1b_1+a_2b_2), \theta(a_2b_1+a_3b_2)$ соответственно.

Этот результат может принимать 4 разных набора значений, и при каждом из таких наборов финальный ответ известен:

$$\begin{gathered} 0,\, 0 \to 0, \\ 0,\, 1 \to \theta(b_4), \\ 1,\, 0 \to \theta(b_3), \\ 1,\, 1 \to \theta(b_3 + b_4) \end{gathered} \tag{7}$$

Заметим, что тогда исходную нейронную схему можно переписать в следующем виде:

$$(\theta(b_3) * a_1^{'} + \theta(b_4) * a_2^{'}) * (1 - a_1^{'}a_2^{'}) + \theta(b_3 + b_4) * a_1^{'}a_2^{'}, \tag{8}$$

что проверяется непосредственно подстановкой. Смысл следующий - если попадаем в область $(1, 1)$, то используем правую часть, если не попадаем - то используем левую.

Упростим такую схему:

$$(\theta(b_3) * a_1^{'} + \theta(b_4) * a_2^{'}) * (1 - a_1^{'}a_2^{'}) + \theta(b_3 + b_4) * a_1^{'}a_2^{'} =$$
$$\theta(b_3) * a_1^{'} + \theta(b_4) * a_2^{'} - \theta(b_3) * a_1^{'}a_1^{'}a_2^{'} - \theta(b_4) * a_2^{'}a_1^{'}a_2^{'}) + \theta(b_3 + b_4) * a_1^{'}a_2^{'} =$$
$$\theta(b_3) * a_1^{'} + \theta(b_4) * a_2^{'} + (\theta(b_3 + b_4) - \theta(b_3) - \theta(b_4)) * a_1^{'}a_2^{'} \tag{9}$$

Такое упрощение возможно, так как $(a_1^{'})^2 = a_1^{'}$ и $(a_2^{'})^2 = a_2^{'}$ - они всегда либо 0, либо 1.

Таким образом, мы переписали одномерную сверточную сеть с $n = 3$ в новом виде, и этот новый вид будет использоваться для сокращения нелинейной глубины при произвольном n.

3.1.2. Случай произвольного n.

Пользуясь аналогичными рассуждениями, разберем случай одномерной сверточной сети в базисе Маккалока-Питтса для произвольного n. В явном виде выпишем рекуррентные преобразования схемы, которые сократят нелинейную глубину до 2.

Выпишем преобразования схемы на рис.2 в более подробном виде: Входной вектор:

$(a_1, a_2, \cdots, a_n)$

Выходы первого сверточного слоя:

$(\theta(a_1b_1 + a_2b_2), \theta(a_2b_1 + a_3b_2), \cdots, \theta(a_{n-1}b_1 + a_nb_2))$

Выходы второго сверточного слоя:

$(\theta(b_3\theta(a_1b_1 + a_2b_2) + b_4\theta(a_2b_1 + a_3b_2)), \cdots)$

Далее, согласно преобразованиям из схемы на рис.2, спускаемся до выхода $n - 1$ слоя:

$\theta(b_{2n-3}\theta(\cdots) + (b_{2n-2}\theta(\cdots)))$

Как уже говорилось, после каждого сверточного слоя длина выхода следующего слоя будет по длине меньше на 1 чем предыдущего, и входом последнего слоя останутся 2 элемента, которые равны выходу предпоследнего сверточного слоя, обозначим их F_1^{n-2}, F_2^{n-2}.

Перед тем, как прийти к этим двум элементам, на предыдущем шаге всегда остается 3 элемента, а F_1^{n-2}, F_2^{n-2} получаются после свертки этих самых трех элементов и последующем применении функции Хевисайда. Обозначим эти три элемента за $F_1^{n-3}, F_2^{n-3}, F_3^{n-3}$. А каждый этот элемент получается на выходе свреточного слоя, вход которго составляют 4 предыдущих, и так далее, пока не дойдем до самого начала. Обозначим тогда за F_i^j элемент, который получается на свертке j, и который в векторе входов стоит на месте i.

Если переписать схему в терминах F_i^j, то получится следующее. Входной вектор:

$(a_1, a_2, \cdots, a_n)$

Выходы первого сверточного слоя:

$(F_1^1, F_2^1, \cdots, F_{n-1}^1)$

Выходы второго сверточного слоя:

$(F_1^2, F_2^2, \cdots, F_{n-2}^2)$

$\cdots$

Выходы $n-2$ сверточного слоя:

(F_1^{n-2}, F_2^{n-2})

Выходы $n-1$ сверточного слоя:

(F_1^{n-1})

Значения F_1^{n-2}, F_2^{n-2} определяются рекуррентными соотношениями, в явном виде выпишем их формулу. Слоем глубины 1 будут самые пер-

вые значение F_i^1.

$$F_1^1 = \theta(a_1b_1 + a_2b_2), F_2^1 = \theta(a_2b_1 + a_3b_2), \cdots, F_{n-1}^1 = \theta(a_{n-1}b_1 + a_nb_2) \quad (10)$$

Далее до произвольного $1 <= k < n$ на $k+1$ слое будут находиться следующие элементы:

$$\begin{gathered}
\mathrm{F}_1^{k+1} = \theta(b_{2k+1}) * F_1^k + \theta(b_{2k+2}) * F_2^k + (\theta(b_{2k+1} + b_{2k+2}) - \theta(b_{2k+1}) - \\
-\theta(b_{2k+2})) * F_1^k F_2^k \\
\mathrm{F}_2^{k+1} = \theta(b_{2k+1}) * F_2^k + \theta(b_{2k+2}) * F_3^k + (\theta(b_{2k+1} + b_{2k+2}) - \theta(b_{2k+1}) - \\
-\theta(b_{2k+2})) * F_2^k F_3^k \\
\cdots \\
\mathrm{F}_{n-k-1}^{k+1} = \theta(b_{2k+1}) * F_{n-k-1}^k + \theta(b_{2k+2}) * F_{n-k}^k + \\
+(\theta(b_{2k+1} + b_{2k+2}) - \theta(b_{2k+1}) - \theta(b_{2k+2})) * F_{n-k-1}^k F_{n-k}^k
\end{gathered} \quad (11)$$

Последний $(n-1)$ слой F_1^{n-1} можно выразить через два предпоследних функциональных элемента F_1^{n-2}, F_2^{n-2}, формула сразу следует из приведенных выше преобразований.

$$\begin{gathered}
F_1^{n-1} = (\theta(b_{2n-3})F_1^{n-2} + \theta(b_{2n-2})F_2^{n-2})) + \\
(\theta(b_{2n-3} + b_{2n-2}) - \theta(b_{2n-3} - \theta(b_{2n-2}))F_1^{n-2}F_2^{n-2}
\end{gathered} \quad (12)$$

Финальный результат такой схемы можно представить не только через последний $n-1$ слой, но и через любой другой - достаточно подставлять слагаемые по формулам, и остановиться на нужном слое. Это наблюдение будем использовать в оценке нелинейной сложности схемы.

Для того, чтобы получить нелинейную глубину 2, нужно по формулам раскрыть такую сумму до первого уровня (F_i^1), и применить лемму 1.

Таким образом, в явном виде выписаны рекуррентные соотношения, которые позволяют для любой сверточной схемы без памяти с нелинейной глубиной n получить эквивалентную ей нейронную схемы без памяти с нелинейной глубиной 2.

3.2. Разложение одномерных сверточных схем.

Если выразить результат вычисления схемы через $F_i^1, i = 1 \cdots n-1$, то получится линейная комбинаций слагаемых, при этом каждое слагаемое состоит из 2 множителей - первый множитель равен в F_i^1 или их всевозможных произведений, а второе слагаемое является комбинацией сверток, кроме первой. Следовательно, каждый такой множитель можно посчитать отдельно, в дальнейшем изобразим такие схемы.

Под схемой & подразумеваем получение всевозможных произведений (коньюкций) из переданных в этот блок элементов, под схемой $+_{selected}$ подразумевается получение определенных линейных комбинаций. Теперь изобразим, как выглядит вид такой схемы с нелинейной глубиной 2, на основе приведенных выше преобразований:

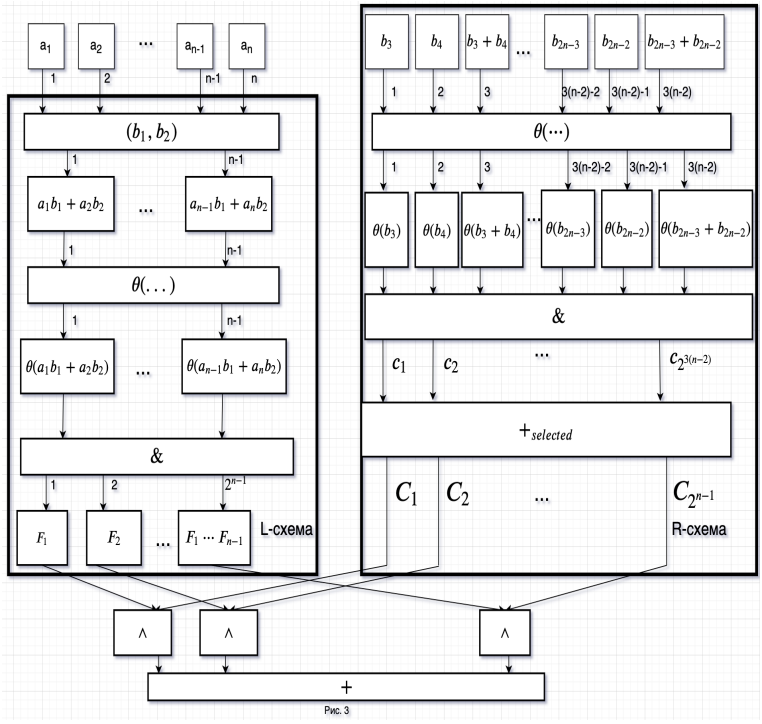


Рис. 3

Здесь c_i - различные произведения операций Хевисайда на множестве
$(b_3, b_4, b_3 + b_4, b_5, \cdots, b_{2n-3}, b_{2n-2}, b_{2n-3} + b_{2n-2})$, а C_i - это определенные линейные комбинации таких операций Хевисайда, которые полу-

чаются при раскрытии нейронной схемы в явном виде. На выходах C_i схемы $+_{selected}$ реализуется конъюнкция с i подможеством из функций $\{F_1, F_2, \cdots .F_{n-1}\}$.

R-схемой назовем схему, реализующую комбинации сверток (априорная часть схемы, которая не зависит от исходных данных), а L-схемой назовем схему, реализующую преобразования с самими данными(динамическая часть схемы). LR-схемой назовем финальную комбинацю L и R-схемы, изображенную на рис.3.

По построению видно, нелинейная глубина R-схемы и L-схемы равна 2. Схема, изображенная на рис.3, имеет глубину 3, чтобы получить нелинейную глубину 2 после объединения R и L, нужно применить лемму 1.

Такой подход обладает интересным свойством - для новых данных нужно считать заново только преобразования из L-схемы, а преобразования R-схемы достаточно посчитать один раз, так как массивы сверток для разных массивов данных будет одинаков.

Также заметим, что полученную схему можно представить в другом виде - когда выходами $L-$схемы будут не F_i^1 и их всевозможные произведения, а F_i^k и их всевозможные произведения, где $1 < k < n$, а выходами $R-$ схемы будут комбинации из меньшего числа сверток.

Имеет место следующая теорема.

Теорема 1. *Любая одномерная сверточная сеть в базисе Маккалока-Питтса представляется в виде схемы нелинейной глубины 3, изображенной на рисунке 3.*

3.3. Пример для $n = 4$.

Приведем пример работы такого алгоритма понижения нелинейной глубины. Дан массив (a_1, a_2, a_3, a_4) и даны 3 ядра свертки $(b_1, b_2), (b_3, b_4), (b_5, b_6)$. При таком алгоритме в финальной схеме останутся $F_1^1 = \theta(a_1b_1 + a_2b_2)$,$F_2^1 = \theta(a_2b_1 + a_3b_2)$, $F_3^1 = \theta(a_3b_1 + a_4b_2)$ и также множители $F_1^1F_2^1$, $F_1^1F_3^1$, $\mathrm{F}_2^1F_3^1$, $F_1^1F_2^1F_3^1$. Опустив расчеты, выпишем множители, стоящие перед каждым из таких слагаемых - результатом финальной схемы станет их сумма. Под умножением здесь понимается $\wedge$.

Перед F_1^1: $\theta(b_5)\theta(b_3)$.

Перед F_2^1: $\theta(b_5)\theta(b_4)+\theta(b_6)\theta(b_3)+(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6))\theta(b_4)\theta(b_3)$

Перед F_3^1: $\theta(b_6)\theta(b_4)$

Перед $F_1^1F_2^1$: $\theta(b_5)(\theta(b_3+b_4)-\theta(b_3)-\theta(b_4))+\theta(b_3)(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6))+$
$(\theta(b_3+b_4)-\theta(b_3)-\theta(b_4))(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6))\theta(b_3)$

Перед $F_2^1F_3^1$: $\theta(b_4)(\theta(b_3+b_4)-\theta(b_3)-\theta(b_4))+\theta(b_6)(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6))+$
$(\theta(b_3+b_4)-\theta(b_3)-\theta(b_4))(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6))\theta(b_4)$

Перед $F_1^1F_3^1$: $(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6))\theta(b_3)\theta(b_4)$

Перед $F_1^1F_2^1F_3^1$: $(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6)))(\theta(b_3+b_4)-\theta(b_3)-\theta(b_4)))$

Тогда результатом будем следующая сумма:

$$\begin{gathered}
F_1^1\wedge\theta(b_5)\theta(b_3)+F_2^1\wedge(\theta(b_5)\theta(b_4)+\theta(b_6)\theta(b_3)+\\
+(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6)\theta(b_4)\theta(b_3))+\\
+F_3^1\wedge\theta(b_6)\theta(b_4)+F_1^1F_2^1\wedge(\theta(b_5)(\theta(b_3+b_4)-\theta(b_3)-\theta(b_4))+\\
+\theta(b_3)(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6))+\\
(\theta(b_3+b_4)-\theta(b_3)-\theta(b_4))(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6))\theta(b_3))+\\
+F_2^1F_3^1\wedge(\theta(b_4)(\theta(b_3+b_4)-\theta(b_3)-\theta(b_4))+\\
+\theta(b_6)(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6))+\\
(\theta(b_3+b_4)-\theta(b_3)-\theta(b_4))(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6))\theta(b_4))+\\
F_1^1F_3^1\wedge((\theta(b_5+b_6)-\theta(b_5)-\theta(b_6))\theta(b_3)\theta(b_4))+\\
+F_1^1F_2^1F_3^1\wedge(\theta(b_5+b_6)-\theta(b_5)-\theta(b_6)))(\theta(b_3+b_4)-\theta(b_3)-\theta(b_4))
\end{gathered}\tag{13}$$

4. Оценка нелинейной сложности полученных нейронных схем.

Будем оценивать нелинейную сложность схемы, изображенной на рис.3, для случая произвольного n.

Для значения F_1^{n-1}, вычисленного схемой, имеем:

$$F_1^{n-1} = (\theta(b_{2n-3})F_1^{n-2} + \theta(b_{2n-2})F_2^{n-2}))+ \\ (\theta(b_{2n-3} + b_{2n-2}) - \theta(b_{2n-3} - \theta(b_{2n-2}))F_1^{n-2}F_2^{n-2} \tag{14}$$

В F_1^{n-1} присутствуют $F_1^{n-2}, F_2^{n-2}, F_1^{n-2}F_2^{n-2}$ в которых по рекуррентным формулам, изложенным выше, присутствуют F_i^{n-3} и все их всевозможные произведения, а в них, в свою очередь, присутсвуют F_i^{n-4} и их всевозможные произведения, и так далее.

Допустим, по рекуррентным формулам перешли к слою, на выходах которого присутствуют F_i^{n-k} и их всевозможные произведения. Выражением схемы будет сумма конъюнкций двух частей. Одна из них ответственна за исходные данные (F_i^{n-k} и их всевозможные произведения), другая от исходных входных данных никак не зависит.

Далее под финальным выражением схемы, изображенной на рис.3, понимаем результат, записанный через линейную комбинацию конъюнкций из сверток, которые не зависят от входного вектора, и F_i^{n-k} и их всевозможных произведений.

Докажем несколько лемм, которые пояснят характер каждой из этих частей.

Лемма 2. *В каждом слагаемом из суммы финального выражения обязательно будет множитель из множества F_i^{n-k} $i = 1, \cdots, k$ и их всевозможных произведений.*

Доказательство. По построению - на самом финальном уровне есть $F_1^{n-2}, F_2^{n-2}, F_1^{n-2}F_2^{n-2}$, и далее по рекуррентным преобразованиям, изложенным выше, из них и будут появляться элементы F_i^{n-k} и их всевозможные произведения. □

Лемма 3. *В каждом слагаемом из суммы финального выражения часть, которая не зависит от входного вектора, будет являться линейной комбинацией из результатов операции Хевисайда на следующем множестве:*

$$b_{2n-3}, b_{2n-2}, b_{2n-3} + b_{2n-2}), \cdots , \\ (b_{2n-(2k-1)}, b_{2n-(2k-1)}, b_{2n-(2k-2)} + b_{2n-(2k-2)}) \tag{15}$$

и их всевозможные произведения.

Доказательство. Доказательство по индукции.
На последнем слое $n-1$ это верно. На слое $n-2$ это также верно, так как в каждом множителе фигурируют элементы $\theta(b_{2n-3}), \theta(b_{2n-2}), \theta(b_{2n-3} + b_{2n-2})$. Допустим это верно для слоя $n-k$. Это значит, что каждая часть слагаемого, которая не зависит от исходного изображения, содержит в себе комбинации из следующего множества:

$$\begin{gathered}(b_{2n-3}, b_{2n-2}, b_{2n-3} + b_{2n-2}), \cdots, \\ (b_{2n-(2k-1)}, b_{2n-(2k-2)}, b_{2n-(2k-1)} + b_{2n-(2k-2)})\end{gathered} \tag{16}$$

и их всевозможные произведения. Обозначим такое множество за Θ^{n-k}.

На слое $n-k$ в финальной сумме в каждом слагаемом есть части F_i^{n-k} и их всевозможные произведения (по прошлой лемме). Части, которые не зависят от начальных данных, для таких множителей будут являться линейными комбинациями результатов операций Хевисайда на множестве Θ^{n-k} по предположению индукции.

Начинаем использовать основное преобразование. Для произвольного j, $1 \le j < n-k$ верно следующее:

$$\begin{gathered}F_j^{n-k} = \theta(b_{2n-(2k-3)}) * F_j^{n-k-1} + \theta(b_{2n-(2k-4)}) * F_{j+1}^{n-k-1} + \\ (\theta(b_{2n-(2k-3)} + b_{2n-(2k-4)}) - \theta(b_{2n-(2k-3)}) - \theta(b_{2n-(2k-4)})) * F_j^{n-k-1} F_{j+1}^{n-k-1}\end{gathered} \tag{17}$$

Согласно такому преобразованию, каждые элементы после раскрытия всех скобок будут умножаться на одно из $\theta(b_{2n-(2k-3)})$, $\theta(b_{2n-(2k-4)})$, $\theta(b_{2n-(2k-3)} + b_{2n-(2k-4)})$, либо их линейные комбинации, и обозначим такое множество за R^k.

