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

Интеллектуальные Системы.

Теория и приложения

ТОМ **30** ВЫПУСК 1 * **2026**

МОСКВА

УДК 519.95; 007:159.955
ББК 32.81

ISSN 2411–4448
Издаётся 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.
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. Входит в дополнительный перечень научных изданий, в которых могут быть опубликованы результаты диссертаций, защищаемых в МГУ.

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@intsysmagazine.ru

*) Прежнее название журнала: «Интеллектуальные системы».

ОГЛАВЛЕНИЕ

Часть 1
Общие проблемы теории интеллектуальных систем

Архитектура упрощённой системы автоматического вывода на основе подхода А. С. Подколзина

А. П. Анненков*

Решение формализуемых математических задач с помощью ЭВМ остаётся актуальной проблемой, для которой существует несколько принципиально различных подходов. Значительный вклад в развитие данного направления внесла монография А. С. Подколзина, в которой предложена оригинальная архитектура решателя задач, основанная на логическом выводе с помощью правил преобразования термов. В данной статье описывается архитектура упрощённой системы, реализующей ключевые идеи этого подхода: механизм переключения внимания через уровни, организацию базы правил и представление контекста с метаданными.

Ключевые слова: решатель математических задач, автоматический вывод, логические процессы, логический язык, логическая формализация задач.

1. Введение

Автоматизация решения формализуемых математических задач остаётся актуальной проблемой на стыке математической логики, теоретической информатики и компьютерной алгебры. Существующие подходы варьируются от систем символьных вычислений и систем автоматических доказательств теорем до алгоритмов, основанных на предобученных языковых моделях, демонстрирующих значительные успехи в обработке математических текстов. Однако создание прозрачного, расширяемого решателя, основанного на принципах формального вывода и понятной стратегии поиска решения, сохраняет свою востребованность как для теоретических исследований, так и для практических приложений.

Значительный вклад в это направление был сделан А. С. Подколзиным, предложившим оригинальную архитектуру решателя, основанную на

* *Анненков Александр Петрович* — аспирант каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: `just.std@yandex.ru`, ORCID: 0009-0002-9332-1569.

Annenkov Alexander Petrovich — graduate student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

логическом выводе с использованием правил преобразования термов. Его подход, изложенный в монографии «Компьютерное моделирование логических процессов», отличается строгой формализацией, механизмом управляемого переключения внимания и идеей ранжирования правил по уровням, что позволяет эффективно управлять пространством поиска решений. Не умаляя теоретической глубины и полноты предложенного решения, стоит отметить, что оригинальная система представляет собой сложный комплекс, что создаёт трудности для её изучения, модификации и интеграции с другими средствами.

В связи с этим целью данной работы является создание и описание упрощённой архитектуры, построенной на тех же ключевых принципах, но реализованной в виде компактной экспериментальной системы. В статье рассматриваются основные проектные решения: структуры данных, алгоритм основного цикла работы и механизмы управления выводом. Особое внимание уделяется созданию целостной минимально достаточной системы, способной решать базовые алгебраические задачи. Предлагаемая реализация может служить основой для учебных моделей, прототипов и дальнейших исследований в области автоматизированного логического вывода.

Структура статьи: в разделе 2 вводятся основные понятия и формализация задач; раздел 3 описывает механизм переключения внимания; раздел 4 посвящён архитектуре системы, включая представление правил, термов и контекста; раздел 5 детализирует цикл работы алгоритма; раздел 6 кратко описывает языки системы; в приложении приводится пошаговая демонстрация работы системы на примере решения линейного уравнения, а также псевдокод основных алгоритмов системы.

2. Основные понятия

Начнём описание системы с формализации некоторых понятий и описания класса решаемых задач. В основе принципа работы лежит получение новых термов с помощью применения правил к полученным ранее термам (либо условиям задачи). *Терм*, в контексте данной работы, будем понимать в классическом индуктивном определении:

1. Переменные и константы (0-арные функциональные символы) являются термами.
2. Если f — n-арный функциональный символ, а $t_1, t_2, \ldots, t_n$ — термы, то $f(t_1, t_2, \ldots, t_n)$ — также терм.