Зафиксируем произвольный элемент из уровня $n-k-1$, который зависит от исходного изображения. Без ограничения общности, пусть это будет $F_{i_1}^{n-k-1} F_{i_2}^{n-k-1} \cdots F_{i_l}^{n-k-1}$. Рассмотрим, какой получится член перед ним, не зависящий от исходного изображения. Для этого нужно сгруппировать все слагаемые с множителем $F_{i_1}^{n-k-1} F_{i_2}^{n-k-1} \cdots F_{i_l}^{n-k-1}$ и

просуммировать его оставшиеся множители, которые не зависят от вектора входных данных.

Допустим, такой множитель встретился t раз. Тогда финальный множитель будет иметь вид:

$$F_{i_1}^k F_{i_2}^k \cdots F_{i_l}^k : \sum_{i=1}^{t} a_i * r_i * W_i \tag{18}$$

Здесь W_i - это множитель, который по предположению индукции является линейной комбинацией из множества Θ^{n-k}, $a_i \in \mathbb{Z}$, а $r_i \in R^k$.

Получается, что такая сумма будет являться линейной комбинацией из следующего множества:

$$\begin{gathered}\theta(b_{2n-3}), \theta(b_{2n-2}), \theta(b_{2n-3} + b_{2n-2}), \cdots \\ \cdots, \theta(b_{2n-(2k-3)}), \theta(b_{2n-(2k-4)}), \theta(b_{2n-(2k-3)} + b_{2n-(2k-4)})\end{gathered} \tag{19}$$

и их всевозможных произведений, то есть для уровня $n-k-1$ утверждение также верно. □

Оценим нелинейную сложность LR-схемы . Схема имеет следующий вид:

$$\begin{gathered}F_1^{n-1} = \sum_{i=1}^{n-1} x_i \wedge F_i^1 + \sum_{i,j=1, i\neq j}^{n-1} x_{ij} \wedge F_i^1 * F_j^1 + \cdots + \\ + x_{12\cdots n-1} \wedge F_1^1 \wedge F_2^1 \cdots \wedge F_{n-1}^1\end{gathered} \tag{20}$$

Здесь всевозможные x - это те самые линейные комбинации из сверток (кроме первой), про которые говорится в предыдущей лемме. Также сгруппировали такую сумму по одинаковому количеству множителей F_i^1.

Будем оценивать количество нелинейных операций, которое требуется на подсчет множителей в финальной схеме, не зависящих от исходного изображения (сверток). Нетрудно посчитать, что количество возможных произведений из сверток на последнем слое равно 0 (там нет сверток), на предпоследнем слое равно $2^3 - 1$ (всевозможные произведения $\theta(b_{2n-3})$, $\theta(b_{2n-2})$, $\theta(b_{2n-3} + b_{2n-2})$), на уровне $n-k$ количиство всевозможных произведений из встретившихся сверток равно $2^{3(k-1)} - 1$. Тогда на самом первом слое количество таких произведений будет равно $2^{3(n-2)} - 1$.

Максимальное количество множителей такого слагаемого равно количеству всевозможных элементов из сверток, то есть $3(n-2)$.

Также нужно учесть, что в рассматриваемом базисе нет операции произведения функций, поэтому ее нужно добавить, для этого нужно использовать то, что произведение Хевисайдов возможно разложить в операцию Хевисайдов от суммы операций Хевисайдов (лемма 1).

Соответственно, для того, чтобы посчитать один раз все такие множители, нужно совершить максимум следующее количество операций умножения на функции Хевисайда:
$3(n-2+1)(2^{3(n-2)}-1) = 3(n-1)(2^{3(n-2)}-1)$

Так как все множители предпосчитаны, то любая линейная комбинация не будет давать прироста к нелинейной сложности схемы - в линейной комбинации помимо произведений сверток есть только сумматор и умножение на число, которые не являются нелинейными операциями.

Теперь оценим количество нелинейных операций, которое нужно потратить на подсчет элементов, зависящих от начального изображения, то есть от F_i^1. Всевозможных комбинаций из произведений F_i^1 имеется 2^{n-1}, при этом длина самого длинного элемента не превосходит $n-1$. Также аналогично в случае с подсчетом множителей нужно учесть, что в нашем базисе нет операции произведения функций, поэтому нужно использовать лемму 1. Соответственно, чтобы посчитать всевозможные произведения, необходимо совершить $(n-1+1)2^{n-1} = n2^{n-1}$ нелинейных операций.

Итого, оценивая финальную нелинейную сложность схемы, приходим к следующей оценке: на подсчет множителей из сверток нужно $3(n-1)(2^{3(n-2)}-1)$ нелинейных операций, на подсчет множителей из элементов, которые зависят от исходного изображения, нужно $n2^{n-1}$ операций, также нужно учесть, что здесь происходит перемножение операций, и добавится еще 2^{n-1} операций для перемножения сверток и F_i^1. Итого $3(n-1)(2^{3(n-2)}-1) + n2^{n-1} + 2^{n-1}$. Видно, что наибольший вклад в рост нелинейной сложности вносит $R-$, в дальнейшем будет вестись работа по сокращению ее нелинейной сложности.

Из такой оценки можно сразу следует теорема.

Теорема 2. *Порядок роста нелинейной сложности $LR-$схемы с нелинейной глубиной 2 не превосходит $n2^{3n}$. При этом нелинейная сложность L-схемы не превосходит $n2^n$.*

Таким образом, в этой работе одномерные сверточные нейронные схемы в базисе Маккалока-Питтса разложены на динамическую и априорную составляющие, каждая из которых имеет нелинейную глубину 2.

Автор выражает благодарность своему научному руководителю, доценту кафедры МаТИС механико-математического факультета МГУ, к.ф.-м.н., Часовских Анатолию Александровичу, за помощь в проведении исследования и постановке задачи.

Список литературы

[1] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, L. D. Jacke, *Backpropagation applied to Handwritten Zip Code Recognition, Neural Computation*, 1989.

[2] *Автоматы. Сборник статей. Под ред. К. Э. Шеннона и Дж. Маккарти. Пер. с англ. под ред. А. А. Ляпунова*, Изд. иностр. лит., 1956.

[3] С. Хайкин, *Нейронные сети: полный курс, 2-е издание, Вильямс*, 2006.

[4] W. S. McCulloch, W. Pitts, *A logical calculus of the ideas immanent in nervous activity*, **5**, 1943.

[5] Кудрявцев В.Б., Алешин С.В., Подколзин А.С., *Введение в теорию автоматов*, «Наука», Москва, 1985, 320 pp.

[6] Половников В.С., *Об оптимизации струтурной реализации нейронных сетей*, МГУ, 2007

On reducing the nonlinear depth of convolutional neural schemes

Khapkin A.V.

The paper considers one-dimensional convolutional schemes in the McCulloch-Pitts basis. It is shown that the considered schemes can be implemented by a scheme from the a priori and dynamic parts, in which the calculations in the a priori part are independent of the input data. In this case, the a priori and dynamic parts have a nonlinear depth equal to 2.

Keywords: convolutional neural network, neural scheme, nonlinear complexity, McCulloch-Pitts model.

О соответствии между правильными семействами и реберными ориентациями булевых кубов

Царегородцев К.Д.[1]

В работе рассматривается соответствие между правильными семействами булевых функций и реберными ориентациями с единственным стоком на булевых кубах. Данное соответствие позволяет перенести часть результатов, полученных для указанных ориентаций, на язык правильных семейств: получить оценки на число правильных семейств порядка $n \geq 5$ и показать, что задача распознавания правильности семейства является coNP-полной.

Ключевые слова: правильные семейства булевых функций, реберные ориентации с единственным стоком.

1. Введение

Настоящая работа посвящена изучению соответствия между правильными семействами булевых функций и реберными ориентациями с единственным стоком на булевых кубах. Правильные семейства функций применяются при построении больших латинских квадратов, которые используются в различных областях математики и кибернетики: теории кодирования, планировании эксперимента, защите информации [1, 2]. В работах [3, 4] был предложен функциональный метод задания латинского квадрата при помощи семейства булевых функций.

Реберные ориентации с единственным стоком изучались в работах [5, 6] в связи с задачами линейного и квадратичного программирования. По правильному семейству можно задать ориентацию ребер на булевом кубе таким образом, что полученный объект будет реберной ориентацией

[1] *Царегородцев Кирилл Денисович* — аспирант каф. алгебры мех.-мат. ф-та МГУ имени М.В.Ломоносова, e-mail: kirill94_12@mail.ru.

Tsaregorodtsev Kirill Denisovich — graduate student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Algebra.

с единственным стоком, и наоборот, каждой такой ориентации можно сопоставить некоторое правильное семейство.

2. Предварительные сведения

Наделим булев куб E_2^n размерности n структурой графа, соединив (неориентированными) ребрами пары точек, находящихся на расстоянии Хэмминга 1. Вершинам сопоставим метки $(\alpha_1, \ldots, \alpha_n)$, $\alpha_i \in \{0, 1\}$. Будем называть такой граф графом булева куба.

Подкубом размерности $n - m$ будем называть подграф графа булева куба E_2^n, индуцированный вершинами, имеющими фиксированные значения некоторых координат $i_1, \ldots, i_m \in \{1, \ldots, n\}$.

Реберной ориентацией с единственным стоком на графе булева куба E_2^n называется такая ориентация всех его ребер, что в каждом подкубе размерности $d = 1, 2, \ldots, n$ существует ровно 1 сток. Для краткости будем в дальнейшем называть такие ориентации одностоковыми.

Следуя работам [3, 4, 7], введем понятие правильности семейства булевых функций.

Семейство булевых функций

$$F = (f_1(x_1, \ldots, x_n), \ldots, f_n(x_1, \ldots, x_n))$$

называется правильным, если для любых двух двоичных наборов α, β, $\alpha \neq \beta$, выполняется следующее условие:

$$\exists i : \alpha_i \neq \beta_i, \quad f_i(\alpha) = f_i(\beta)$$

О способах построения латинских квадратов по правильным семействам можно прочитать в работе [3].

Семейству функций $F = (f_1, \ldots, f_n)$ от n переменных с тем свойством, что f_i не зависит от x_i, $i = 1, \ldots, n$, сопоставим реберную ориентацию на графе n-мерного булева куба следующих образом. Пусть (v, w) — смежные вершины в E_2^n, $v_i \neq w_i$. По свойству семейства F имеем $f_i(v) = f_i(w)$. Если $f_i(v) = v_i$, то ребро ориентируется от w к v, в противном случае $(f_i(v) = w_i)$ ребро ориентируется от v к w. Построенный таким образом по семейству F граф обозначим за G_F.

Вершина v является стоком в данной ориентации тогда и только тогда, когда v — неподвижная точка отображения $f : E_2^n \to E_2^n$, задаваемого семейством F.

3. Связь между одностоковыми ориентациями и правильными семействами

Для правильных семейств верны следующие два утверждения, которые представляют самостоятельный интерес и играют ключевую роль в доказательстве основного результата (теорема 3).

Теорема 1. *Пусть $F = (f_1, \ldots, f_n)$ — правильное семейство булевых функций. Рассмотрим ограничение семейства F на подкуб E_2^n вида $x_{i_1} = a_1, \ldots, x_{i_m} = a_m$. Исключим из семейства F функции с номерами $i_1, \ldots, i_m$ и подставим в оставшиеся функции константы $a_1, \ldots, a_m$ вместо соответствующих переменных $x_{i_1}, \ldots, x_{i_m}$. Тогда полученное семейство функций также будет правильным.*

Теорема 2. *У отображения $E_2^n \to E_2^n$, задаваемого правильным семейством, всегда существует единственная неподвижная точка.*

Данные утверждения позволяют доказать основное утверждение про взаимосвязь правильных семейств и одностоковых ориентаций.

Теорема 3. *Граф G_F семейства $F = (f_1, \ldots, f_n)$, является одностоковой ориентацией на графе булева куба E_2^n тогда и только тогда, когда F — правильное семейство.*

Данное соответствие позволяет перенести часть результатов из теории, развитой в работах [6, 8, 9], на правильные семейства. В частности, верны следующие утверждения:

Следствие 1 ([8], [10])**.** *Пусть семейство булевых функций задано в виде конъюнктивной нормальной формы. Тогда задача распознавания правильности семейства является соNP-полной.*

Обозначим за $T(n)$ количество правильных семейств порядка n.

Следствие 2 ([6])**.** $T(n) \geq 4(T(n-1))^2$

Следствие 3 ([9])**.** *Существуют константы $B \geq A > 0$, такие что для $n \geq 2$ выполняются неравенства:*

$$n^{A \cdot 2^n} \leq T(n) \leq n^{B \cdot 2^n}$$

Автор выражает признательность своему научному руководителю А. Е. Панкратьеву и с.н.с. А. В. Галатенко за оказанную помощь при написании настоящей статьи.

Список литературы

[1] Keedwell, D., and Dénes, J., *Latin Squares and their Applications, 2nd Edition*, North Holland, 2015, 438 pp.

[2] Глухов, М. М., "О применениях квазигрупп в криптографии", *ПДМ*, 2008, 28–32.

[3] Носов, В. А., "Построение классов латинских квадратов в булевой базе данных", *Интеллектуальные системы. Теория и приложения (ранее: Интеллектуальные системы по 2014, № 2, ISSN 2075-9460)*, **4**:3–4 (1999), 307–320.

[4] Носов, В. А., "Построение параметрического семейства латинских квадратов в векторной базе данных", *Интеллектуальные системы. Теория и приложения (ранее: Интеллектуальные системы по 2014, № 2, ISSN 2075-9460)*, **8** (2006), 517–529.

[5] Szabo, T., and Welzl, E., "Unique Sink Orientations of Cubes", *Proceedings of the 42nd IEEE Symposiumon Foundations of Computer Science (FOCS)*, 2001, 547–555.

[6] Schurr, I. A., *Unique sink orientations of cubes*, ETH Zurich, 2004.

[7] Носов, В. А., Панкратьев, А. Е., "О функциональном задании латинских квадратов", *Фундамент. и прикл. матем.*, **15**:4 (2009), 177–187.

[8] Gärtner, B. and Thomas, A., "The Complexity of Recognizing Unique Sink Orientations", *Leibniz International Proceedings in Informatics, LIPIcs*, **30** (2015).

[9] Matoušek, J., "The Number Of Unique-Sink Orientations of the Hypercube", *Combinatorica*, **26** (2006), 91–99.

[10] Носов, В. А., "Критерий регулярности булевского неавтономного автомата с разделенным входом", **3** (1998), 269–280.

On the relationship between proper families and edge orientations of Boolean cubes

Tsaregorodtsev K.D.

This paper is devoted to the study of relationship between proper families of Boolean functions and unique sink orientations of cubes. A one-to-one correspondence between these two objects is established, a number of properties are carried over from unique sink orientations to proper families. These results include an upper bound for the number of proper families of given size and coNP-completeness of the problem of recognizing properness.

Keywords: proper families of Boolean functions, unique sink orientations.

Часть 3.
Математические модели

Класс автоматов, достаточный для оптимального прогнозирования общерегулярных сверхсобытий

Ведерников И.К.[1]

Автомат прогнозирует следующий символ входного сверхслова, если он выдает этот символ на выходе в предыдущий момент времени.

В работе исследуется вопрос сужения класса автоматов для задачи прогнозирования произвольного общерегулярного сверхсобытия в многозначном алфавите. Получен класс автоматов достаточный для задачи прогнозирования, а также с его помощью доказан переход от оценок простых сверхсобытий к оценкам составных.

Ключевые слова: прогнозирование автоматами, общерегулярное сверхсобытие, автоматное прогнозирование общерегулярных сверхсобытий.

1. Введение

В статье А.Г. Вереникина и Э.Э. Гасанова [1] были введены прогнозирующие автоматы — конечные автоматы, предсказывающие сверхслово или множество сверхслов. Говорят, что автомат прогнозирует сверхслово, если через некоторое конечное время после начала, он начинает в момент времени t выдавать элемент входной последовательности под номером $t+1$.

Оказалось, что полностью прогнозируемы только периодические сверхслова, изначально это было доказано для двоичного алфавита, но в работе [2] данный результат был обобщен на случай k-значных логик.

[1] *Ведерников Илья Константинович* — инженер-разработчик в ООО Независимое конструкторское бюро «Новые исследования и разработки», e-mail: nowaday@bk.ru.

Vedernikov Iliya Konstantinovich — development engineer at the Independent Design Bureau “New Research and Development” LLC.

В работе [3] А.А. Мастихиной было введено понятие частичного прогнозирования, которое имеет место в случае, когда автомат угадывает следующий символ не обязательно в каждый момент времени, но достаточно часто. В одной из следующих работ А.А. Мастихина [4] предъявила критерий частичной прогнозируемости общерегулярных сверхсобытий в двоичном алфавите. Также вопрос частичного прогнозирования сверхсобытий исследовался в работах [3, 5, 6, 7].

Первые результаты по сужению класса автоматов для задачи прогнозирования получены в [6], в частности, было доказано, что достаточно автоматов с размеченными функциями выхода. Функция выхода считается размеченной, если для каждого состояния автомата верно, что для всех значений функции перехода, ведущих в это состояние, функция выхода одинакова.

В данной работе показано, что для общерегулярных сверхсобытий достаточно рассматривать класс представляющих автоматов. Автомат считается представляющим, если для него можно выделить некоторое семейство подмножеств состояний, с помощью которого автомат будет представлять данное сверхсобытие согласно классическому определению из [8].

Также в конце приводится способ переноса оценок простых общерегулярных сверхсобытий на составные. Общерегулярное сверхсобытие простое, если его можно представить в виде сверхитерации регулярного с предпериодом, соответственно, если нельзя, то сверхсобытие составное. В частности, результаты данной работы позволяют перенести выводы работы [7] на произвольные общерегулярные сверхсобытия.

Автор выражает благодарность профессору Э.Э. Гасанову за постановку задачи и помощь в работе.

2. Основные понятия и формулировка результатов

Введем основные определения.

Пусть $E_k = \{0, 1, \ldots, k-1\}$ – конечный алфавит. Через E_k^* и E_k^∞ обозначим соответственно множество всех слов конечной длины и множество всех сверхслов в алфавите E_k. По определению будем считать, что пустое слово Λ принадлежит E_k^*. Подмножества E_k^* называются *событиями*, а подмножества E_k^∞ — *сверхсобытиями*.

Длину слова α обозначим $|\alpha|$, по определению $|\Lambda| = 0$. Если α — сверхслово, то $|\alpha| = \infty$.

Если α — сверхслово в алфавите E_k, n — натуральное число, то n-ую букву сверхслова α будем обозначать $\alpha(n)$, а через $\alpha\,]_n$ обозначим префикс длины n сверхслова α, т.е. $\alpha\,]_n = \alpha(1)\alpha(2)\ldots\alpha(n)$.

Если α — слово в алфавите E_k, $|\alpha| = m$, n — натуральное число, $n < m$, то через $[_n\,\alpha$ обозначим суффикс длины n слова α, т.е. $[_n\,\alpha = \alpha(m-n+1)\ldots\alpha(m)$. Если $n = 0$, то положим $[_n\,\alpha = \Lambda$.

В работе рассматриваются конечные инициальные автоматы в соответствии с нотацией из [8]:

$$V = (E_k, Q, E_k, \varphi, \psi, q_0),$$

где $E_k = \{0, 1, ..., k-1\}$ – входной и выходной алфавит, Q – множество состояний, которое является конечным подмножеством некоторого фиксированного счетного множества, $\varphi : Q \times E_k \to Q$ – функция переходов, $\psi : Q \times E_k \to E_k$ – функция выходов, q_0 – начальное состояние.

Если на вход инициальному автомату V подается слово или сверхслово $x = x(1)x(2)\ldots$, на выходе получается слово или сверхслово $y = y(1)y(2)\ldots$, и $q(t)$ означает состояние автомата в момент времени t, то функционирование автомата задается системой уравнений

$$\begin{cases} q(1) = q_0, \\ q(t+1) = \varphi\big(q(t), x(t)\big), \\ y(t) = \psi\big(q(t), x(t)\big), \end{cases}$$

где $t \in \mathbb{N}$.

Также определим автомат без выхода $V = (E_k, Q, \varphi, q_0)$, где E_k, Q, φ, q_0 определяются аналогично определению выше, а функционирование задается системой

$$\begin{cases} q(1) = q_0, \\ q(t+1) = \varphi\big(q(t), x(t)\big), \end{cases}$$

где $t \in \mathbb{N}$.

Функции φ и ψ естественно расширяются на $Q \times E_k^*$, а именно, если $\alpha \in E_k^*$, $a \in E_k$, то индуктивно определим

$$\varphi(q, \alpha a) = \varphi(\varphi(q, \alpha), a),$$

$$\psi(q, \alpha a) = \psi(\varphi(q, \alpha), a).$$

Введем также обозначения

$$\overline{\varphi}(q, \alpha) = \varphi(q, \alpha]_1)\varphi(q, \alpha]_2) \dots \varphi(q, \alpha),$$

$$\overline{\psi}(q, \alpha) = \psi(q, \alpha]_1)\psi(q, \alpha]_2) \dots \psi(q, \alpha),$$

если α — слово, если же α — сверхслово, то

$$\overline{\varphi}(q, \alpha) = \varphi(q, \alpha]_1)\varphi(q, \alpha]_2) \dots \varphi(q, \alpha]_n) \dots,$$

$$\overline{\psi}(q, \alpha) = \psi(q, \alpha]_1)\psi(q, \alpha]_2) \dots \psi(q, \alpha]_n) \dots.$$

Если α — сверхслово в алфавите A, то *пределом* сверхслова α назовем такое множество $A' \subseteq A$, что в сверхслове α бесконечное число раз встречаются символы из A' и только они. Этот факт будем обозначать $A' = \lim \alpha$.

Если есть событие R_1 и событие или сверхсобытие R_2, то через R_1R_2 обозначим их *произведение*, то есть все слова (сверхслова) вида ab, где $a \in R_1, b \in R_2$ (ab – конкатенация слов a и b).

Если R – событие, то обозначим R^* – *итерация* события R, то есть $R^* = R \cup R^2 \cup R^3 \cup ... \cup R^i ...$, а R^∞ – *сверхитерация* события R, $R^\infty = \{a_1a_2a_3...|\ a_i \in R, i = 1, 2, 3...\}$.

Если $Q = Q_1 \times Q_2$, то обозначим $Q_1 = Pr_1(Q),\ Q_2 = Pr_2(Q)$, и будем говорить, что $Pr_i(Q)$ – проекция множества Q на i-ую компоненту.

Сверхсобытие R *представимо* автоматом $V = (E_k, Q, E_k, \varphi, \psi, q_0)$ с помощью семейства F, $F \subseteq 2^Q$, тогда и только тогда, когда для любого $\alpha \in R$, существует $Q' \in F$, такое, что $\lim \overline{\varphi}(q_0, \alpha) = Q'$.

Определим *регулярное событие* над алфавитом E_k:

1) $\varnothing, \{a\}, a \in E_k$, — регулярные события;

2) пусть R_1, R_2 — регулярные события, тогда события $R_1R_2, R_1 \cup R_2, R_1^*$ также регулярны.

Определим *общерегулярное сверхсобытие* над алфавитом E_k:

1) если R — регулярное событие над алфавитом E_k, то R^∞ — общерегулярное сверхсобытие над алфавитом E_k;

2) если R_1 — регулярное событие над алфавитом E_k, R_2 — общерегулярное сверхсобытие над алфавитом E_k, то R_1R_2 — общерегулярное сверхсобытие над алфавитом E_k;

3) если R_1, R_2 — общерегулярные сверхсобытия над алфавитом E_k, то $R_1 \cup R_2$ — общерегулярное сверхсобытие над алфавитом E_k.

Заметим, что в соответствии с [8] любое общерегулярное сверхсобытие представимо в виде $R = T_1R_1^\infty \cup ... \cup T_hR_h^\infty$, где T_i, R_i — регулярные события.

Если общерегулярное сверхсобытие R можно представить в виде $R = T_1R_1^\infty$, будем говорить, что оно *простое*. Иначе, скажем, что оно *составное*.

Пусть $t \in \mathbb{N}$, скажем, что $(t+1)$-й символ сверхслова $\alpha = \alpha(1)\alpha(2)\ldots\alpha(t+1)\ldots$ или слова $\alpha = \alpha(1)\alpha(2)\ldots\alpha(t')$, $t' > t$, *угадан* автоматом $V = (A, Q, B, \varphi, \psi, q_0)$, если $\psi(q_0, \alpha]_t) = \alpha(t+1)$.

Пусть $\alpha \in E_k^\infty$, обозначим $\sigma_\alpha(V) = \underline{\lim}_{n\to\infty} N_n/n$, где N_n – количество угаданных автоматом V символов в слове $\alpha\,]_n$. Будем говорить, что $\sigma_\alpha(V)$ – *степень прогнозирования сверхслова* α автоматом V. Если $\alpha \in E_k^*$, $|\alpha| = n$, обозначим $\sigma_\alpha(V) = N/n$, где N – количество угаданных автоматом V символов в слове α. Будем говорить, что $\sigma_\alpha(V)$ – *степень прогнозирования слова* α автоматом V.

Считаем, что множество сверхслов R *частично прогнозируемо*, если существует такой автомат V, что степень прогнозирования для каждого сверхслова множества R строго больше нуля. Обозначим $\sigma_R(V) = \inf\limits_{\alpha\in R} \sigma_\alpha(V)$.

Если $\mathfrak{K}$ — некоторый класс автоматов, то степень прогнозирования сверхсобытия R на автоматах из этого класса определим как $\sigma_R(\mathfrak{K}) = \sup\limits_{V\in\mathfrak{K}} \sigma_R(V)$.

Будем говорить, что автомат $V = (E_k, Q, E_k, \varphi, \psi, q_0)$, прогнозирующий сверхсобытие R, лежит в классе представляющих относительно R, если существует автомат $V_0 = (E_k, Q, \varphi, q_0)$, представляющий R с помощью некоторого F, $F \subseteq 2^Q$.

Определим некоторый способ задания выходов, т.е. способ описания функции $\psi : Q \times E_k \to E_k$. Задавать выходную функцию будем, помечая ребра диаграммы Мура – в каждом состоянии отметим одно исходящее ребро. Тогда функция выхода будет описана следующим образом: если у нас отмечено ребро по символу b, $b \in E_k$, исходящее из некоторого состояния q', то для всех q и a, таких что $\varphi(q, a) = q'$, значение выходной функции $\psi(q, a)$ будет равно b. Понятно, что задавая таким образом функцию ψ для каждого состояния, в итоге мы полностью определим ее. Функцию выхода, полученную таким образом, будем называть *размеченной*.

Заметим, что угадывание происходит, когда выход в предыдущий момент времени равен входу в данный момент времени. Поэтому если автомат проходит по отмеченной стрелке, то угадывание происходит.

Обозначим $\mathfrak{A}$ – класс всех автоматов, $\mathfrak{R}(R)$ — множество автоматов, которые лежат в классе представляющих относительно R, $\mathfrak{M}$ – множество всех автоматов с размеченными функциями выхода, а $\mathfrak{MR}(R)$ — все автоматы из $\mathfrak{R}(R)$ с размеченной функцией выхода.

Далее сформулируем основные результаты данной работы:

Теорема 1. *Пусть есть общерегулярное сверхсобытие R и автомат $V = (E_k, Q, E_k, \varphi, \psi, q_0)$ такой, что $V \notin \mathfrak{R}(R)$, тогда существует автомат $V' = (E_k, Q', E_k, \varphi', \psi', q_0')$, $V' \in \mathfrak{R}(R)$, для которого выполнено, что $\sigma_R(V') \geq \sigma_R(V)$.*

Теорема 2. *Для любого общерегулярного сверхсобытия R выполнено*

$$\sigma_R(\mathfrak{A}) = \sigma_R(\mathfrak{MR}(R))$$

Утверждение 1. *Пусть есть общерегулярное сверхсобытие $R = T_1 R_1^\infty \cup \cdots \cup T_h R_h^\infty$ и выполнено, что $\sigma_{T_i R_i^\infty}(\mathfrak{MR}(T_i R_i^\infty)) \leq g_i$, тогда верна оценка*

$$\sigma_R(\mathfrak{MR}(R)) \leq \min_{i=\overline{1,h}} g_i$$

3. Доказательство основных теорем

Теорема 1. *Пусть есть общерегулярное сверхсобытие R и автомат $V = (E_k, Q, E_k, \varphi, \psi, q_0)$ такой, что $V \notin \mathfrak{R}(R)$, тогда существует автомат $V' = (E_k, Q', E_k, \varphi', \psi', q_0')$, $V' \in \mathfrak{R}(R)$, для которого выполнено, что $\sigma_R(V') \geq \sigma_R(V)$.*

Доказательство. Докажем от противного. Пусть существует автомат $V_1 = (E_k, Q_1, E_k, \varphi_1, \psi_1, q_{0,1})$, $V_1 \notin \mathfrak{R}(R)$, такой, что для него выполнено $\sigma_R(V_1) \geq \sigma_R(\mathfrak{R}(R))$.

Положим $V_2 = (E_k, Q_2, E_k, \varphi_2, \psi_2, q_{0,2})$ – произвольный автомат из $\mathfrak{R}(R)$, и пусть он представляет R с помощью семейства F_2, $F_2 \subset 2^{Q_2}$. Далее построим автомат V' такой, что $V' \in \mathfrak{R}(R)$ и $\sigma_R(V') \geq \sigma_R(V_1)$. Сделаем это следующим образом:

1) Положим $Q' = Q_1 \times Q_2$, а начальное состояние $q_0' = (q_{0,1}, q_{0,2})$.

2) Функцию переходов определим как $\varphi'((q_1, q_2), a) = (\varphi_1(q_1, a), \varphi_2(q_2, a))$, где $q_1 \in Q_1,\ q_2 \in Q_2,\ a \in E_k$.

3) Функцию выходов зададим аналогично автомату V_1, т.е. $\psi'((q_1, q_2), a) = \psi_1(q_1, a)$, где $q_1 \in Q_1,\ q_2 \in Q_2,\ a \in E_k$.

Заметим, что по построению сразу выполнено, что степени прогнозирования автоматов V_1 и V' равны. Однако функция выхода автомата V' может не быть оптимальной после построения, поэтому будем считать, что $\sigma_R(V') \geq \sigma_R(V_1)$.

Далее построим F', с помощью которого автомат V' представляет R. Если существует такое $\alpha \in R$, что $\lim \varphi'(q'_0, \alpha) = L$, то добавим L в F'.

Теперь покажем, что автомат V' действительно представляет R с помощью F'. По построению семейства F' сразу выполнено, что, если $\alpha \in R$, то $\lim \varphi'(q'_0, \alpha) \in F'$. С другой стороны верно, что $\lim \varphi_2(q_{0,2}, \alpha) \in F_2$, и по построению функции φ' для любого элемента L из F' выполнено, что $Pr_2(L) \in F_2$. Отсюда получим, что если $\lim \varphi'(q'_0, \alpha)$ принадлежит F', то $Pr_2(\lim \varphi'(q'_0, \alpha))$ принадлежит F_2, а значит α было из R.

Таким образом построен автомат V', который прогнозирует R с помощью F' и угадывает R не хуже чем автомат V_1. Противоречие с оптимальностью V_1. Теорема доказана.

□

Для доказательства следующей теоремы понадобится результат, изложенный в работе [6]. В ней исследуется вопрос о достаточности размеченных функций выхода для задачи прогнозирования, и была доказана следующая теорема

Теорема 3. *Пусть автомат $V = (E_k, Q, E_k, \varphi, \psi, q_0)$ прогнозирует сверхсобытие R со степенью σ, ψ – не размеченная, тогда существует автомат $\widehat{V} = (E_k, \widehat{Q}, E_k, \widehat{\varphi}, \widehat{\psi}, \widehat{q_0})$, где $\widehat{\psi}$ – размеченная, который прогнозирует R со степенью σ. Более того, если $V \in \mathfrak{R}(R)$, то $\widehat{V} \in \mathfrak{MR}(R)$.*

Зная этот результат, докажем теорему 2.

Теорема 2. *Для любого общерегулярного сверхсобытия R выполнено*

$$\sigma_R(\mathfrak{A}) = \sigma_R(\mathfrak{MR}(R))$$

Доказательство. Возьмем произвольное общерегулярное сверхсобытие R и произвольный автомат V из $\mathfrak{A}$. Для них по теореме 1 существует автомат V' из $\mathfrak{R}(R)$ такой, что $\sigma_R(V) \leq \sigma_R(V')$. Далее по теореме 3 существует автомат V'' из $\mathfrak{MR}(R)$ такой, что $\sigma_R(V') \leq \sigma_R(V'')$. В итоге получаем, что $\sigma_R(V) \leq \sigma_R(V'')$, и поскольку R и V произвольные, то $\sigma_R(\mathfrak{A}) = \sigma_R(\mathfrak{MR}(R))$. □

Утверждение 1. *Пусть есть общерегулярное сверхсобытие* $R = T_1R_1^\infty \cup \cdots \cup T_hR_h^\infty$ *и выполнено, что* $\sigma_{T_iR_i^\infty}(\mathfrak{MR}(T_iR_i^\infty)) \leq g_i$, *тогда верна оценка*

$$\sigma_R(\mathfrak{MR}(R)) \leq \min_{i=\overline{1,h}} g_i$$

Доказательство. Докажем от противного, пусть $\sigma_R(\mathfrak{MR}(R)) > g_j$, где $g_j = \min_{i=\overline{1,h}} g_i$.

Возьмем автомат V из $\mathfrak{MR}(R)$ такой, что $\sigma_R(V) > g_j$. Заметим, что $T_jR_j^\infty \subset R$, следовательно, $\sigma_{T_jR_j^\infty}(V) > g_j$.

Предположим автомат V принадлежит $\mathfrak{MR}(T_jR_j^\infty)$, но тогда получим противоречие с оценкой $\sigma_{T_jR_j^\infty}\mathfrak{MR}(T_jR_j^\infty) \leq g_j$, следовательно верно обратное, т.е. $V \notin \mathfrak{MR}(T_jR_j^\infty)$. Но тогда по теореме 1 существует автомат V', $V' \in \mathfrak{MR}(T_jR_j^\infty)$, такой, что $\sigma_{T_jR_j^\infty}(V') \geq \sigma_{T_jR_j^\infty}(V) > g_j$. Получили противоречие с исходной оценкой. Утверждение доказано.

□

Список литературы

[1] Вереникин А.Г., Гасанов Э.Э., “Об автоматной детерминизации множеств сверхслов”, *Дискретная математика*, **18**:2 (2006), 84–97.

[2] Гасанов Э.Э., “Прогнозирование периодических сверхсобытий автоматами”, *Интеллектуальные системы*, **19**:1 (2015), 23–34.

[3] Мастихина А.А., “О частичном угадывании сверхслов”, *Интеллектуальные системы*, **11**:1–4 (2007), 561–572.

[4] Мастихина А.А., “Критерий частичного предвосхищения общерегулярных свехсобытий”, *Дискретная математика*, **23**:4 (2011), 103–114.

[5] Ведерников И.К., “Исследование алгоритма, задающего функцию выхода прогнозирующего автомата”, *Интеллектуальные системы*, **20**:3 (2016), 103–111.

[6] Ведерников И.К., “Критерий почти полного прогнозирования сверхслова в многозначном алфавите”, *Интеллектуальные системы*, **23**:2 (2019), 87–103.

[7] Ведерников И.К., “О верхней оценке степени частичного прогнозирования общерегулярных сверхсобытий”, *Вестн. Моск. ун-та. Матем. Механ.*, 2019, № 5, 10–16.

[8] Кудрявцев В.Б., Алешин С.В., Подколзин А.С., *Введение в теорию автоматов*, «Наука», Москва, 1985, 320 с.

A class of machine sufficient for optimal prediction of general regular super-events

Vedernikov I.K.

The machine predicts the next character of the input sequence if it outputs that character the moment before.

The paper considers the machine class contraction for the task of predicting an arbitrary general regular super-event in the multivalued alphabet. The class of machine sufficient for the predicting problem is obtained in this paper. In addition, the transition from the estimations of simple super-events to the estimations of the complex super-events is proved with the help of the class aforementioned.

Keywords: predicting machine, prediction of superwords by a machine, general regular super-events.

Оценка максимального числа ненулевых коэффициентов многочлена функции при действии группы перестановок на таблицу значений функции

Гремяков А.О.[1]

Для коэффициентов полиномов функций над конечными полями F_q рассматривается задача отыскания нижней оценки на максимум минимума числа ненулевых коэффициентов в полиноме, где максимум берется по всем функциям, а минимум — по их преобразованиям, соответствующим различным заданиям поля. При этом возможно рассматривать различные типы таких преобразований.

В работе получена оценка $L(q) \geq q - 2$ на максимум минимума числа ненулевых коэффициентов в полиноме для определенного типа преобразований, оставляющих нулевой элемент поля на месте.

Ключевые слова: коэффициенты полиномов, булевы функции, полином булевой функции, таблица значений функции, sagemath.

Основные определения, обозначения и утверждения

$\mathbb{F}_q$ — конечное поле из $q = p^n$ элементов, где p — простое.

f — функция над конечным полем $\mathbb{F}_q$, f: $\mathbb{F}_q \to \mathbb{F}_q$.

Пусть дана исходная таблица значений функции.

[1] *Гремяков Александр Олегович* — аспирант каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ имени М.В.Ломоносова, e-mail: sandshats@gmail.com.