2.1. Формализация задач

Задачей в данной работе будем называть пару, состоящую из *целевой установки* и *списка условий*.

Целевая установка — определяет тип задачи и терм, который является предметом поиска или преобразования. В системе целевая установка также представляется термом.

Система поддерживает три типа целевых установок:

1. **Find**(x) — описать значение неизвестного терма x через известные термы.
2. **Prove**(s) — вывести терм s из условий задачи (доказать истинность s в случае истинности условий).
3. **Transform**(s) — преобразовать терм к требуемому виду или упростить терм.

Список условий представляет собой конечный набор термов, считающихся истинными в контексте решаемой задачи.

Пример 1. Рассмотрим задачу решения уравнения $4x + 7 = 2x + 1$. Требуется найти значение x, удовлетворяющее равенству. Её формализация:

- целевая установка: **Find**(x);
- список условий: $\{4x + 7 = 2x + 1\}$.

2.2. Правила преобразования

Правило, помимо классического вида ***шаблон*** $\Rightarrow$ ***вывод*** содержит список термов, ограничивающих условия его применения. Последние будем называть *ограничениями*. Переменные, входящие в шаблон правила, будем называть *параметрами*.

Применение правила к терму происходит по следующему алгоритму:

1. Найти подстановку параметров, которая обращает шаблон правила в подтерм обрабатываемого терма.
2. Проверить истинность ограничений после применения такой подстановки. В случае неудачи попробовать снова с другой подстановкой.
3. Заменить подтерм, соответствующий шаблону правила, на терм, получаемый применением подстановки к выводу правила.

Следует отметить, что результат такого алгоритма неоднозначен. Для одного набора значений параметров возможно несколько различных вхождений шаблона в обрабатываемый терм. В то же время возможно, что подходящий набор таких значений может быть не единственным. Таким образом, результатом применения правила к терму считается множество (возможно пустое) термов.

Пример 2. Рассмотрим правило переноса $a = b \Rightarrow a - b = 0$ с ограничением, что b не является синтаксически нулём. Применим его к терму $4x + 7 = 2x + 1$ из примера 1:

1. Подстановка: $\{a \mapsto 4x + 7, b \mapsto 2x + 1\}$.
2. Ограничение проверяется тривиальным сравнением термов.
3. Новый терм: $(4x + 7) - (2x + 1) = 0$.

2.3. Процесс обучения

Система представляет собой непосредственно алгоритм, который будем называть *ядром*, и набор правил, записанных на специальном языке. *Обучением* системы мы будем называть добавление в систему новых правил, либо внесение изменений в существующие для расширения класса решаемых задач, а также уменьшения времени, затрачиваемого системой на поиск ответа.

3. Переключение внимания

Важным вопросом при реализации ядра системы является порядок обхода термов. Классические подходы, такие как обход в глубину или обход в ширину, в данной задаче приводят к зацикливанию алгоритма на отдельных термах задачи, в то время как важные для получения ответа термы не будут рассмотрены вовсе.

Одно решение для этой задачи было предложено А. С. Подколзиным. Его суть состоит в ранжировании правил по уровням в зависимости от их *общеупотребимости* — доли задач, в которых срабатывание правила существенно ускоряет решение, либо является обязательным для получения ответа.