Gremyakov Alexander Olegovich — graduate student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intelligent Systems.

x	$f(x)$
x_0	$f(x_0)$
$\vdots$	$\vdots$
x_{q-1}	$f(x_{q-1})$

Рассмотрим биективное преобразование σ, после которого таблица преобразуется следующим образом:

x	$f(x)$
$\sigma(x_0)$	$\sigma(f(x_0))$
$\vdots$	$\vdots$
$\sigma(x_{q-1})$	$\sigma(f(x_{q-1}))$

$pol_f(x_1, \ldots, x_m) = \sum\limits_{0 \le j_k < q-1} c_{j_1,\ldots,j_m} x_1^{j_1} \cdot \ldots \cdot x_m^{j_m}$ —- полином функции f в поле, $c_{j_1,\ldots,j_m} \in \mathbb{F}_q$. Заметим, что количество таких полиномов совпадает с количеством функций и равенство полиномов означает равенство функций, из чего следует, что каждая функция задается ровно одним полиномом.

Если функция зависит только от одной переменной, будем применять следующее обозначение.

Через v_f обозначим вектор значений функции. А через v_{pol_f} — вектор коэффициентов полинома функции.

Рассмотрим в поле $\mathbb{F}_7$ функцию от одной переменной с таблицей

x	$f(x)$
0	1
3	1
2	1
6	1
4	2
5	1
1	2

Тогда v_f = (1,1,1,1,2,1,2), $pol_f(x) = 1 + 4x + 2x^2 + 5x^3 + 4x^4 + 2x^5 + 5x^6$, а v_{pol_f} = (1,4,2,5,4,2,5).

S_q — множество перестановок элементов поля (симметрическая группа из q элементов), а s_q — множество перестановок элементов поля, переводящие базис в базис.

Зафиксируем S как группу перестановок элементов поля сохраняющих ноль на месте.

Через $zero_{pol}(f)$ обозначим число нулевых коэффициентов функции f в ее представлении полиномом $pol_f(x_1, \ldots, x_m)$, а число ненулевых коэффициентов обозначим через $nonzero_{pol}(f)$

Обозначим за $T^{\sigma}(f)$ множество функций, которые получаются из функции f над конечным полем $\mathbb{F}_q$, с помощью многократного применения преобразования σ и через $T^{S}(f)$ множество функций, которые получаются из функции f над конечным полем $\mathbb{F}_q$, с помощью преобразований из множества S. Соответственно $T_{pol}(f)$ множество полиномов функций, которые получаются из функции f над конечным полем $\mathbb{F}_q$.

Через $P_q(m)$ обозначим множество всех функций от m переменных над над конечным полем $\mathbb{F}_q$.

Через $l_m(f)$ обозначим минимум числа ненулевых коэффициентов в полиноме от m переменных, представляющим функцию над конечным полем $\mathbb{F}_q$, где минимум берется по классу функции, или иначе говоря её орбите на множестве всех функций под действием группы преобразований.

Через $L_m(q)$ обозначим максимум минимума числа ненулевых коэффициентов в полиноме от m переменных, представляющим функцию над конечным полем $\mathbb{F}_q$, где максимум берется по всем функциям, а минимум по их преобразованиям, соответствующим различным заданиям поля.

Условие задачи в введенных обозначениях можно записать следующим образом:

$$L = L_1(p^n) = \max_{g \in P_q} \min_{f \in T(g)} nonzero\big(pol_f(x)\big), \text{ где } P_q = P_q(1)$$

.

$$l_m(f) = l(f) = \min_{f \in T(f)} nonzero\big(pol_f(x)\big).$$

В указанных обозначениях можно сформулировать полученный результат: $L(q) \geq$ q-2 при действии на множество функций рассматриваемых биективных преобразований, оставляющих ноль на месте.

Быстрое вычисление полинома функции

Покажем, как коэффициенты полинома функции выражаются через значения некоторой матрицы и вектора значений функции одной переменной. Результат утверждения 2 можно найти в [1].

Индикаторные функции

$$j_{\delta_i}(x) = \begin{cases} 1, & x = \delta_i \\ 0, & x \neq \delta_i \end{cases}.$$

Индикаторные функции представляются такими полиномами

$$j_{\delta_i}(x) = 1 - (x - \delta_i)^{q-1}.$$

Доказательство. Каждая функция в поле представляется только одним полиномом. Очевидно индикаторные функции представляются таким полиномом. □

Если $v_{p_f} = (c_0, c_1, \cdots, c_{q-1})$, то эти коэффициенты выражаются через транспонированный вектор значений функции следующим образом:

$$\begin{pmatrix} c_0 \\ c_1 \\ c_2 \\ \vdots \\ c_{q-3} \\ c_{q-2} \\ c_{q-1} \end{pmatrix} = - \begin{pmatrix} -1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1^{q-2} & a_2^{q-2} & \cdots & a_{q-2}^{q-2} & a_{q-1}^{q-2} \\ 0 & 1^{q-3} & a_2^{q-3} & \cdots & a_{q-2}^{q-3} & a_{q-1}^{q-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1^2 & a_2^2 & \cdots & a_{q-2}^2 & a_{q-1}^2 \\ 0 & 1 & a_2 & \cdots & a_{q-2} & a_{q-1} \\ 1 & 1 & 1 & \cdots & 1 & 1 \end{pmatrix} \begin{pmatrix} f(0) \\ f(1) \\ f(a_2) \\ \vdots \\ f(a_{q-3}) \\ f(a_{q-2}) \\ f(a_{q-1}) \end{pmatrix}.$$

Доказательство. Очевидно, что функцию можно представить следующим образом $f(x) = \sum_{\sigma \in \mathbf{F}_\sigma} j_\sigma(x) \cdot f(\sigma)$. После подстановки вместо индикаторных функций их представление полиномом после простых преобразований выражения получаем

$$f(x) = \sum_{\sigma\in\mathbf{F}_\sigma} j_\sigma(x)\cdot f(\sigma) = \sum_{\sigma\in\mathbf{F}_\sigma}\Big(1-(x-\sigma)^{q-1}\Big)f(\sigma) =$$
$$\sum_{\sigma\in\mathbf{F}_\sigma}\Big(1-\frac{(x-\sigma)^q}{x-\sigma}\Big)f(\sigma) = \sum_{\sigma\in\mathbf{F}_\sigma}\Big(1-\frac{x^q-\sigma^q}{x-\sigma}\Big)f(\sigma) =$$
$$\sum_{\sigma\in\mathbf{F}_\sigma}\Big(1-(x^{q-1}+x^{q-2}\sigma+\cdots+x\sigma^{q-2}+\sigma^{q-1})\Big)f(\sigma) =$$
$$-\sum_{i=0}^{k-2}x^{q-1-i}\Big(\sum_{\sigma\in\mathbf{F}_\sigma}\sigma^i f(\sigma)\Big)+f(0).$$

Отсюда получаем представление коэффициентов полинома из утверждения, так как последнее выражение можно представить с помощью матрицы.

□

Матрицу из утверждения обозначим

$$M_q(0,1,a_2,\cdots,a_{q-2},a_{q-1}) = -\begin{pmatrix} -1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1^{q-2} & a_2^{q-2} & \cdots & a_{q-2}^{q-2} & a_{q-1}^{q-2} \\ 0 & 1^{q-3} & a_2^{q-3} & \cdots & a_{q-2}^{q-3} & a_{q-1}^{q-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1^2 & a_2^2 & \cdots & a_{q-2}^2 & a_{q-1}^2 \\ 0 & 1 & a_2 & \cdots & a_{q-2} & a_{q-1} \\ 1 & 1 & 1 & \cdots & 1 & 1 \end{pmatrix}.$$

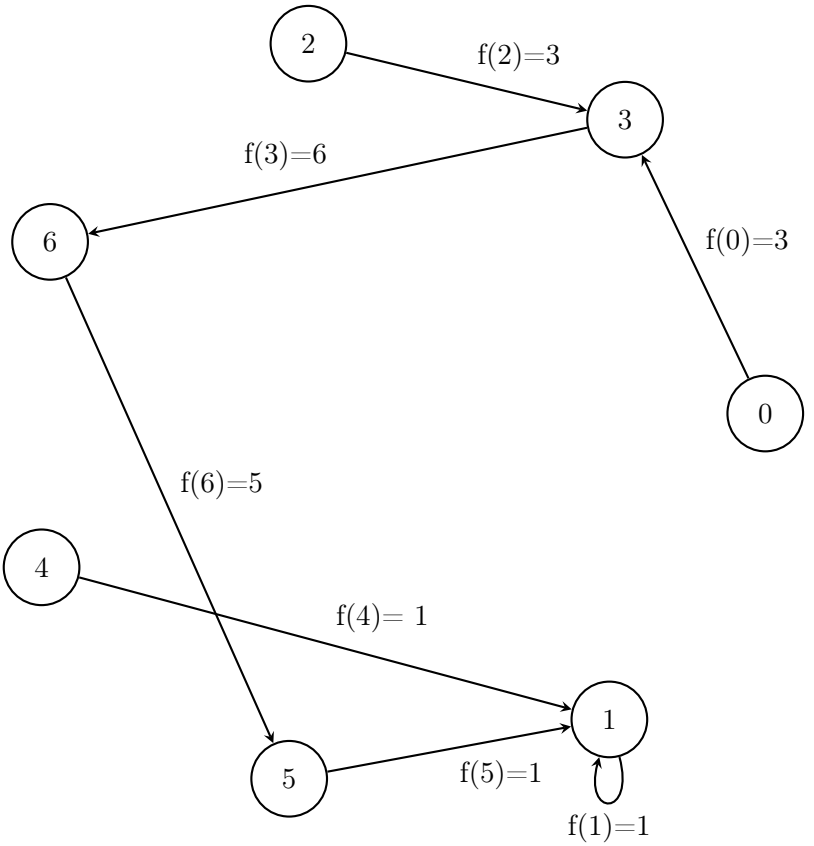
Разбиение на классы и количество классов (орбит) на множестве функций при действии на них рассматриваемыми преобразованиями

Функции из P_q можно представлять в виде ориентированных графов: вершины графов будут соответствовать значениям аргументов функций, а стрелки графа, выходящие из этих вершин будут направлены в вершины, приписанное значение аргумента которым соответствует значению функции на аргументе, значению которого соответствует вершина, откуда направлена стрелка.

Рассмотрим функцию f(x) в P_7, которая задаётся следующей таблицей:

x	$f(x)$
0	3
3	6
2	3
6	5
4	1
5	1
1	1

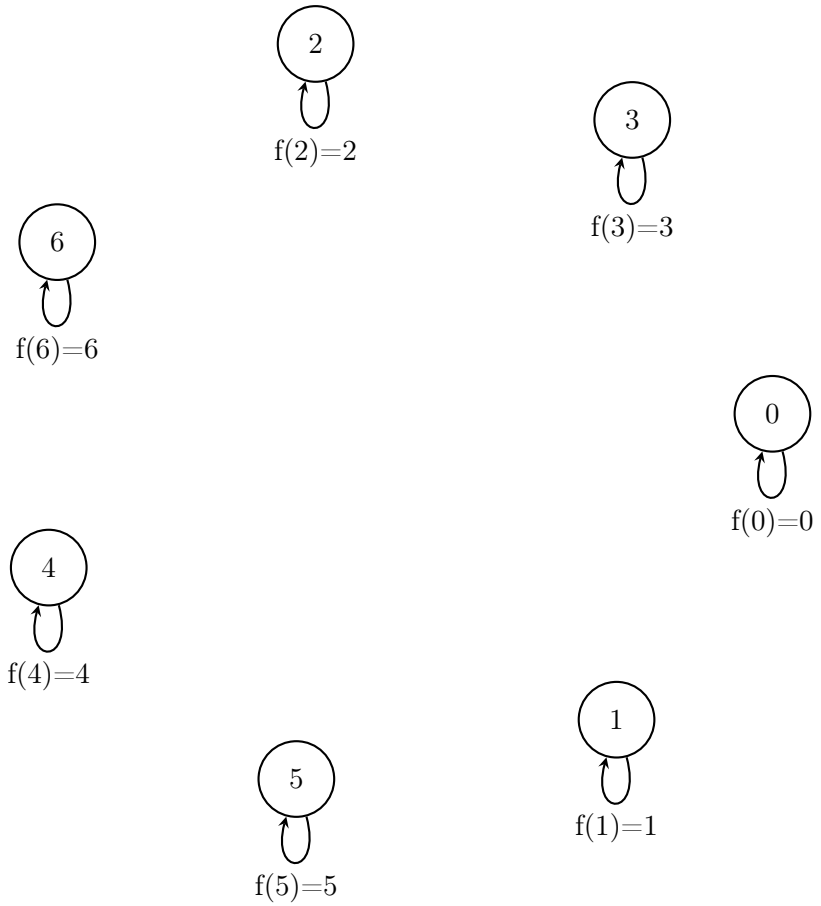
Тогда она задаётся следующим графом.



В следующих разделах не будем подписывать ребра ориентированного графа.

Рассмотрим функцию f(x) = x. Тогда очевидно, её вершины будут переходить сами в себя.

Нарисуем граф такой функции в P_7

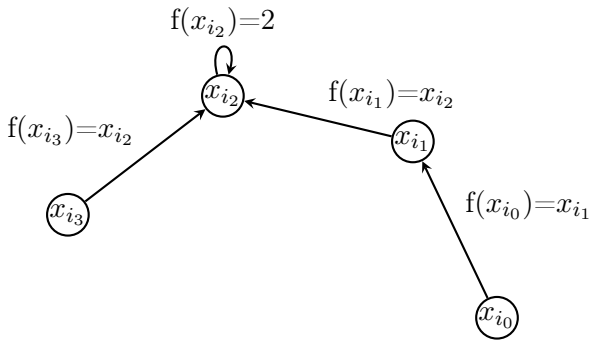


Следующее утверждение описывает разбиение функций из P_q на классы при действии преобразования рассматриваемого типа.

Граф функции не меняется под действием преобразования рассматриваемого типа.

Доказательство. Рассмотрим связный подграф какой-то функции из P_q, покажем, что он не изменится под действием преобразования.

Но очевидно, что если x_{i_j} перейдут в $\sigma(x_{i_j})$, то у рёбер и вершин ориентированного графа просто будет меняться обозначения, а сам граф будет переходить в изоморфный граф.



□

Таким образом, преобразования рассматриваемого типа сохраняют ориентированные графы функций: отображения из q точек на самих себя. Это и даёт разбиение функций на классы.

Количество таких классов описывается последовательностью A001372 на OIES.

Приведем некоторые значения этой последовательности: 1, 1, 3, 7, 19, 47, 130, 343, 951, 2615, 7318, 20491, 57903, 163898, 466199, 1328993, 3799624, 10884049, 31241170, 89814958, 258604642, 745568756, 2152118306, 6218869389, 17988233052, 52078309200, ...

Соответственно те члены, которые соответствуют $q = p^n$, будут описывать количество классов, на которые функции рабиваются под действием такого преобразования, в P_q.

Примеры, перебор $\mathbb{F}_q$ для $q \in \{2, 3, 7\}$ для преобразований рассматриваемого типа

Рассмотрим вырожденный случай: q = 2. Матрица для вычисления полиномов в поле $\mathbb{F}_2$

$$M_2(0,1) = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

В этом случае всего 4 функции и только три класса.

$l(f) = 2$ в таком классе:

Это единственное q, для которого L(q) = q.

$l(f) = 1$ в таком классе:

Это функция f(x) = x.

И остаётся класс, где содержатся константы 0 и 1. Где l(f) = 0, из-за того, что там содержится 0.

Матрица для вычисления полиномов в поле $\mathbb{F}_3$

$$M_3((0,2,1)) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 2 \\ 2 & 2 & 2 \end{pmatrix}$$

Вектора, в классах которых l(f) = q-1 (l(f)=2):

Вектора, где l(f) = q-2 (l(f) = 1):

Рассмотрим класс с вектором $v_{f_0}(x) = (2,\ 1,\ 1)$

$$v_{y_0}(x) = (0,\ 0,\ 2)\,, v_{pol_{y_0}}(x) = (0,\ 1,\ 1)$$
$$v_{y_1}(x) = (1,\ 2,\ 2)\,, v_{pol_{y_1}}(x) = (1,\ 0,\ 1)$$
$$v_{y_2}(x) = (0,\ 1,\ 0)\,, v_{pol_{y_2}}(x) = (0,\ 1,\ 2)$$
$$v_{y_3}(x) = (2,\ 1,\ 1)\,, v_{pol_{y_3}}(x) = (2,\ 0,\ 2)$$

Как видим, $v_{pol_{y_j}}$ здесь имеют не более одного нулевого коэффициента. Всего векторов в классе 6.

Рассмотрим так же и остальные классы, где достигается такая оценка: $v_{f_1}(x) = (1,\ 1,\ 2)$ и $v_{f_2}(x) = (1,\ 0,\ 2)$. Для $v_{f_1}(x) = (1,\ 1,\ 2)$ оценка в классе достигается на векторах:

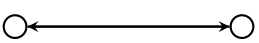
$$v_{y_4}(x) = (2,\ 0,\ 0)\,, v_{pol_{y_4}}(x) = (2,\ 0,\ 1)$$
$$v_{y_5}(x) = (1,\ 0,\ 0)\,, v_{pol_{y_5}}(x) = (1,\ 0,\ 2)$$

Все вектора в этом классе:

$$v_{y_4}(x) = (2,\ 0,\ 0)\,, v_{pol_{y_4}}(x) = (2,\ 0,\ 1)$$
$$v_{y_5}(x) = (1,\ 0,\ 0)\,, v_{pol_{y_8}}(x) = (1,\ 0,\ 2)$$
$$v_{y_6}(x) = (2,\ 1,\ 2)\,, v_{pol_{y_6}}(x) = (2,\ 2,\ 1)$$
$$v_{y_7}(x) = (1,\ 1,\ 0)\,, v_{pol_{y_7}}(x) = (1,\ 1,\ 1)$$
$$v_{y_8}(x) = (2,\ 0,\ 2)\,, v_{pol_{y_5}}(x) = (2,\ 1,\ 2)$$
$$v_{y_9}(x) = (1,\ 1,\ 2)\,, v_{pol_{y_9}}(x) = (1,\ 2,\ 2)$$

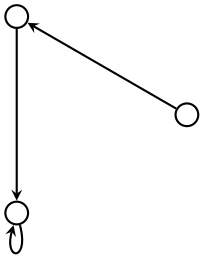
Для вектора $v_{f_2}(x) = (1,\ 0,\ 2)$ всего два вектора в классе:

$v_{f_0}(x) = (1,\, 0)$
$v_{pol_{f_0}}(x) = (1,\, 1)$

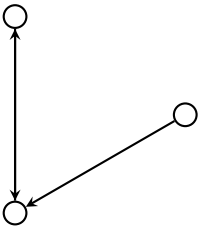


$v_{f_0}(x) = (0,\, 1)$
$v_{pol_{f_0}}(x) = (0,\, 1)$

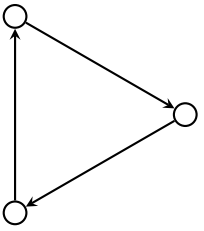
$v_{f_0}(x) = (2,\, 1,\, 1)$
$v_{pol_{f_0}}(x) = (2,\, 0,\, 2)$



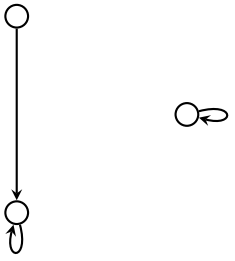
$v_{f_1}(x) = (1,\, 1,\, 2)$
$v_{pol_{f_1}}(x) = (1,\, 2,\, 2)$



$v_{f_2}(x) = (1,\, 0,\, 2)$
$v_{pol_{f_2}}(x) = (1,\, 1,\, 0)$



$v_{f_3}(x) = (0,\, 1,\, 1)$
$v_{pol_{f_3}}(x) = (0,\, 0,\, 1)$



$$v_{y_{10}}(x) = (2,\ 1,\ 0)\,, v_{pol_{y_{10}}}(x) = (2,\ 1,\ 0)$$

$$v_{y_{11}}(x) = (1,\ 0,\ 2)\,, v_{pol_{y_{11}}}(x) = (1,\ 1,\ 0)$$

И оценка достигается на обоих векторах.

Матрица для вычисления полиномов в поле $\mathbb{F}_7$

$$M_7(0,3,2,6,4,5,1) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 3 & 1 & 5 & 4 & 6 \\ 0 & 3 & 5 & 6 & 3 & 5 & 6 \\ 0 & 1 & 6 & 1 & 6 & 1 & 6 \\ 0 & 5 & 3 & 6 & 5 & 3 & 6 \\ 0 & 4 & 5 & 1 & 3 & 2 & 6 \\ 6 & 6 & 6 & 6 & 6 & 6 & 6 \end{pmatrix}$$

В P_7 оценка L(q) = q-2 достигается уже всего в трёх классах:

Порядок класса, содержащего вектор $v_{f_2}(x) = (1,\ 1,\ 1,\ 1,\ 6,\ 4,\ 5)$ равен 840, выписывать его полностью не имеет смысла, но можно привести некоторые вектора, на которых достигается оценка l(f) = q-2, самая низкая в этом классе. Опять же, таких векторов в нём всего 161, вот некоторые из них:

$$v_{y_0}(x) = (2,\ 6,\ 3,\ 4,\ 2,\ 2,\ 2)\,, v_{pol_{y_0}}(x) = (2,\ 6,\ 1,\ 5,\ 0,\ 2,\ 0)$$

$$v_{y_1}(x) = (6,\ 6,\ 4,\ 2,\ 5,\ 6,\ 6)\,, v_{pol_{y_1}}(x) = (6,\ 6,\ 5,\ 6,\ 0,\ 4,\ 0)$$

$$v_{y_2}(x) = (4,\ 4,\ 4,\ 4,\ 5,\ 1,\ 0)\,, v_{pol_{y_2}}(x) = (4,\ 4,\ 6,\ 0,\ 0,\ 1,\ 6)$$

$$v_{y_3}(x) = (4,\ 4,\ 4,\ 5,\ 6,\ 1,\ 4)\,, v_{pol_{y_3}}(x) = (4,\ 6,\ 4,\ 3,\ 0,\ 1,\ 0)$$

$$v_{y_4}(x) = (5,\ 0,\ 5,\ 5,\ 5,\ 1,\ 3)\,, v_{pol_{y_4}}(x) = (5,\ 4,\ 2,\ 0,\ 0,\ 2,\ 4)$$

$$v_{y_5}(x) = (5,\ 5,\ 5,\ 5,\ 1,\ 4,\ 3)\,, v_{pol_{y_5}}(x) = (5,\ 6,\ 6,\ 5,\ 0,\ 2,\ 0)$$

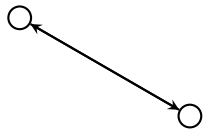
$$v_{y_6}(x) = (2,\ 2,\ 5,\ 2,\ 2,\ 1,\ 0)\,, v_{pol_{y_6}}(x) = (2,\ 0,\ 5,\ 5,\ 1,\ 1,\ 0)$$

$$v_{y_7}(x) = (2,\ 1,\ 3,\ 1,\ 1,\ 1,\ 0)\,, v_{pol_{y_7}}(x) = (2,\ 0,\ 4,\ 6,\ 0,\ 4,\ 5)$$

$$v_{y_8}(x) = (3,\ 5,\ 3,\ 3,\ 3,\ 1,\ 0)\,, v_{pol_{y_8}}(x) = (3,\ 6,\ 6,\ 3,\ 0,\ 0,\ 3)$$

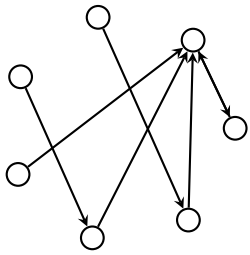
Вот некоторые вектора на которых l(f) = q-3 (l(f) = 4):

$v_{f_4}(x) = (0,\, 2,\, 1)$
$v_{pol_{f_4}}(x) = (0,\, 1,\, 0)$

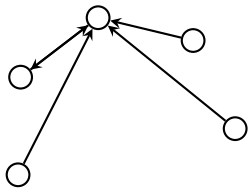


$v_{f_5}(x) = (2,\, 0,\, 1)$
$v_{pol_{f_5}}(x) = (2,\, 2,\, 0)$

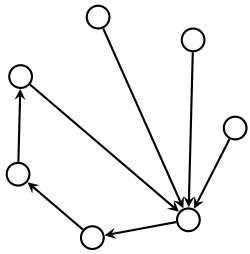
$v_{f_0}(x) = (3,\, 0,\, 1,\, 5,\, 3,\, 3,\, 3)$
$v_{pol_{f_0}}(x) = (3,\, 4,\, 0,\, 1,\, 5,\, 1,\, 3)$



$v_{f_1}(x) = (2,\, 2,\, 6,\, 2,\, 2,\, 1,\, 5)$
$v_{pol_{f_1}}(x) = (2,\, 5,\, 5,\, 6,\, 6,\, 1,\, 1)$



$v_{f_2}(x) = (1,\, 1,\, 1,\, 1,\, 6,\, 4,\, 5)$
$v_{pol_{f_2}}(x) = (1,\, 5,\, 5,\, 1,\, 2,\, 3,\, 2)$



Оценка $L(q) \geq q - 2$

Теорема 1. *$L(q) \geq q$-2 при действии на множество функций биективных преобразований, оставляющих ноль на месте.*

Доказательство. Рассмотрим класс функций, который задаётся следующим графом:

Введём обозначения $I = \{1, 2, 3, ..., q-1\}$ и $\mathfrak{I}_k = I/\{k\}, k \neq 0$.

Такие функции задаются следующим образом: $f(0) = 0, f(x_i) = x_i$, для $i \in \mathfrak{I}_k$, и $f(x_k) = x_j$, где $j \in \mathfrak{I}_k$. Заметим, что $x_k \neq x_j$, так как это разные элементы поля.

Вектор значений этой функции имеет вид, соответствующий следующим двум вариантам: 1)$v_f = (0, x_1, \cdots, x_j, \cdots, x_{q-1})$;
2)$v_f = (0, x_1, \cdots, x_j)$.

Подействуем на этот вектор матрицей $M_q(0, 1, a_2, \cdots, a_{q-2}, a_{q-1})$ (обозначение 13) и посмотрим, какие получаются коэффициенты у полинома, т.е. рассмотрим значения вектора

$$v_{pol_f} = (c_0, c_1, \cdots, c_{q_2}, c_{q-1}).$$

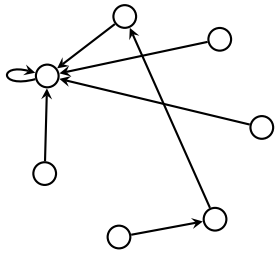
Поскольку для рассматриваемых функций все аргументы кроме одного переходят в себя, то поменяем обозначения аргументов матрицы: $M_q(0, x_1, \cdots, x_{q-1})$.

Рассмотрим коэффициент c_{q-1}. Так как последняя строка матрицы состоит из единиц, имеем $c_{q-1} = \sum\limits_{i \in \mathfrak{I}_k} x_i + x_j = -x_k + x_j \neq 0$. Отсюда получаем, что коэффициент при старшем мономе в этом классе всегда отличен от нуля.

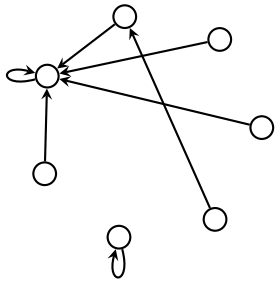
Коэффициент $c_0 = x_0 = 0$, поскольку $k \neq 0$ и x_0 это нулевой элемент поля.

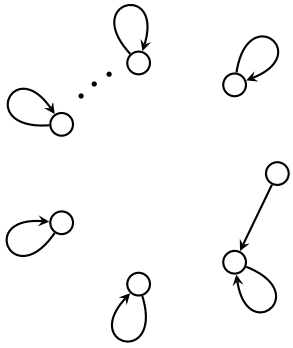
В силу того, что $k \neq 0$ коэффициенты c_{q-1-t}, где $1 \leq t \leq q-3$, равны $\sum\limits_{i \in \mathfrak{I}_k} x_i^t \cdot x_i + x_j \cdot x_k^t = \sum\limits_{i \in \mathfrak{I}_k} x_i^{t+1} + x_j \cdot x_k^t$. В силу утверждения имеем: $\sum\limits_{i \in \mathfrak{I}_k} x_i^{t+1} = -x_k^{t+1}$. Тогда получаем, что коэффициенты выражаются следующим образом: $c_{q-1-t} = -x_k^{t+1} + x_j \cdot x_k^t = x_k^t \cdot (x_j - x_k)$. В поле отсутствуют делители нуля, поэтому произведение не равно нулю, если ни один из множителей не равен нулю. При этом $x_j - x_k$ не равно нулю, так как x_j и x_k разные элементы поля, а $x_k \neq 0$ в силу того, что ноль неподвижен.

$v_{f_3}(x) = (6,\ 6,\ 6,\ 6,\ 6,\ 1,\ 2)$
$v_{pol_{f_3}}(x) = (6,\ 5,\ 0,\ 6,\ 3,\ 1,\ 2)$



$v_{f_4}(x) = (6,\ 6,\ 6,\ 6,\ 6,\ 5,\ 2)$
$v_{pol_{f_4}}(x) = (6,\ 0,\ 6,\ 3,\ 1,\ 2,\ 5)$





Отдельно рассмотрим коэффициент c_1. В этом случае имеем $c_1 = \sum\limits_{i \in \mathfrak{I}_k} x_i^{q-2} \cdot x_i + x_j \cdot x_k^{q-2} = \sum\limits_{i \in \mathfrak{I}_k} x_i^{q-1} + x_j \cdot x_k^{q-2} = \sum\limits_{i=1}^{q-2} 1 + x_j \cdot x_k^{q-2}$, который равен нулю в случае, если $x_j \cdot x_k^{q-2} = 2$.

Получаем, что в ноль обращается только c_0 и в некоторых случаях c_1.

Таким образом, на таком классе $l(f) \geq q - 2$, что доказывает утверждение теоремы.

□

Список литературы

[1] Таранников, Ю. В. Дискретная математика. Задачник : учебное пособие для академического бакалавриата / Ю. В. Таранников. — М. : Издательство Юрайт, 2016. — 385 с.

[2] Лидл Р., Нидеррайтер Г. Конечные поля. Издательство Мир, 1988

[3] Pinaki Das, The number of permutation polynomials of a given degree over a finite field 2002, Finite fields and their application 8, 478-490.

[4] Официальный сайт системы sagemath: http://www.sagemath.org/index.html

[5] The On-Line Encyclopedia of Integer Sequences, https://oeis.org/

Estimation of the maximum number of nonzero coefficients of a function polynomial under the action of a permutation group on a table of function values

Gremyakov A.O.

For coefficients of polynomials of functions over finite fields F_q, we consider the problem of finding a lower bound for the maximum minimum of the number of nonzero coefficients in the polynomial, where the maximum is taken over all functions and the minimum is taken from their transformations corresponding to various field assignments. Moreover, various types of such transformations are considered. The main results of the paper relate to two types of transformations, the description of which is given in the first section of the paper.

The paper estimates $L(q) \geq q - 2$ for the maximum minimum of the number of nonzero coefficients in the polynomial for the certain type of transformations that leave the zero field element in place.

Keywords: coefficients of polynomials, Boolean functions, polynomial of a Boolean function, table of values of a function, sagemath.

Об изменении размерности периодических подмножеств натурального ряда

Дергач П.С.[1], Булгаков Л.Р.[2]

Данная статья посвящена описанию изменения размерности периодических подмножеств натурального ряда при, казалось бы, таких незначительных операциях, как *удаление/добавление* к множеству одного числа. В работе исследуется случай, когда размерность исходного множества равна 1 или 2. Под размерностью множества понимается минимальное число непересекающихся арифметических прогрессий, дающих в объединениее это множество. Для множеств размерности 2 результат получен только в случаях пар прогрессий общего положения. В работе приводятся результаты о том, как именно меняется размерность в зависимости от того, *откуда удаляется/куда добавляется* число x.

Ключевые слова: арифметическая прогрессия, натуральный ряд, прогрессивное множество.

Введение

Данная статья основана на результатах дипломной работы Л.Р. Булгакова, выполненной под научным руководством П.С. Дергача, и посвящена описанию изменения размерности периодических подмножеств натурального ряда. Интерес к данной теме был вызван вопросом устойчи-

[1] *Дергач Пётр Сергеевич* — к.ф.-м.н., младший научный сотрудник каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ имени М.В.Ломоносова, e-mail: dergachpes@mail.ru.

Dergach Pyotr Sergeevich — Candidate of Physical and Mathematical Sciences, Junior Researcher, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intelligent Systems.

[2] *Булгаков Леон Русланович* — Студент магистратуры НИЯУ МИФИ, e-mail: leon1bulgakov@gmail.com.

Bulgakov Leon Ruslanovich — magistracy student, National Research Nuclear University MEPhI.

вости размерности множеств к операциям над ними. В качестве операций были взяты *добавление / удаление* одного элемента, а размерность исследуемых множеств было решено взять не более двух. В ходе работы были полностью решены случаи множеств размерности 1 и большинство случаев размерности 2. Оказывается, что при неудачном выборе *удаляемого/добавляемого* числа x размерность может измениться на сколь угодно большую величину. Более того, при операции добавления возможна ситуация, в которой размерность нового множества может стать равной бесконечности. При этом случай размерности 2 в равной мере является как развитием идей для размерности 1, так и добавляет новые нетривиальные идеи, присущие только множествам размерности больше 1. Приводимый результат существенно продвигает нас в решении задачи для множеств произвольной размерности, а полученные оценки являются неулучшаемыми.

Основные определения

Множество натуральных чисел обозначаем через $\mathbb{N}$. Множество целых неотрицательных чисел обозначаем через $\mathbb{N}_0$. Пусть $a \in \mathbb{N}$, $b \in \mathbb{N}$. Тогда *арифметической прогрессией с началом a и шагом b* называется множество

$$(a, b) := \{a + ib \mid i \in \mathbb{N}_0\}.$$

Множество всех арифметических прогрессий обозначаем через $\mathbb{P}^+$.

Называем множество $P \subset \mathbb{N}$ прогрессивным, если его можно представить в виде конечного объединения непересекающихся арифметических прогрессий. Минимальное число прогрессий в таком представлении называем его *мерой* и обозначаем это число через $\mu(P)$. Называем это представление *минимальным представлением* множества P.

Пусть $(a, b) \in \mathbb{P}^+$. Обозначим через $(a, b)^+$ множество

$$(a, b)^+ := \{x \in \mathbb{N} \mid b|(x - a)\}$$

и называем его *расширением* множества (a, b). Через $(a, b)^-$ обозначаем множество

$$(a, b)^- := (a, b)^+ \setminus (a, b)$$

и называем его *предпериодом* множества (a, b).

Пусть $k \in \mathbb{N}$, $k \geq 2$ и $k = p_1^{a_1} \cdot p_2^{a_2} \ldots p_{s(k)}^{a_{s(k)}}$ — разложение числа k на простые множители. Тогда вводим обозначение

$$f(k) = a_1(p_1 - 1) + a_2(p_2 - 1) + \ldots + a_{s(k)}(p_{s(k)} - 1).$$

Пусть $(a,b),(c,d) \in \mathbb{P}^+$ и $(a,b) \cap (c,d) = \emptyset$. Называем прогрессию $(e,f) \subset (a,b) \cup (c,d)$ *зигзагом*, если $(a,b) \cap (e,f) \neq \emptyset$ и $(c,d) \cap (e,f) \neq \emptyset$.

Называем множество $(a,b) \cup (c,d)$ *сжимаемым*, если все его элементы сравнимы по некоторому общему натуральному модулю $n > 1$. В противном случае, называем множество $(a,b) \cup (c,d)$ *несжимаемым.*

Если множество $(a,b) \cup (c,d)$ несжимаемо и реализуется один из трех случаев $\{b,d\} = \{2\}$, $\{b,d\} = \{2,6\}$ и $\{b,d\} = \{2,18\}$, то называем такие прогрессии *прогрессиями частного положения.*

Если множество $(a,b) \cup (c,d)$ сжимаемо и после сжатия образует прогрессии частного положения, то, опять же, называем такие прогрессии *прогрессиями частного положения.*

Прогрессиями общего положения называем пары прогрессий, которые не являются прогрессиями частного положения.

Теорема 1. *Пусть* $m \in \mathbb{N}$, $2^k < m \leq 2^{k+1}$. *Тогда* $\mu(\mathbb{N} \setminus \{m\}) = k+2$.

Замечание. Данная теорема, по сути, описывает решение не только для прогрессии $(1,1)$, но и для общего случая, так как любую прогрессию можно сжать до $(1,1)$.

Теорема 2. *Пусть* $a,b,c \in \mathbb{N}$. *Если* $a \in (b,c)^-$ *и* $2^{k-1} < \frac{b-a}{c} \leq 2^k$, *то* $\mu(\{a\} \cup (b,c)) = k+1$, *а если* $a \notin (b,c)^+$, *то* $\mu(\{a\} \cup (b,c)) = \infty$.

Теорема 3. *Пусть* $a,b,c,d,e \in \mathbb{N}, k \in \mathbb{N}_0$ *и пусть* $T = (b,c) \sqcup (d,e)$ *— пара прогрессий общего положения.*
Если $a \notin (b,c)^- \cup (d,e)^-$ *то* $\mu(\{a\} \cup T) = \infty$.
Если $a \in (b,c)^-$, $2^{k-1} < \frac{b-a}{c} \leq 2^k$, *то* $\mu(\{a\} \cup T) = k+2$.
Если $a \in (d,e)^-$, $2^{k-1} < \frac{d-a}{e} \leq 2^k$, *то* $\mu(\{a\} \cup T) = k+2$.