Описываемая система наследует решение, описанное А. С. Подколзиным: на этапе обучения каждому правилу приписывается целое неотрицательное число, называемое *уровнем правила* — L_r. При решении задачи каждому терму приписывается целое неотрицательное число, называемое

уровнем терма L_t. Вновь сгенерированному терму приписывается нулевой уровень. В процессе решения система пытается применять к терму уровня L_t правила уровня $L_r = L_t$. Если применение всех подходящих правил этого уровня не дало новых термов, уровень терма увеличивается на единицу. Заметим, что в системе А. С. Подколзина допускается, чтобы правило срабатывало на нескольких уровнях. Однако эта возможность используется там нечасто, поэтому в рассматриваемой здесь системе в целях простоты правило привязывается к конкретному уровню, и применение правила на нескольких уровнях при необходимости можно смоделировать созданием нескольких копий одного правила для всех уровней, на которых его предполагается пытаться применить.

В начале каждой итерации выбирается один из термов с наименьшим уровнем. Если таких термов несколько, выбирается тот, что был получен раньше. В случае, если все термы имеют больший уровень, чем максимальный существующий уровень правила, алгоритм завершается. Если к моменту завершения не был получен ответ, результатом считается специальный флаг, сигнализирующий о том, что решение не найдено.

4. Архитектура системы

В данном разделе описываются ключевые компоненты экспериментальной системы, их организация и взаимодействие. Основное внимание уделяется структурам данных, используемым для представления информации: правила, термы, контекст задачи и механизмы их обработки.

4.1. Система подзадач

Часто решение задачи требует промежуточных шагов, выходящих за рамки целевой установки самой задачи. Например, требуется упрощение терма или доказательство вспомогательного утверждения в рамках задачи на поиск значения неизвестной. В таких случаях используется механизм подзадач. *Подзадача* — это вспомогательная задача, которая решается независимо, и результат которой используется в основной задаче.

Подзадачи используют те же целевые установки, что были описаны ранее:

1. **Transform** — преобразование терма используется для упрощения всех новых термов в контексте задачи.
2. **Prove** — доказательство утверждения может быть использовано для проверки истинности ограничений в процессе применения правила.

3. **Find** — поиск значения переменной может быть использован для рассмотрения случаев с последующим объединением ответов.

Механизм подзадач также позволяет изолировать контекст решения задачи от попадания в него вспомогательных термов, полученных в ходе решения подзадачи. А также — значительно упрощает процесс отладки, позволяя быстрее находить места, в которых у системы возникли затруднения.

Подзадачи могут создавать свои подзадачи, образуя иерархию с ограничением глубины вложенности. Результаты решения подзадач кэшируются, чтобы избежать повторного решения одинаковых подзадач. Добавление подзадачи в кэш производится до начала её решения, при этом специальным образом отмечается, что решение задачи ещё не получено. При попытке рекурсивно решить такую подзадачу вместо ответа будет получен флаг, сигнализирующий о том, что решение не найдено.

4.2. Представление правил

Полный перебор всех правил для каждого терма приводит к большому количеству вычислений. В связи с этим актуальной становится задача разработки эффективных алгоритмов и структур данных, позволяющих на ранних этапах идентифицировать и исключать заведомо безуспешные попытки применения правил, что существенно повышает производительность системы в целом.

4.2.1. Привязка правил к функциональным символам

Наблюдения показывают, что значительная часть правил оказывается отсеяна на этапе сопоставления терма с шаблоном. Это наводит на мысль о поиске некоторого инварианта, который позволит сократить количество перебираемых правил на как можно более раннем этапе.

Таким инвариантом выступает требование присутствия всех функциональных символов шаблона в обрабатываемом терме как необходимое условие успешной унификации. Опираясь на этот факт, А. С. Подколзин предлагает привязывать правила к символу из шаблона. При обработке терма система выбирает только те правила, ключевые функциональные символы которых присутствуют в данном терме, что существенно сокращает пространство перебора. Выбор символа осуществляется пользователем при добавлении правила.

В описываемой системе дополнительно производится проверка полного вхождения символов шаблона в терм до попытки сопоставления.

4.2.2. Разбиение правил по уровням

Как было описано в разделе 3, система применяет к терму лишь те правила, чей уровень соответствует уровню терма $L_r = L_t$. При поиске подходящих правил разумно ограничиться лишь множеством правил этого уровня.

4.2.3. База правил

Для эффективного поиска подходящих правил система организует их в специальную структуру — *базу правил*. Учитывая оптимизации, описанные в подразделах 4.2.1 и 4.2.2, база правил строится как двухуровневая структура. На первом уровне правила распределяются по уровням срабатывания L_r, соответствующим уровням термов. На втором уровне для каждого значения L_r создаётся отображение, связывающее каждый функциональный символ со списком правил данного уровня, привязанных к данному символу.

Алгоритм поиска правил для терма t уровня L_t в базе правил $\mathcal{R}$ состоит из следующих шагов:

1. **Выбор уровня:** Получаем $\mathcal{R}[L_t]$ — подмножество правил, соответствующее уровню терма L_t.
2. **Выбор правил по символам:** Для каждого функционального символа s, присутствующего в терме t, извлекаем из $\mathcal{R}[L_t]$ список правил $\mathcal{R}_s[L_t]$, привязанных к символу s. Объединяем все полученные списки в множество кандидатов $\mathcal{R}_1[t]$.

$$\mathcal{R}_1[t] = \bigcup_{s \in t} \mathcal{R}_s[L_t]$$

3. **Фильтрация по входящим символам:** Из $\mathcal{R}_1[t]$ исключаем те правила, шаблон которых содержит хотя бы один функциональный символ, отсутствующий в t. В результате остаются только правила $\mathcal{R}_2[t]$, все символы шаблона которых присутствуют в обрабатываемом терме.

Такая организация позволяет на этапе сбора кандидатов быстро отбросить заведомо неприменимые правила (те, что не содержат ни одного общего символа с термом), а на этапе финальной фильтрации — уточнить результат, проверив полное вхождение символов шаблона.

Псевдокод данного алгоритма приведён в приложении В.

4.3. Представление контекста задачи

Термы используются системой для представления правил, целевых установок, условий и выведенных утверждений. Термы, представляющие утверждения, а также термы, описывающие целевую установку, образуют рабочее множество[1] $\mathbb{T}$. Все термы сопровождаются дополнительной информацией, которую будем называть *метаданными*. Метаданные включают в себя уровень терма, историю его вывода, списки применённых и заблокированных правил и другую служебную информацию.

4.3.1. Иммутабельность термов

Все термы в $\mathbb{T}$ неизменяемы (иммутабельны) — если применение правила порождает новый терм, отсутствующий в $\mathbb{T}$, он добавляется в $\mathbb{T}$, не затрагивая исходный. Это свойство касается только самого терма, но не его метаданных.

Для случаев, когда необходимо изменить исходный терм, предусмотрена его *замена*: терм помечается как неактивный и больше не выбирается для обработки в основном цикле. В текущей версии системы такой подход используется для замены терма после его упрощения с помощью подзадачи **Transform** или после применения к терму правила с атрибутом `replace`.

4.3.2. Представление терма

В системе используется классическое представление терма в виде древовидной структуры. Терм строится из трёх типов атомарных компонентов:

1. **Константы** — числовые значения.
2. **Функциональные символы** — n-арные операторы из заранее определённого множества для целых неотрицательных n. Узлы дерева представляют собой функциональные символы, аргументами которых являются дочерние термы.
3. **Идентификаторы** — строковые имена, не являющиеся константами или функциональными символами.

Рациональные числа представляются обыкновенными дробями, иррациональные числа — как результаты операций (например, $\sqrt{2}$), а широко

[1] Формально $\mathbb{T}$ — это мультимножество, где могут быть два синтаксически совпадающих терма разного происхождения: один выведенный из условий, другой — из целевой установки задачи.

применяемые константы (например, π, e) — как 0-арные функциональные символы.

Идентификаторы в системе бывают двух типов: *переменные* и *параметры*. Тип идентификатора зависит от роли терма в системе, таким образом все идентификаторы конкретного терма являются либо переменными, либо параметрами. Переменные используются в выводимых термах, в то время как параметры используются в термах правил.