Теорема 4. *Пусть* $a,b,c,d,e \in \mathbb{N}, k \in \mathbb{N}_0$ *и пусть* $T = (b,c) \sqcup (d,e)$ *— пара прогрессий общего положения,* $a \in T$.
Если $a \in (b,c)$, $b + c \cdot (2^k - 1) < a \leq b + c \cdot (2^{k+1} - 1)$, *то* $\mu(T \setminus \{a\}) = k+3$.
Если $a \in (d,e)$, $d + e \cdot (2^k - 1) < a \leq d + e \cdot (2^{k+1} - 1)$, *то* $\mu(T \setminus \{a\}) = k+3$.
Иначе $\mu(T \setminus \{a\}) = 2$.

Доказательство вспомогательных утверждений

Лемма 1. *Пусть* $a,n \in \mathbb{N}$, $a \leq 2^n$. *Тогда*

$$\mu(\mathbb{N} \setminus (a, 2^n)) \leq n.$$

Доказательство.

Будем доказывать утверждение индукцией по n. При $n = 1$ получаем, что или $a = 1$ и тогда

$$\mu(\mathbb{N} \setminus (a, 2^n)) = \mu(\mathbb{N} \setminus (1, 2)) = \mu((2, 2)) = 1,$$

или $a = 2$ и тогда

$$\mu(\mathbb{N} \setminus (a, 2^n)) = \mu(\mathbb{N} \setminus (2, 2)) = \mu((1, 2)) = 1.$$

Для доказательства перехода индукции нужно рассмотреть два случая. Если $a = 2b$ — четное число, то

$$\mu(\mathbb{N} \setminus (a, 2^n)) = \mu(\mathbb{N} \setminus (2b, 2^n)) =$$

$$= \mu((1, 2) \cup \{2x | x \in (b, 2^{n-1})\}) \leq 1 + \mu((b, 2^{n-1})). \quad (1)$$

Так как $b = \frac{a}{2} \leq \frac{2^n}{2} = 2^{n-1}$, то по предположению индукции

$$\mu((b, 2^{n-1})) \leq n - 1. \quad (2)$$

Из (1) и (2) получаем требуемое в индуктивном переходе неравенство. Если же $a = 2b - 1$ — нечетное число, то

$$\mu(\mathbb{N} \setminus (a, 2^n)) = \mu(\mathbb{N} \setminus (2b - 1, 2^n)) =$$

$$= \mu((1, 2) \cup \{2x - 1 | x \in (b, 2^{n-1})\}) \leq 1 + \mu((b, 2^{n-1})). \quad (3)$$

Так как $b = \frac{a+1}{2} \leq \frac{2^n+1}{2} = 2^{n-1} + \frac{1}{2}$, то $b \leq 2^{n-1}$ и по предположению индукции

$$\mu((b, 2^{n-1})) \leq n - 1. \quad (4)$$

Из (3) и (4) получаем требуемое в индуктивном переходе неравенство.■

Лемма 2. *Пусть $k \in \mathbb{N}$ и $k = p_1^{a_1} \cdot p_2^{a_2} \dots p_{s(k)}^{a_{s(k)}}$ — разложение числа k на простые множители. Тогда*

$$f_1(k) = f_2(k) = a_1(p_1 - 1) + a_2(p_2 - 1) + \dots + a_{s(k)}(p_{s(k)} - 1).$$

Доказательство.

Доказательство леммы см. в [1].

Комментарий. Здесь под $f_1(k)$ и $f_2(k)$ подразумевается мера множества $\mathbb{N} \setminus (k, k)$ с условием на попарное непересечение прогрессий в представлении и без него соответственно.

Лемма 3. *Используя $n \in \mathbb{N}$ арифметических прогрессий невозможно накрыть подряд 2^n натуральных чисел, не накрыв при этом весь последующий числовой ряд.*
Доказательство.

Доказательство леммы приведено в [2] как теорема 3.

Лемма 4. *Пусть $k \in \mathbb{N}$, $x \leq k$ и $k = p_1^{a_1} \cdot p_2^{a_2} \dots p_{s(k)}^{a_{s(k)}}$ — разложение числа k на простые множители. Тогда*

$$\mu(\mathbb{N} \setminus (x, k)) = a_1(p_1 - 1) + a_2(p_2 - 1) + \dots + a_{s(k)}(p_{s(k)} - 1).$$

Доказательство.

При $x = k$ утверждение совпадает с леммой 2, в которой f_1 обозначает то же, что и в наших обозначениях μ, а f_2 — такую же оценку на минимальное число прогрессий в представлении, но уже без требования на попарное непересечение. Если же $x < k$, то верхняя оценка (конструктив) получается отсечением от конструкции из [1] некоторой начальной части, что, конечно же, никак не влияет на общее количество прогрессий и их попарное непересечение. Нижняя оценка тоже получается теми же самыми рассуждениями, что и в [1], а именно построением опорного семейства необходимой мощности, причем само опорное семейство отличается от использованного в [1] лишь сдвигом на константу. ■

Лемма 5. *Пусть $k, m \in \mathbb{N}$ и $2^k < m < 2^{k+1}$. Тогда*

$$f(m) \geq k + 1$$

и равенство достигается только при $m = 2^{k-1} * 3$ или $m = 2^{k-3} * 3^2$.
Доказательство.

Заметим, в первую очередь, что

$$f(m) = a_1(p_1 - 1) + a_2(p_2 - 1) + \dots + a_{s(m)}(p_{s(m)} - 1),$$

где

$$m = p_1^{a_1} \cdot p_2^{a_2} \dots p_{s(m)}^{a_{s(m)}} \tag{5}$$

— разложение числа m на простые множители. Заменим в этом разложении каждое вхождение простого числа $p > 2$ на 2^{p-1}. Значение функции f, очевидно не изменится, а число увеличится. Если в разложении (5) есть простое число $p > 3$, то число увеличится хотя бы в два раза, так как

$$2^{p-1} \geq 2p.$$

Аналогично, число увеличится хотя бы в два раза и в случае, когда в (5) есть хотя бы три тройки, так как

$$2^2 * 2^2 * 2^2 \geq 2 * 3^3.$$

Пусть мы в итоге получили новое число $n = 2^l$. Так как $n > m > 2^k$, то $l \geq k + 1$. Значит

$$f(m) = f(n) = f(2^l) = l \geq k + 1. \tag{6}$$

При этом, неравенство (6) может превратиться в равенство только если n меньше $2m$. Как уже отмечалось выше, это возможно лишь для случаев $m = 2^{k-1} * 3$ или $m = 2^{k-3} * 3^2$ (С учетом ограничения $2^k < m < 2^{k+1}$). Осталось заметить, что для этих значений равенство действительно выполняется:

$$f(2^{k-1} \cdot 3) = (k-1) \cdot 1 + 2 = k + 1,$$
$$f(2^{k-3} \cdot 3^2) = (k-3) \cdot 1 + 2 \cdot 2 = k + 1.$$

Лемма доказана. ■

Лемма 6. *Пусть $m \in \mathbb{N}$, $2^k < m \leq 2^{k+1}$. Тогда в минимальном представлении множества $\mathbb{N} \setminus \{m\}$ непересекающимися арифметическими прогрессиями шаг одной из них будет обязательно равен 2^{k+1}, $2^{k-1} * 3$ или $2^{k-3} * 3^2$.*

Доказательство.

Рассмотрим множество

$$H := \{1, 2, \ldots, m-1\}.$$

В нем хотя бы 2^k чисел. Из леммы 3 следует, что для его накрытия нужно использовать хотя бы $k+1$ прогрессий. По теореме 1 (доказательство теоремы 1 см. ниже; оно использует только уже доказанную выше лемму 1) в минимальном представлении множества $\mathbb{N} \setminus \{m\}$ будет ровно $k+2$ прогрессий. Кроме того, хотя бы одна из этих прогрессий не содержит точек из H, так как иначе они накрывали бы, начиная с некоторого момента, весь ряд и не накрывали при этом точку m. Инвертацией прогрессий в обратную сторону получили бы (как и в доказательстве теоремы 1) противоречие с леммой 3. Объединяя полученные результаты, заключаем, что $k+1$ прогрессий начинаются в H и только одна прогрессия начинается в числе $x > m$. Выясним, каким будет шаг t этой прогрессии. Если он меньше m, то

$$H \cap (x, t) \neq \emptyset.$$

Это противоречит попарному непересечению прогрессий в минимальном представлении, так как пересечение прогрессий — всегда бесконечное множество. Если шаг t больше 2^{k+1}, то остальные $k+1$ прогрессий накрывают подряд как минимум 2^{k+1} чисел и не накрывают при этом (x,t). Это противоречило бы лемме 3. Значит

$$m \le t \le 2^{k+1}.$$

Если $t = 2^{k+1}$, то утверждение леммы доказано. В противном случае возможность представить оставшееся множество

$$(\mathbb{N} \setminus \{m\}) \setminus (x,t)$$

объединением $k+1$ прогрессий влечет за собой и возможность представить множество

$$\mathbb{N} \setminus (t,t)$$

объединением $k+1$ прогрессий. Но тогда

$$f(t) \le k+1.$$

Мы знаем, что

$$2^k < m \le t < 2^{k+1}.$$

По лемме 5 получаем

$$f(t) \ge k+1.$$

Значит $f(t) = k+1$. Опять же по лемме 5 отсюда получаем, что тогда $t = 2^{k-1} * 3$ или $t = 2^{k-3} * 3^2$. Лемма доказана. ■

Лемма 7. *Допустим, что* $(a,b),(c,d) \in \mathbb{P}^+$, $(a,b) \cap (c,d) = \emptyset$ *и пусть* $(e,f) \subset (a,b) \cup (c,d)$ *— зигзаг. Пусть, кроме того, множество* $(a,b) \cup (c,d)$ *несжимаемо. Тогда* f *— нечетное число.*
Доказательство.

Прежде всего, заметим, что если $e \in (a,b)$, то $e+f \in (c,d)$, так как иначе $(e,f) \subset (a,b)$, а это противоречит определению зигзага. Аналогично, если $e \in (c,d)$, то $e+f \in (a,b)$. Не ограничивая общности, считаем далее, что $e \in (a,b)$. Повторяя приведенное рассуждение для $e+f$ и $e+2f$, получаем в итоге, что

$$(e,2f) \in (a,b),\ (e+f,2f) \in (c,d).$$

Это свойство показывает, почему (e, f) названо словом *зигзаг*. Отсюда следует, что

$$2f = bx = dy \tag{7}$$

для некоторых $x, y \in \mathbb{N}$. При этом, число x нечетно, так как иначе f делилось бы на b и тогда бы $e + f \in (a, b)$. А это невозможно, ведь $e + f \in (c, d)$. Аналогично, число y нечетно. Значит, в силу (7), b, d — четные числа. Так как множество $(a, b) \cup (c, d)$ несжимаемо, то числа a и c разной четности. Это, в свою очередь, означает, что f нечетно. ■

Лемма 8. *Пусть множество $\mathbb{N}\setminus(n, n)$ представлено в виде объединения $f(n)$ непересекающихся арифметических прогрессий. Тогда каждый шаг этих прогрессий является делителем числа n.*
Доказательство.

Обозначим множество этих $f(n)$ прогрессий за I. Из [1] известно, что существует опорное семейство из $f(n)$ точек внутри $(1, 2, \ldots, n-1)$, то есть такое множество, для которого любая проходящая через пару его элементов арифметическая прогрессия будет пересекаться с (n, n). При этом, если заменить любую из точек x семейства на точку $x + n$, то новое множество точек по прежнему будет опорным. Это означает, что прогрессия из I, которая проходит через x будет проходить и через $x+n$. Значит ее шаг будет делителем числа n. Осталось заметить, что любая прогрессия из I проходит через какую-то из точек опорного семейства. ■

Доказательство основных утверждений

Теорема 1. *Пусть $k, m \in \mathbb{N}$, $2^k < m \leq 2^{k+1}$. Тогда $\mu(\mathbb{N}\setminus\{m\}) = k+2$.*
Доказательство.

Докажем, что $\mu(\mathbb{N} \setminus \{m\}) \leq k + 2$. Так как

$$\mathbb{N} \setminus \{m\} = (\mathbb{N} \setminus (m, 2^{k+1})) \cup (m + 2^{k+1}, 2^{k+1})$$

и

$$m \leq 2^{k+1},$$

то по лемме 1 получаем

$$\mu(\mathbb{N} \setminus \{m\}) \leq k + 1 + 1 = k + 2.$$

Докажем нижнюю оценку $\mu(\mathbb{N} \setminus \{m\}) \geq k+2$. Из [2] известно, что, используя n прогрессий, невозможно накрыть подряд более чем $2^n - 1$ чисел, не накрыв при этом весь ряд. Поэтому на накрытие множества $\{1, 2, \ldots, m-1\}$ потребуется хотя бы $k+1$ прогрессий. Если бы этого количества прогрессий было достаточно для накрытия множества $\mathbb{N} \setminus \{m\}$, то применяя рассуждения, аналогичные приводимым в [2], получили бы, что, используя $k+1$ прогрессий, удается, начиная с некоторого момента, накрыть сколь угодно длинную последовательность идущих подряд чисел и при этом не накрыть весь ряд. Для этого надо лишь инвертировать все $k+1$ прогрессий справа налево. Их продолжения влево от их начал все еще не будут содержать точку m, так как все эти начала меньше m. Получаем противоречие с леммой 3. ■

Теорема 2. *Пусть* $a, b, c \in \mathbb{N}$. *Если* $a \in (b,c)^-$ *и* $2^{k-1} < \frac{b-a}{c} \leq 2^k$, *то* $\mu(\{a\} \cup (b,c)) = k+1$, *а если* $a \notin (b,c)^-$, *то* $\mu(\{a\} \cup (b,c)) = \infty$.
Доказательство.

Докажем сначала простую часть теоремы, когда $a \notin (b,c)^-$. В этом случае в любом представлении множества $\{a\} \cup (b,c)$ все прогрессии, начиная с некоторого момента, лежали бы в (b,c). Значит такие прогрессии изначально лежали бы в $(b,c)^+$ и не накрыли бы точку $\{a\}$. Поэтому $\mu(\{a\} \cup (b,c)) = \infty$.

Пусть теперь $a \in (b,c)^-$. Тогда сделаем с нашим множеством преобразование, переводящее $(b,c)^+$ в $(1,1)$. После него множество $\{a\} \cup (b,c)$ превратится в

$$\{1\} \cup \left(1 + \frac{b-a}{c}, 1\right). \tag{8}$$

При этом, мера μ при таком преобразовании, очевидно, не меняется. Обозначим $1 + \frac{b-a}{c}$ за d. Рассмотрим теперь какое-нибудь минимальное представление прогрессиями множества (8). Пусть точку $\{1\}$ в нем накрывает прогрессия с шагом n. Тогда $n \geq d-1$ и

$$\mu(\{1\} \cup (d,1)) = 1 + \mu((d,1) \setminus (n+1, n)).$$

Из леммы 4 известно, что

$$\mu((d,1) \setminus (n+1,n)) = a_1(p_1 - 1) + a_2(p_2 - 1) + \ldots + a_{s(k)}(p_{s(k)} - 1), \tag{9}$$

где

$$n = p_1^{a_1} \cdot p_2^{a_2} \ldots p_{s(k)}^{a_{s(k)}} \tag{10}$$

— разложение числа n на простые множители. В силу минимальности покрытия получаем отсюда, что n доставляет минимум значения $\mu((d,1) \setminus (l+1,l))$ по всем $l \geq d-1$. Заметим, что если в произведении (10) заменить простое число p_1 на 2^{p_1-1}, то произведение не уменьшится, а сумма (8) не изменится. Значит можно считать, что n — степень двойки. Чем больше степень, тем больше значение суммы (8). Значит, n — минимальная степень двойки, удовлетворяющая условию $n \geq d-1$. Но $d = 1 + \frac{b-a}{c}$. Из условия $2^{k-1} < \frac{b-a}{c} \leq 2^k$ следует, что $n = k$. Значит

$$\mu(\{1\} \cup (d,1)) = k+1.$$

Теорема доказана. ■

Теорема 3. *Пусть $a,b,c,d,e \in \mathbb{N}, k \in \mathbb{N}_0$ и пусть $T = (b,c) \sqcup (d,e)$ — пара прогрессий общего положения.*
Если $a \notin (b,c)^- \cup (d,e)^-$ то $\mu(\{a\} \cup T) = \infty$.
Если $a \in (b,c)^-,\ 2^{k-1} < \frac{b-a}{c} \leq 2^k$, то $\mu(\{a\} \cup T) = k+2$.
Если $a \in (d,e)^-,\ 2^{k-1} < \frac{d-a}{e} \leq 2^k$, то $\mu(\{a\} \cup T) = k+2$.
Доказательство.

Докажем сначала первую часть теоремы, когда $a \notin (b,c)^- \cup (d,e)^-$. Если бы множество $\{a\} \cup T$ можно было представить конечным объединением арифметических прогрессий, то все они, начиная с некоторого момента, лежали бы в $(b,c) \cup (d,e)$. Значит такие прогрессии изначально лежали бы в $(b,c)^+ \cup (d,e)^+$ и не накрыли бы точку $\{a\}$. Поэтому $\mu(\{a\} \cup T) = \infty$.

Докажем вторую часть теоремы, где $a \in (b,c)^-$ и $2^{k-1} < \frac{b-a}{c} \leq 2^k$. Считаем, что множество T несжимаемо, так как иначе множество $\{a\} \cup T$ тоже можно было сжать и свести задачу к случаю, когда T несжимаемо. Рассмотрим оптимальное разбиение множества $\{a\} \cup T$. Пересекаем каждую прогрессию в разбиении с $(b,c)^+$. Таким образом мы получили разбиение $a \cup (b,c)$. Из теоремы 2 мы знаем, что для этого потребуется не меньше $k+1$ прогрессий. Опять же, используя теорему 2, получаем, что мы сможем представить $\{a\} \cup T$ объединением $k+1$ прогрессии для $a \cup (b,c)$ и прогрессии (d,e). Получится $k+2$ прогрессии.

Покажем, что $k+1$ прогрессий для этого не хватит. Если бы это было не так, то оптимальное разбиение получалось бы из оптимального разбиения множества $a \cup (b,c)$ с помощью расширения некоторых прогрессий разбиения до зигзага в $\{a\} \cup T$. Из леммы 7 получаем, что зигзаги могут возникнуть только на прогрессиях с нечетным внутри $a \cup (b,c)$ шагом. Из леммы 6 выводим, что у одной из прогрессий в разбиении $a \cup (b,c)$

шаг равен 2^{k+1}, $2^{k-1} * 3$ или $2^{k-3} * 3^2$. В первом случае, из леммы 8, получаем, что все прогрессии разбиения, кроме, возможно, прогрессии с единичным шагом, имеют четные шаги и не порождают зигзаг. А прогрессия с единичным шагом возникает, только если $a = b - c$, а в этом случае $\mu(\{a\} \cup T) = 2$ — получаем оценку из утверждения теоремы при $k = 0$.