4.3.3. Нормализация терма

Одной из самых трудоемких задач при работе системы остаётся сопоставление терма с образцом правила. Отчасти это связано с неоднозначным представлением семантики выражения с помощью терма. Так, например, ассоциативные операции допускают произвольную расстановку скобок, а коммутативные — возможность менять местами аргументы.

Чтобы сократить перебор возможных вариантов, система пытается привести все термы к некоторому каноническому виду. Так вложенные ассоциативные операции преобразуются в одну операцию от нескольких аргументов, аргументы коммутативных операций упорядочиваются. Помимо этого функциональный символ может иметь дополнительные процедуры приведения, описанные внутри системы.

Данная процедура носит название *нормализация*. Некоторые примеры нормализации:

- Приведение подобных слагаемых: $x + 2 * x + 3 \to 3 * x + 3$.
- Вычисление константных выражений: $2 + 3 * 5 \to 17$.
- Упорядочивание: $3 + x \to x + 3$.
- Удаление лишних скобок и операций: $1 * (x + y) + z \to +(x, y, z)$.

Важно различать нормализацию и упрощение термов. Нормализация — это строгое синтаксическое приведение к канонической форме, которое гарантированно сохраняет семантическую эквивалентность. Упрощение же, реализуемое через подзадачу **Transform**, может применять более сложные, контекстно-зависимые преобразования (например, вычисление производной или раскрытие скобок по формуле), подразумевающие семантическую корректность, обеспечиваемую корректностью введённых в систему правил преобразования.

Однако существуют ситуации, когда нормализация может быть нежелательна. Например, правило может специально представлять терм в неканонической форме, чтобы обеспечить срабатывание другого правила.

В таких случаях для конкретного терма или класса преобразований нормализация может быть временно отключена или ограничена с помощью соответствующих атрибутов правил.

В рамках процесса обучения системы (определённого в разделе 2.3) процедура нормализации остаётся неизменной. Её модификация не входит в стандартный цикл обучения и требует внесения изменений на уровне ядра системы.

Пример 3. Нормализация терма $(4x + 7) - (2x + 1) = 0$ из примера 2:

- Устранение вложенности сложения: $+(4 * x, 7, -1 * 2 * x, -1 * 1) = 0$.
- Арифметические вычисления: $+(4 * x, 7, -2 * x, -1) = 0$.
- Приведение подобных: $+((4 - 2) * x, (7 - 1)) = 0$.
- Арифметические вычисления: $+(2 * x, 6) = 0$.
- Итоговый нормализованный терм: $2x + 6 = 0$.

4.3.4. Отслеживание вывода

Каждый терм в системе сопровождается метаданными, которые фиксируют историю его происхождения — так называемый *источник вывода*. Эта информация позволяет восстановить полную цепочку рассуждений, приведшую к получению терма.

Система различает три основных типа источников:

- **Condition** — терм входит в условие задачи. Такие термы образуют начальное множество, от которого начинается процесс вывода.
- **RuleApplication** — терм получен в результате применения правила преобразования. В метаданных сохраняются идентификатор применённого правила, используемая подстановка параметров, а также доказательства всех ограничений, проверенных при применении.
- **Transform** — терм является результатом упрощения другого терма через подзадачу типа **Transform** (см. раздел 4.1). Метаданные содержат ссылку на контекст подзадачи, что позволяет при необходимости воспроизвести процесс упрощения.

Хотя источник вывода не влияет непосредственно на процесс применения правил во время решения (все термы равноправны с точки зрения алгоритма вывода), он выполняет несколько важных функций:

1. **Реконструкция решения**: после успешного завершения задачи система может восстановить полную последовательность шагов, демонстрируя логику рассуждений.

2. **Диагностика**: при возникновении затруднений у системы в процессе решения новой задачи анализ источников вывода помогает идентифицировать проблемные участки в цепочке рассуждений.
3. **Сбор статистики**: информация о частоте применения различных правил и эффективности преобразований служит ценным материалом для дальнейшего обучения и оптимизации системы.
4. **Верификация**: наличие полной истории вывода позволяет проводить апостериорную проверку корректности решения.

4.3.5. Блокировка правил

В процессе обучения системы могут возникнуть ситуации, когда к результату применения правила нежелательно применение другого правила. Например, после получения явного значения переменной нет смысла переносить правую часть влево и решать ещё раз линейное уравнение с тем же результатом.

Для предотвращения подобных ситуаций правило может хранить ссылки на правила, которые следует заблокировать для выведенного терма. Пометки о блокировке сохраняются в метаданных нового терма и используются в качестве фильтров на этапе выбора нового правила для этого терма.

4.3.6. Термы-подстановки

В процессе решения задач иногда возникает необходимость получения следствия одновременно из нескольких термов. Рассмотрение всех возможных подмножеств термов для такого преобразования является вычислительно трудоёмкой задачей.

Важным частным случаем такой проблемы являются термы вида *переменная = значение*, называемые *подстановками*. Обычно это равенство необходимо подставить в другие термы, содержащие эту переменную.

Обработка подстановок осуществляется в два этапа.

1. При обнаружении подстановки в метаданных соответствующего терма устанавливается специальный флаг.
2. Применение подстановки к другому терму производится с помощью специального правила лишь в тот момент, когда этот терм окажется выбран процедурой переключения внимания (см. раздел 3).

4.3.7. Краткий список метаданных терма

Ниже приведён сводный список ключевых метаданных терма t, используемых системой, часть из которых уже была рассмотрена в предыдущих разделах:

1. **Уровень терма** L_t — определяет приоритет обработки и ограничивает множество применяемых правил (см. раздел 3).
2. **Источник вывода** — указывает, как был получен терм (см. раздел 4.3.4).
3. **Фильтры правил**:
   - $\mathcal{R}_{\text{applied}}(t)$ — правила, уже применявшиеся к терму t (во избежание повторной обработки).
   - $\mathcal{R}_{\text{blocked}}(t)$ — правила, заблокированные для применения к t (см. раздел 4.3.5).
4. **Кэшированные данные**:
   - Symbols(t) — множество функциональных символов, входящих в терм. Вычисляется однократно и используется при поиске правил в базе (см. раздел 4.2.3).
5. **Флаги состояния**:
   - Simplified — терм уже был упрощён через подзадачу **Transform**.
   - IsSubstitution — терм имеет вид $x = v$ и может быть использован для подстановки (см. раздел 4.3.6).
   - IsAlternativeTarget — терм является альтернативной целевой установкой, например, равносильным утверждением в задаче на доказательство. Подробнее процесс получения альтернативных целевых установок описывается в разделе 5.1.
   - IsReplaced — терм был исключён из рассмотрения после применения правила с атрибутом `replace` или преобразования через подзадачу **Transform**.

Для удобства введём обозначение для множества активных термов:

$$\mathbb{T}_{\text{active}} = \{t \in \mathbb{T} \mid t.\text{IsReplaced} = \text{false}\}.$$

4.4. Инициализация и начальные преобразования

Перед запуском основного цикла (раздел 5) система преобразует исходную постановку задачи во внутреннее представление.