Теперь рассмотрим случай шага $2^{k-1} \cdot 3$. единственные нечетные делители этого числа — 1 или 3, но случай с 1 был только что разобран выше, а при использовании прогрессии с шагом 3 в оптимальном разбиении множества $a \cup (b, c)$ нужно сразу же использовать тогда и вторую прогрессию с шагом 3. Однако, построение зигзага на любой из них дает прогрессии частного положения, а построить зигзаги на обеих нельзя, так как мы не получим прогрессию (d, e).

Для случая с шагом $2^{k-3} \cdot 3^2$ аналогично возникают только три нечетных делителя — $1, 3$ или 9. Случай с 1 уже разобран, случай с 3 разбирается так же, как и для шага $2^{k-1} \cdot 3$. Разберем теперь случай с шагом 9. В этом случае все наши прогрессии делятся на три непересекающихся класса с остатками по модулю 3, и, так как это оптимальное представление, два из этих классов заняты одной прогрессией с шагом 3, но тогда единственным таким разбиением будет разбиение с шагами $3, 3, 6, 12...$, и шага 9 среди них нет.

В каждом из возможных случаев показали, что попытка взять оптимальное разбиение множества $a \cup (b, c)$ и, расширив его зигзагами, получить разбиение для множества $\{a\} \cup T$ приводит нас лишь к случаю прогрессий частного положения.

Случай, когда $a \in (d, e)^-$ и $2^{k-1} < \frac{d-a}{e} \leq 2^k$, разбирается аналогично. ■

Теорема 4. *Пусть $a, b, c, d, e \in \mathbb{N}, k \in \mathbb{N}_0$ и пусть $T = (b, c) \sqcup (d, e)$ — пара прогрессий общего положения, $a \in T$.*
Если $a \in (b, c)$, $b + c \cdot (2^k - 1) < a \leq b + c \cdot (2^{k+1} - 1)$, то $\mu(T \setminus \{a\}) = k + 3$.
Если $a \in (d, e)$, $d + e \cdot (2^k - 1) < a \leq d + e \cdot (2^{k+1} - 1)$, то $\mu(T \setminus \{a\}) = k + 3$.
Иначе $\mu(T \setminus \{a\}) = 2$.
Доказательство.

Ход доказательства повторяет доказательство предыдущей теоремы. Пусть $a \in (b, c)$, $b + c \cdot (2^k - 1) < a \leq b + c \cdot (2^{k+1} - 1)$. Тогда, по теореме 1, мы умеем разбивать множество $(b, c) \setminus a$ в объединение $k + 2$ прогрессий и в итоге получаем оценку $k + 3$. Предполагаем, от противного, что это можно сделать меньшим числом прогрессий. Тогда каждую из них пере-

секаем с (b, c) и получаем разбиение множества $(b, c) \setminus a$, а, по теореме 1, для этого надо хотя бы $k + 2$ прогрессий. Значит их ровно $k + 2$, причем полученное разбиение для $(b, c) \setminus a$ оптимально. По лемме 6, в нем есть прогрессия с шагом 2^{k+1}, $2^{k-1} \cdot 3$ или $2^{k-3} \cdot 3^2$. Каждый из этих вариантов, по лемме 7, снова приводит нас к расширению множества зигзагами на прогрессиях с шагом $1, 3$ или 9. Но все они приводят нас лишь к парам прогрессий частного положения.

Случай, когда $a \in (d, e)$, $d + e \cdot (2^k - 1) < a \leq d + e \cdot (2^{k+1} - 1)$, разбирается аналогично.

Наконец, если оба эти случая не реализуются, то $a = b$ или $a = d$. В любом случае, двух прогрессий для представления множества $T \setminus \{a\}$ хватит. А одна прогрессия может возникнуть лишь в случае, когда, с точностью до сжатия, $a = b = 1$, $d = 4$, $c = e = 2$. Это прогрессии частного положения. ■

Заключение

В работе было найдено точное значение для меры множеств, получаемых из множеств размерности 1 при добавлении или удалении из них одного числа. Также эти значения были найдены для множеств размерности 2, образуемых парой прогрессий общего положения. В этом случае оптимальное разбиение достигается тривиально с использованием оптимальных разбиений для случаев с множествами размерности 1. Необходимо отметить, что для множеств, образуемых парой прогрессий частного положения, данное утверждение уже неверно. Этот класс случаев требует дополнительного исследования.

Список литературы

[1] Э. С. Айрапетов, П. С. Дергач. *О прогрессивном разбиении некоторых подмножеств натурального ряда.* Интеллектуальные системы, 2015. Т.19, вып. 3, М., Сс. 79-86.

[2] М. А. Коххарова. *О максимальном покрытии начала натурального ряда с ограничениями.* Дипломная работа. Филиал МГУ им. М.В. Ломоносова в г. Ташкент, 2017.

Сведения об авторах

Дергач Петр Сергеевич, Dergach Pyotr Sergeevich
Младший научный сотрудник МГУ имени М. В. Ломоносова в городе Москве;
адрес: Россия, г. Москва, 125565, Ленинградское ш., 88-19;
тел. моб.: +79037189288;
e-mail: dergachpes@mail.ru.

Булгаков Леон Русланович, Bulgakov Leon Ruslanovich
Студент магистратуры НИЯУ МИФИ;
адрес: Россия, г. Москва, 115409, Каширское ш., 31;
тел. моб.: +79647751528;
e-mail: leon1bulgakov@gmail.com.

About the dimension difference of periodic subsets of the natural series

Dergach P.S., Bulgakov L.R.

This work is devoted to describing the change in the dimension of periodic subsets of the natural series with seemingly insignificant operations like *removal/addition* to the set of one number. The case is investigated when the dimension of the initial set is equal to 1 or 2. By the dimension of the set is meant the minimum number of disjoint arithmetic progressions that give this set in the union. For sets of dimension 2 the result is obtained only in cases of pairs of general position progressions. In this paper we give the results on how the dimension changes depending on whether the number x is *deleted/added* to.

Keywords: arithmetic progression, natural series, progressive set.

Короткие единичные диагностические тесты для контактных схем при обрывах и замыканиях контактов

Попков К.А.[1]

Доказано, что почти любую булеву функцию от n переменных можно реализовать неизбыточной двухполюсной контактной схемой, допускающей единичный диагностический тест длины 8 относительно обрывов и замыканий контактов.

Ключевые слова: контактная схема, булева функция, обрыв контакта, замыкание контакта, единичный диагностический тест.

1. Введение

В работе рассматривается задача синтеза легкотестируемых двухполюсных контактных схем [1], реализующих заданные булевы функции. (Слово «двухполюсная» в дальнейшем будем опускать.) Логический подход к тестированию контактных схем предложен С.В. Яблонским и И.А. Чегис в [2]. Представим, что имеется контактная схема S, реализующая булеву функцию $f(\tilde{x}^n)$, где $\tilde{x}^n = (x_1, \ldots, x_n)$. Под воздействием некоторого источника неисправностей один или несколько контактов схемы S могут перейти в неисправное состояние. В качестве неисправностей контактов обычно рассматриваются их обрывы и замыкания. При обрыве контакта проводимость между его концами становится тождественно нулевой, а при замыкании — тождественно единичной. В результате схема S вместо исходной функции $f(\tilde{x}^n)$ будет реализовывать некоторую булеву функцию $g(\tilde{x}^n)$, вообще говоря, отличную от f. Все такие функции $g(\tilde{x}^n)$,

[1] *Попков Кирилл Андреевич* — кандидат физико-математических наук, научный сотрудник Института прикладной математики им. М.В. Келдыша РАН, e-mail: kirill-formulist@mail.ru.

Popkov Kirill Andreevich — Candidate of Physical and Mathematical Sciences, Researcher, Keldysh Institute of Applied Mathematics RAS.

получающиеся при всевозможных допустимых для рассматриваемой задачи неисправностях контактов схемы S, называются *функциями неисправности* данной схемы.

Введём следующие определения [3, 4, 5]. *Проверяющим тестом* для схемы S называется такое множество T наборов значений переменных $x_1, \ldots, x_n$, что для любой отличной от $f(\tilde{x}^n)$ функции неисправности $g(\tilde{x}^n)$ схемы S в T найдётся набор $\tilde{\sigma}$, на котором $f(\tilde{\sigma}) \neq g(\tilde{\sigma})$. *Диагностическим тестом* для схемы S называется такое множество T наборов значений переменных $x_1, \ldots, x_n$, что T является проверяющим тестом и, кроме того, для любых двух различных функций неисправности $g_1(\tilde{x}^n)$ и $g_2(\tilde{x}^n)$ схемы S в T найдётся набор $\tilde{\sigma}$, на котором $g_1(\tilde{\sigma}) \neq g_2(\tilde{\sigma})$. Число наборов в T называется *длиной* теста. В качестве тривиального диагностического (и проверяющего) теста длины 2^n для схемы S всегда можно взять множество, состоящее из всех двоичных наборов длины n. Тест называется *полным*, если в схеме могут быть неисправны сколько угодно контактов, и *единичным*, если в схеме может быть неисправен только один контакт. Единичные тесты обычно рассматривают для *неизбыточных схем* (см. [5, с. 110–111]), т.е. для таких схем, в которых любая допустимая неисправность любого одного контакта приводит к функции неисправности, отличной от исходной функции, реализуемой данной схемой.

В дальнейшем будем считать, что в схемах могут происходить как обрывы, так и замыкания контактов независимо друг от друга.

Пусть множество T является единичным диагностическим тестом (ЕДТ) для некоторой контактной схемы S. Введём следующие обозначения: $D_{\text{ЕД}}(T)$ — длина теста T; $D_{\text{ЕД}}(S) = \min D_{\text{ЕД}}(T)$, где минимум берётся по всем ЕДТ T для контактной схемы S; $D_{\text{ЕД}}(f) = \min D_{\text{ЕД}}(S)$, где минимум берётся по всем неизбыточным контактным схемам S, реализующим функцию f; $D_{\text{ЕД}}(n) = \max D_{\text{ЕД}}(f)$, где максимум берётся по всем булевым функциям f от n переменных. Функция $D_{\text{ЕД}}(n)$ называется *функцией Шеннона* длины ЕДТ. По аналогии с функциями $D_{\text{ЕД}}$ можно ввести функции $D_{\text{ЕП}}$, $D_{\text{ПП}}$ и $D_{\text{ПД}}$ для соответственно единичного проверяющего теста (ЕПТ), полного проверяющего и полного диагностического тестов, зависящие от T, от S, от f и от n (в определениях функций $D_{\text{ПП}}(f)$ и $D_{\text{ПД}}(f)$ не предполагается неизбыточности схем). Так, например, $D_{\text{ПП}}(n)$ — функция Шеннона длины полного проверяющего теста.

Будем говорить, что некоторое свойство выполняется *для почти всех булевых функций от n переменных*, если отношение числа булевых

функций от n переменных, для которых это свойство не выполняется, к числу всех булевых функций от n переменных (т.е. к 2^{2^n}) стремится к нулю при $n \to \infty$.

Перечислим основные результаты, касающиеся задачи синтеза легкотестируемых контактных схем при рассматриваемых неисправностях. В [5, с. 113, теорема 9] с использованием идей С.В. Яблонского установлено, что функция $D_{\text{ЕД}}(n)$ асимптотически не превосходит $\frac{2^{n+1}}{n}$. Х.А. Мадатян в [6] доказал равенство $D_{\text{ПД}}(n) = 2^n$. Н.П. Редькин в [7] получил оценку $D_{\text{ПП}}(n) \leqslant \frac{15}{16} \cdot 2^n$. Из утверждения 2) теоремы 1 работы [8] следует, что для любой булевой функции $f(\tilde{x}^n)$ вида $\varphi(\tilde{x}^{n-1}) \oplus x^n$, где $\varphi(\tilde{x}^{n-1})$ — произвольная неконстантная булева функция, существует неизбыточная контактная схема, содержащая, помимо переменных из множества $\{x_1, \ldots, x_n\}$, дополнительную входную переменную x_0, допускающая ЕПТ длины $4n + 8$ и реализующая булеву функцию, не зависящую существенно от переменной x_0 и равную функции $f(\tilde{x}^n)$. В работе [9], в частности, доказано (теорема 4), что для любой булевой функции $f(\tilde{x}^n)$ существует неизбыточная контактная схема, содержащая не более пяти дополнительных входных переменных, допускающая ЕПТ длины не более 35 и реализующая такую булеву функцию, что $f(\tilde{x}^n)$ получается из неё подстановкой вместо этих дополнительных переменных некоторых булевых констант. В [10] установлено, что для почти всех булевых функций f от n переменных $D_{\text{ЕП}}(f) = 4$, а также описаны все булевы функции f, для которых величина $D_{\text{ЕП}}(f)$ равна 0, 1, 2 и 3.

В настоящей работе будет описан достаточно обширный класс булевых функций f, для которых $D_{\text{ЕД}}(f) \leqslant 8$ (теорема 2); будет показано, что в этом классе содержатся почти все булевы функции от n переменных (теорема 3).

2. Основные результаты

Теорема 1. *Пусть для булевой функции $h(\tilde{x}^n)$, $n \geqslant 3$, существуют такой индекс $i \in \{1, \ldots, n\}$ и такие булевы константы $\sigma_1, \ldots, \sigma_n$, что*

$$h(\tilde{\pi}_1) = h(\tilde{\pi}_2) = 1,$$
$$h(\tilde{\pi}_3) = h(\tilde{\pi}_4) = 0,$$

где

$$\begin{aligned}
\tilde{\pi}_1 &= (\sigma_1, \ldots, \sigma_n),\\
\tilde{\pi}_2 &= (\overline{\sigma}_1, \ldots, \overline{\sigma}_n),\\
\tilde{\pi}_3 &= (\sigma_1, \ldots, \sigma_{i-1}, \overline{\sigma}_i, \sigma_{i+1}, \ldots, \sigma_n),\\
\tilde{\pi}_4 &= (\overline{\sigma}_1, \ldots, \overline{\sigma}_{i-1}, \sigma_i, \overline{\sigma}_{i+1}, \ldots, \overline{\sigma}_n).
\end{aligned}$$

Тогда функцию $h(\tilde{x}^n)$ можно реализовать неизбыточной контактной схемой, допускающей ЕПТ $\{\tilde{\pi}_1, \tilde{\pi}_2, \tilde{\pi}_3, \tilde{\pi}_4\}$.

Теорема 1 несложным образом вытекает из доказательства неравенства $D(f) \leqslant 4$ в теореме 2 работы [10], а также из перехода от множества T_1 к множеству T_2 в доказательстве леммы 1 той же работы.

Всюду ниже считаем, что запись вида $\alpha_{i_1}, \ldots, \alpha_{i_2}$ или $\overline{\alpha}_{i_1}, \ldots, \overline{\alpha}_{i_2}$ при $i_1 > i_2$ обозначает пустую строку.

Теорема 2. *Пусть для булевой функции $f(\tilde{x}^n)$, $n \geqslant 3$, существуют такие индексы $j, s \in \{1, \ldots, n\}$, $j < s$, и такие булевы константы $\alpha_1, \ldots, \alpha_n$, что*

$$f(\tilde{\rho}_1) = f(\tilde{\rho}_2) = f(\tilde{\rho}_3) = f(\tilde{\rho}_4) = 1, \tag{1}$$
$$f(\tilde{\rho}_5) = f(\tilde{\rho}_6) = f(\tilde{\rho}_7) = f(\tilde{\rho}_8) = 0, \tag{2}$$

где

$$\begin{aligned}
\tilde{\rho}_1 &= (\alpha_1, \ldots, \alpha_n),\\
\tilde{\rho}_2 &= (\overline{\alpha}_1, \ldots, \overline{\alpha}_n),\\
\tilde{\rho}_3 &= (\alpha_1, \ldots, \alpha_{j-1}, \overline{\alpha}_j, \alpha_{j+1}, \ldots, \alpha_{s-1}, \overline{\alpha}_s, \alpha_{s+1}, \ldots, \alpha_n),\\
\tilde{\rho}_4 &= (\overline{\alpha}_1, \ldots, \overline{\alpha}_{j-1}, \alpha_j, \overline{\alpha}_{j+1}, \ldots, \overline{\alpha}_{s-1}, \alpha_s, \overline{\alpha}_{s+1}, \ldots, \overline{\alpha}_n),\\
\tilde{\rho}_5 &= (\alpha_1, \ldots, \alpha_{j-1}, \overline{\alpha}_j, \alpha_{j+1}, \ldots, \alpha_n),\\
\tilde{\rho}_6 &= (\overline{\alpha}_1, \ldots, \overline{\alpha}_{j-1}, \alpha_j, \overline{\alpha}_{j+1}, \ldots, \overline{\alpha}_n),\\
\tilde{\rho}_7 &= (\alpha_1, \ldots, \alpha_{s-1}, \overline{\alpha}_s, \alpha_{s+1}, \ldots, \alpha_n),\\
\tilde{\rho}_8 &= (\overline{\alpha}_1, \ldots, \overline{\alpha}_{s-1}, \alpha_s, \overline{\alpha}_{s+1}, \ldots, \overline{\alpha}_n).
\end{aligned}$$

Тогда $D_{\text{ЕД}}(f) \leqslant 8$.

Доказательство. Пусть i_1, i_2, i_3, i_4 — произвольные попарно различные индексы от 1 до 8. Обозначим через $I_{i_1 i_2 i_3 i_4}(\tilde{x}^n)$ булеву функцию, равную

1 на наборах $\tilde{\rho}_{i_1}, \tilde{\rho}_{i_2}, \tilde{\rho}_{i_3}, \tilde{\rho}_{i_4}$ и равную 0 на всех остальных наборах длины n. Положим

$$f_1 = f \oplus I_{3478}, \tag{3}$$
$$f_2 = f \oplus I_{3456}, \tag{4}$$
$$f_3 = f \oplus I_{1278}, \tag{5}$$
$$f_4 = f \oplus I_{1256}. \tag{6}$$

Составим таблицу значений функций $f_1(\tilde{x}^n), f_2(\tilde{x}^n), f_3(\tilde{x}^n), f_4(\tilde{x}^n)$ на наборах $\tilde{\rho}_1, \tilde{\rho}_2, \tilde{\rho}_3, \tilde{\rho}_4, \tilde{\rho}_5, \tilde{\rho}_6, \tilde{\rho}_7, \tilde{\rho}_8$. Для этого используем соотношения (1)–(6): например,

$$f_1(\tilde{\rho}_1) = f(\tilde{\rho}_1) \oplus I_{3478}(\tilde{\rho}_1) = 1 \oplus 0 = 1.$$

	$\tilde{\rho}_1$	$\tilde{\rho}_2$	$\tilde{\rho}_3$	$\tilde{\rho}_4$	$\tilde{\rho}_5$	$\tilde{\rho}_6$	$\tilde{\rho}_7$	$\tilde{\rho}_8$
f_1	1	1	0	0	0	0	1	1
f_2	1	1	0	0	1	1	0	0
f_3	0	0	1	1	0	0	1	1
f_4	0	0	1	1	1	1	0	0

Применяя теорему 1 при $h = f_1$, $i = j$ и $(\sigma_1, \ldots, \sigma_n) = (\alpha_1, \ldots, \alpha_n)$, получаем, что функцию $f_1(\tilde{x}^n)$ можно реализовать неизбыточной контактной схемой S_1, допускающей ЕПТ $T_1 = \{\tilde{\rho}_1, \tilde{\rho}_2, \tilde{\rho}_5, \tilde{\rho}_6\}$.