1. **Обработка условий задачи**.
   Каждое условие преобразуется в терм и нормализуется согласно разделу 4.3.3. Терму назначаются метаданные: уровень $L_t = 0$, источник $\text{Source}(t) = \text{Condition}$, вычисляется множество символов $\text{Symbols}(t)$, списки применённых и заблокированных правил инициализируются как пустые. Полученные термы образуют множество $\mathbb{T}_{\text{cond}}$.
2. **Обработка целевой установки**.
   Из целевой установки извлекается внутренний терм t_{goal} и преобразуется аналогично термам из условий: не будет противоречием считать источником целевой установки обобщённое условие задачи. Поэтому метаданные инициализируются следующим образом: уровень $L_t = 0$, источник $\text{Source}(t) = \text{Condition}$. Дополнительно терм целевой установки получает флаг состояния $\text{IsAlternativeTarget} = \text{true}$. Образуется множество $\mathbb{T}_{\text{goal}} = \{t_{\text{goal}}\}$.
3. **Инициализация структур данных**.
   Начальное рабочее множество термов получается объединением мультимножеств:
   $$\mathbb{T} = \mathbb{T}_{\text{cond}} \uplus \mathbb{T}_{\text{goal}}.$$
   Создаётся пустой кэш решённых подзадач из раздела 4.1.
4. **Инициализация служебных структур**.
   Вычисляется максимальный уровень правил $L_{\max}$. Сохраняется информация о типе исходной цели.

После выполнения этих шагов система переходит к основному циклу работы.

5. Цикл работы алгоритма

Решение задачи происходит в цикле, на каждой итерации которого выполняются следующие шаги:

1. **Выбор терма** — из множества активных термов $\mathbb{T}_{\text{active}} \subseteq \mathbb{T}$ для обработки выбирается первый в порядке получения терм t с наименьшим уровнем. Это может быть терм из условия задачи, ранее

выведенный терм или терм целевой установки (подробнее о преобразованиях терма целевой установки в разделе 5.1).

2. **Упрощение** — попытка упростить терм через подзадачу **Transform**, если ранее терм не был упрощён (подробнее про подзадачи в разделе 4.1). Для этого вызывается подзадача с целевой установкой **Transform**(t), за исключением случая, когда исходная задача уже имеет целевую установку вида **Transform**$(\dots)$. Исходный терм заменяется на ответ подзадачи с отметкой, исключающей повторное срабатывание.
3. **Усмотрение ответа** — проверка, является ли выбранный терм ответом на задачу (подробнее про усмотрение ответа в разделе 5.2).
4. **Создание пометок** — проверяется, может ли терм быть использован для замены переменной, как описано в 4.3.6. В случае успеха добавляется соответствующая пометка.
5. **Формирование списка правил** — из базы правил $\mathcal{R}$ выбирается список правил $\mathcal{R}_2[t]$ для терма t текущего уровня L_t, как это было описано в разделе 4.2.3. Затем происходит дополнительная фильтрация: отбрасываются правила, применённые к t ранее (раздел 4.3.7), либо заблокированные для применения к нему (раздел 4.3.5):

$$\mathcal{R}_3[t] = \mathcal{R}_2[t] \setminus \mathcal{R}_{\text{applied}}(t) \setminus \mathcal{R}_{\text{blocked}}(t)$$

6. **Применение правил** — система последовательно пытается применить каждое правило из отфильтрованного списка $\mathcal{R}_3[t]$ к терму t до первого успешного применения. Подробное описание этого процесса приведено в разделе 5.4.
7. **Обновление статуса терма** — Если в результате применения правила r с атрибутом замены был получен новый терм, исходный терм t помечается как заменённый (IsReplaced = true) и таким образом исключается из множества активных термов $\mathbb{T}_{\text{active}}$.
8. **Обновление уровня терма** — если процесс подошёл к концу списка правил, но не смог получить новых термов, уровень выбранного терма увеличивается на единицу, а алгоритм возвращается к выбору нового терма.

Цикл продолжается до одного из следующих условий:

- Найден ответ.
- Достигнут лимит итераций.
- Не удалось выбрать терм для обработки.

Псевдокод алгоритма приведён в приложении Д.

5.1. Преобразование целевого терма

Преобразуемым термом может быть терм из условия задачи, ранее выведенный терм или терм целевой установки. Это может быть полезно для упрощения доказываемого терма, либо для вывода термов, равносильных доказываемому. Термы, полученные преобразованием терма целевой установки, помечаются флагом состояния IsAlternativeTarget.