Применяя теорему 1 при $h = f_2$, $i = s$ и $(\sigma_1, \ldots, \sigma_n) = (\alpha_1, \ldots, \alpha_n)$, получаем, что функцию $f_2(\tilde{x}^n)$ можно реализовать неизбыточной контактной схемой S_2, допускающей ЕПТ $T_2 = \{\tilde{\rho}_1, \tilde{\rho}_2, \tilde{\rho}_7, \tilde{\rho}_8\}$.

Применяя теорему 1 при $h = f_3$, $i = s$ и $(\sigma_1, \ldots, \sigma_n) = (\alpha_1, \ldots, \alpha_{j-1}, \overline{\alpha}_j, \alpha_{j+1}, \ldots, \alpha_{s-1}, \overline{\alpha}_s, \alpha_{s+1}, \ldots, \alpha_n)$, получаем, что функцию $f_3(\tilde{x}^n)$ можно реализовать неизбыточной контактной схемой S_3, допускающей ЕПТ $T_3 = \{\tilde{\rho}_3, \tilde{\rho}_4, \tilde{\rho}_5, \tilde{\rho}_6\}$.

Применяя теорему 1 при $h = f_4$, $i = j$ и $(\sigma_1, \ldots, \sigma_n) = (\alpha_1, \ldots, \alpha_{j-1}, \overline{\alpha}_j, \alpha_{j+1}, \ldots, \alpha_{s-1}, \overline{\alpha}_s, \alpha_{s+1}, \ldots, \alpha_n)$, получаем, что функцию $f_4(\tilde{x}^n)$ можно реализовать неизбыточной контактной схемой S_4, допускающей ЕПТ $T_4 = \{\tilde{\rho}_3, \tilde{\rho}_4, \tilde{\rho}_7, \tilde{\rho}_8\}$.

Соединим последовательно схемы S_1 и S_2, а также схемы S_3 и S_4. Полученные две схемы соединим параллельно. Обозначим итоговую схему через S. Докажем, что данная схема при отсутствии в ней неисправностей реализует функцию $f(\tilde{x}^n)$. Для любого $i \in \{1, \ldots, 8\}$ на наборе

$\tilde{\rho}_i$ она выдаёт значение $f_1(\tilde{\rho}_i)f_2(\tilde{\rho}_i) \vee f_3(\tilde{\rho}_i)f_4(\tilde{\rho}_i)$, которое, как нетрудно видеть из таблицы, равно

$$\begin{cases} 1, & \text{если } i \leqslant 4, \\ 0, & \text{если } i \geqslant 5, \end{cases}$$

и в силу (1), (2) равно $f(\tilde{\rho}_i)$. В то же время, на любом двоичном наборе $\tilde{\tau}$ длины n, не принадлежащем множеству $T = \{\tilde{\rho}_1, \tilde{\rho}_2, \tilde{\rho}_3, \tilde{\rho}_4, \tilde{\rho}_5, \tilde{\rho}_6, \tilde{\rho}_7, \tilde{\rho}_8\}$, подсхема S_m, $m = 1, 2, 3, 4$, выдаёт значение

$$f_m(\tilde{\tau}) = f(\tilde{\tau}) \oplus I_{\dots}(\tilde{\tau}) = f(\tilde{\tau}) \oplus 0 = f(\tilde{\tau}), \tag{7}$$

а схема S — значение

$$f_1(\tilde{\tau})f_2(\tilde{\tau}) \vee f_3(\tilde{\tau})f_4(\tilde{\tau}) = f(\tilde{\tau}).$$

Тем самым доказано, что схема S реализует функцию $f(\tilde{x}^n)$.

Докажем теперь, что данная схема неизбыточна, а множество T является для неё ЕДТ. Предположим, что имеет место неисправность (обрыв или замыкание) произвольного одного контакта схемы S. Пусть при этом неисправный контакт содержится в подсхеме S_m для некоторого $m \in \{1, 2, 3, 4\}$. Множество T_m является ЕПТ для неизбыточной схемы S_m, поэтому существует такой набор $\tilde{\rho} \in T_m$, что при указанной неисправности подсхема S_m выдаёт на наборе $\tilde{\rho}$ «неправильное» значение $\overline{f}_m(\tilde{\rho})$. Вместе с тем, для любого $m' \in \{1, 2, 3, 4\} \setminus \{m\}$ подсхема $S_{m'}$ выдаёт на этом наборе «правильное» значение $f_{m'}(\tilde{\rho})$. Рассмотрим четыре случая.

1. Пусть $m = 1$. Тогда $\tilde{\rho} \in \{\tilde{\rho}_1, \tilde{\rho}_2, \tilde{\rho}_5, \tilde{\rho}_6\}$ и схема S выдаёт на наборе $\tilde{\rho}$ значение $\overline{f}_1(\tilde{\rho})f_2(\tilde{\rho}) \vee f_3(\tilde{\rho})f_4(\tilde{\rho})$. Просматривая 1-й, 2-й, 5-й и 6-й столбцы таблицы, получаем, что это значение равно

$$\begin{cases} 0 \cdot 1 \vee 0 \cdot 0 = 0, & \text{если } \tilde{\rho} \in \{\tilde{\rho}_1, \tilde{\rho}_2\}, \\ 1 \cdot 1 \vee 0 \cdot 1 = 1, & \text{если } \tilde{\rho} \in \{\tilde{\rho}_5, \tilde{\rho}_6\}, \end{cases}$$

и в силу (1), (2) равно $\overline{f}(\tilde{\rho})$.

2. Пусть $m = 2$. Тогда $\tilde{\rho} \in \{\tilde{\rho}_1, \tilde{\rho}_2, \tilde{\rho}_7, \tilde{\rho}_8\}$ и схема S выдаёт на наборе $\tilde{\rho}$ значение $f_1(\tilde{\rho})\overline{f}_2(\tilde{\rho}) \vee f_3(\tilde{\rho})f_4(\tilde{\rho})$. Просматривая 1-й, 2-й, 7-й и 8-й столбцы таблицы, получаем, что это значение равно

$$\begin{cases} 1 \cdot 0 \vee 0 \cdot 0 = 0, & \text{если } \tilde{\rho} \in \{\tilde{\rho}_1, \tilde{\rho}_2\}, \\ 1 \cdot 1 \vee 1 \cdot 0 = 1, & \text{если } \tilde{\rho} \in \{\tilde{\rho}_7, \tilde{\rho}_8\}, \end{cases}$$

и в силу (1), (2) равно $\overline{f}(\tilde{\rho})$.

3. Пусть $m = 3$. Тогда $\tilde{\rho} \in \{\tilde{\rho}_3, \tilde{\rho}_4, \tilde{\rho}_5, \tilde{\rho}_6\}$ и схема S выдаёт на наборе $\tilde{\rho}$ значение $f_1(\tilde{\rho})f_2(\tilde{\rho}) \vee \overline{f}_3(\tilde{\rho})f_4(\tilde{\rho})$. Просматривая 3-й, 4-й, 5-й и 6-й столбцы таблицы, получаем, что это значение равно

$$\begin{cases} 0 \cdot 0 \vee 0 \cdot 1 = 0, & \text{если } \tilde{\rho} \in \{\tilde{\rho}_3, \tilde{\rho}_4\}, \\ 0 \cdot 1 \vee 1 \cdot 1 = 1, & \text{если } \tilde{\rho} \in \{\tilde{\rho}_5, \tilde{\rho}_6\}, \end{cases}$$

и в силу (1), (2) равно $\overline{f}(\tilde{\rho})$.

4. Пусть $m = 4$. Тогда $\tilde{\rho} \in \{\tilde{\rho}_3, \tilde{\rho}_4, \tilde{\rho}_7, \tilde{\rho}_8\}$ и схема S выдаёт на наборе $\tilde{\rho}$ значение $f_1(\tilde{\rho})f_2(\tilde{\rho}) \vee f_3(\tilde{\rho})\overline{f}_4(\tilde{\rho})$. Просматривая 3-й, 4-й, 7-й и 8-й столбцы таблицы, получаем, что это значение равно

$$\begin{cases} 0 \cdot 0 \vee 1 \cdot 0 = 0, & \text{если } \tilde{\rho} \in \{\tilde{\rho}_3, \tilde{\rho}_4\}, \\ 1 \cdot 0 \vee 1 \cdot 1 = 1, & \text{если } \tilde{\rho} \in \{\tilde{\rho}_7, \tilde{\rho}_8\}, \end{cases}$$

и в силу (1), (2) равно $\overline{f}(\tilde{\rho})$.

В каждом из случаев 1–4 установлено, что на наборе $\tilde{\rho} \in T_m \subset T$ схема S выдаёт значение $\overline{f}(\tilde{\rho})$, отличное от значения $f(\tilde{\rho})$, выдаваемого этой схемой при отсутствии в ней неисправностей. Тем самым доказано, что схема S неизбыточна, а множество T является для неё ЕПТ.

Далее, на любом двоичном наборе $\tilde{\tau}$ длины n, не принадлежащем множеству T, при рассматриваемой неисправности подсхема S_m выдаёт некоторое значение $a_{\tilde{\tau}} \in \{0, 1\}$, а подсхема $S_{m'}$ — «правильное» значение $f_{m'}(\tilde{\tau})$, равное $f(\tilde{\tau})$ в силу (7), для любого $m' \in \{1, 2, 3, 4\} \setminus \{m\}$. Тогда схема S выдаёт на данном наборе значение

$$a_{\tilde{\tau}} f(\tilde{\tau}) \vee f(\tilde{\tau}) f(\tilde{\tau}) = f(\tilde{\tau}).$$

Таким образом, значение любой функции неисправности схемы S на любом двоичном наборе длины n, не принадлежащем множеству T, совпадает со значением функции f на этом наборе. Поэтому любые две различные функции неисправности данной схемы могут различаться только на наборах из множества T и обязаны различаться хотя бы на одном наборе из этого множества, откуда вытекает, что T — ЕДТ для схемы S.

В итоге получаем, что функцию $f(\tilde{x}^n)$ можно реализовать неизбыточной контактной схемой S, допускающей ЕДТ T длины 8. Следовательно,

$$D_{\text{ЕД}}(f) \leqslant D_{\text{ЕД}}(S) \leqslant D_{\text{ЕД}}(T) = 8.$$

Теорема 2 доказана. □

Теорема 3. *Для почти всех булевых функций f от n переменных справедливо неравенство $D_{\text{ЕД}}(f) \leqslant 8$.*

Доказательство. Пусть $n \geqslant 4$. Нетрудно заметить, что множество всех двоичных наборов длины n можно разбить на 2^{n-3} попарно непересекающихся упорядоченных восьмёрок наборов

$$\begin{aligned} U_{\alpha_1,\ldots,\alpha_{n-3}} = ((&\alpha_1,\ldots,\alpha_{n-3},1,1,1),(\overline{\alpha}_1,\ldots,\overline{\alpha}_{n-3},0,0,0),\\ &(\alpha_1,\ldots,\alpha_{n-3},0,0,1),(\overline{\alpha}_1,\ldots,\overline{\alpha}_{n-3},1,1,0),\\ &(\alpha_1,\ldots,\alpha_{n-3},0,1,1),(\overline{\alpha}_1,\ldots,\overline{\alpha}_{n-3},1,0,0),\\ &(\alpha_1,\ldots,\alpha_{n-3},1,0,1),(\overline{\alpha}_1,\ldots,\overline{\alpha}_{n-3},0,1,0)), \end{aligned}$$

где $\alpha_1,\ldots,\alpha_{n-3}$ — булевы константы, образующие все возможные комбинации (для краткости обозначим данные восемь наборов через $\tilde{\rho}_1^{\tilde{\alpha}}, \tilde{\rho}_2^{\tilde{\alpha}},\ldots,$ $\tilde{\rho}_8^{\tilde{\alpha}}$ соответственно, где $\tilde{\alpha} = (\alpha_1,\ldots,\alpha_{n-3})$). Пусть F_n — множество булевых функций от n переменных, не принимающих ни на одной из этих восьмёрок наборов ни одну из восьмёрок значений $(1,1,1,1,0,0,0,0)$, $(0,0,0,0,1,1,1,1)$. Любая булева функция $f(\tilde{x}^n)$, не принадлежащая множеству F_n, удовлетворяет либо соотношениям

$$\begin{gathered} f(\tilde{\rho}_1^{\tilde{\alpha}}) = f(\tilde{\rho}_2^{\tilde{\alpha}}) = f(\tilde{\rho}_3^{\tilde{\alpha}}) = f(\tilde{\rho}_4^{\tilde{\alpha}}) = 1,\\ f(\tilde{\rho}_5^{\tilde{\alpha}}) = f(\tilde{\rho}_6^{\tilde{\alpha}}) = f(\tilde{\rho}_7^{\tilde{\alpha}}) = f(\tilde{\rho}_8^{\tilde{\alpha}}) = 0, \end{gathered}$$

либо соотношениям

$$\begin{gathered} f(\tilde{\rho}_1^{\tilde{\alpha}}) = f(\tilde{\rho}_2^{\tilde{\alpha}}) = f(\tilde{\rho}_3^{\tilde{\alpha}}) = f(\tilde{\rho}_4^{\tilde{\alpha}}) = 0,\\ f(\tilde{\rho}_5^{\tilde{\alpha}}) = f(\tilde{\rho}_6^{\tilde{\alpha}}) = f(\tilde{\rho}_7^{\tilde{\alpha}}) = f(\tilde{\rho}_8^{\tilde{\alpha}}) = 1 \end{gathered}$$

для некоторых $\alpha_1,\ldots,\alpha_{n-3} \in \{0,1\}$. Тогда функция $f(\tilde{x}^n)$ удовлетворяет условиям теоремы 2 при $j = n-3$, $s = n-2$, $\alpha_{n-1} = \alpha_n = 1$ и некотором $\alpha_{n-2} \in \{0,1\}$, поэтому $D_{\text{ЕД}}(f) \leqslant 8$.

Найдём мощность множества F_n. На каждой из 2^{n-3} восьмёрок наборов $U_{\alpha_1,\ldots,\alpha_{n-3}}$ любая функция из этого множества может принимать любую из $2^8 - 2 = 254$ восьмёрок значений. Следовательно, $|F_n| = 254^{2^{n-3}}$. Тогда

$$\frac{|F_n|}{2^{2^n}} = \frac{254^{2^{n-3}}}{256^{2^{n-3}}} = \left(\frac{127}{128}\right)^{2^{n-3}} \to 0 \ \ (n \to \infty).$$

Таким образом, отношение числа булевых функций из множества F_n к общему числу булевых функций от n переменных стремится к 0 при $n \to \infty$. Выше было показано, что для любой булевой функции $f(\tilde{x}^n)$, не принадлежащей множеству F_n, выполнено неравенство $D_{\text{ЕД}}(f) \leqslant 8$, откуда следует справедливость теоремы 3. $\square$

3. Заключение

Описанный в работе для почти всех булевых функций от n переменных метод синтеза реализующих их контактных схем, допускающих единичные диагностические тесты длины 8 относительно обрывов и замыканий контактов, ввиду малой длины тестов, а значит, малого времени, необходимого для тестирования таких схем, может найти практическое применение. Вместе с тем, пока не удалось получить единой верхней оценки длины минимального единичного диагностического теста для **любой** булевой функции от n переменных, т.е. верхней оценки величины $D_{\text{ЕД}}(n)$, улучшающей оценку $D_{\text{ЕД}}(n) \lesssim \frac{2^{n+1}}{n}$ из [5, с. 113, теорема 9]. Представляет интерес также изучение возможностей реализации всех или почти булевых функций от n переменных контактными схемами, допускающими короткие проверяющие или диагностические тесты относительно обрывов и замыканий не более k контактов, где $k \geqslant 2$ — заданное натуральное число.

Список литературы

[1] Лупанов О.Б., *Асимптотические оценки сложности управляющих систем*, Изд-во Моск. ун-та, Москва, 1984, 138 с.

[2] Чегис И.А., Яблонский С.В., "Логические способы контроля работы электрических схем", *Труды МИАН*, **51** (1958), 270–360

[3] Яблонский С.В., "Надёжность и контроль управляющих систем", *Материалы Всесоюзного семинара по дискретной математике и её приложениям (Москва, 31 января–2 февраля 1984 г.)*, Изд-во Моск. ун-та, Москва, 1986, 7–12.

[4] Яблонский С.В., "Некоторые вопросы надёжности и контроля управляющих систем", *Математические вопросы кибернетики. Вып. 1*, Наука, Москва, 1988, 5–25.

[5] Редькин Н.П., *Надёжность и диагностика схем*, Изд-во Моск. ун-та, Москва, 1992, 192 с.

[6] Мадатян Х.А., "Полный тест для бесповторных контактных схем", *Проблемы кибернетики. Вып. 23*, Наука, Москва, 1970, 103–118.

[7] Редькин Н.П., "О полных проверяющих тестах для контактных схем", *Методы дискретного анализа в исследовании экстремальных структур. Вып. 39*, Изд-во ИМ СО АН СССР, Новосибирск, 1983, 80–87.

[8] Романов Д.С., "О синтезе контактных схем, допускающих короткие проверяющие тесты", *Учёные записки Казанского университета. Физико-математические науки*, **156**:3 (2014), 110–115.

[9] Романов Д.С., Романова Е.Ю., “О единичных проверяющих тестах для схем переключательного типа”, *Известия высших учебных заведений. Поволжский регион. Физико-математические науки*, 2015, № 1, 5–23.

[10] Попков К.А., “Короткие единичные проверяющие тесты для контактных схем при обрывах и замыканиях контактов”, *Интеллектуальные системы. Теория и приложения*, **23**:3 (2019), 97–130.

Short single diagnostic tests for contact circuits under breaks and closures of contacts

Popkov K.A.

We prove that almost any Boolean function on n variables can be implemented by an irredundant two-pole contact circuit allowing a single diagnostic test of length 8 regarding breaks and closures of contacts.

Keywords: contact circuit, Boolean function, contact break, contact closure, single diagnostic test.

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 звание (если имеется), контактные телефоны (с кодом города и страны), e-mail, почтовый адрес с индексом города (домашний или служебный), для студентов и аспирантов надо указывать Ф.И.О. научного руководителя полностью.

3. Для оформления списка литературы используется пакет amsbib.sty (см. статью с примерами использования: http://intsysjournal.org/templates/amsbib.pdf).

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: 18.03.2020
Дата выхода: 25.03.2020
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).