В задачах с целевой установкой вида **Prove**(...) выводится множество термов, равносильных доказываемому. Эти термы используются при усмотрении ответа (см. раздел 5.2).

Задача с целевой установкой вида **Transform**(...) подразумевает преобразование целевой установки для получения ответа.

5.2. Усмотрение ответа

Процедура *усмотрения ответа* определяет, является ли данный терм t ответом на задачу с текущей целевой установкой. Эта проверка вызывается в цикле работы системы (см. раздел 5) для каждого выбранного терма.

Система предоставляет возможность явно формировать ответ на задачу в выводе правила, что позволяет реализовать сложную логику усмотрения ответа на этапе обучения решателя.

Также присутствуют некоторые общие правила усмотрения ответа.

- **Prove**(s): терм t считается ответом, если t не является целевым термом сам по себе (t.IsAlternativeTarget = false), и синтаксически совпадает с s или с одним из термов, помеченных флагом состояния IsAlternativeTarget (см. 5.1).
- **Find**(x): терм t считается ответом, если он имеет одну из следующих форм:
  1. $x = v$, где v — известное выражение (подробнее термы, считающиеся известными, описаны в разделе 5.3).
  2. $x \in A$, где A — известное.

В отличие от других типов задач, раннее усмотрение ответа для Transform не производится. Процесс преобразования заключается в генерации цепочки термов с флагом IsAlternativeTarget. По завершении основного цикла, но перед завершением алгоритма, ответом помечается последний сгенерированный терм, имеющий этот флаг.

5.3. Концепция «Известное» (known)

Важным понятием в системе является концепция *«известное»* (known). Изначально известными считаются числовые константы. Также допускается пометка известными переменных в условии задачи, если допускается их появление в ответе (так называемые задачи с параметром).

Проверка, что терм является известным, осуществляется через **Prove**-подзадачи. Чаще всего правила выводят известность для арифметических операций над известными термами.

Концепция «известное» имеет важное значение в процедуре усмотрения ответа, а также в ограничениях для срабатывания правила. Для задач типа **Find**(x) ответ считается найденным, если получен терм вида *переменная* = *значение*, где значение известно, либо когда *переменная* $\in A$, где A — известно.

5.4. Применение правил

Применение правил к текущему терму t выполняется последовательно для каждого правила $r_i \in \mathcal{R}_3[t]$ из списка правил, подготовленных для применения к данному терму.

1. **Поиск подстановок параметров**
   Для правила r_i строится последовательность $\{p_k\}$ всевозможных подстановок параметров, обращающих шаблон r_i в терм t.
2. **Формирование гипотезы**
   Подстановка p_k применяется к выводу правила r_i, образуя новый терм $\hat{t}$. Также подстановка p_k применяется к ограничениям правила r_i для получения списка требований $\{c_m\}$, необходимых для обоснования терма $\hat{t}$. Пара $(\hat{t}, \{c_m\})$ в системе называется *гипотезой*. Термы гипотезы проходят нормализацию в процессе её формирования.
3. **Исключение дубликатов**
   Прежде чем приступать к обоснованию гипотезы, система проверяет, не встречается ли $\hat{t}$ среди выведенных ранее термов с тем же значением флага IsAlternativeTarget. В случае, если $\hat{t}$ уже был выведен, система переходит к следующей подстановке p_{k+1}, не тратя время на обоснование уже полученного утверждения.
4. **Обоснование гипотезы**
   Обоснование гипотезы осуществляется методом последовательного решения подзадач с целевой установкой **Prove**(c_m) (подробнее подзадачи описаны в разделе 4.1). Если какая-то из этих задач не

может быть решена системой, происходит переход к следующей подстановке p_{k+1}.

5. **Добавление нового терма**
   При успешном обосновании гипотезы терм $\hat{t}$ добавляется в множество выведенных термов $\mathbb{T}$ со следующими метаданными:
   - Уровень: $L_{\hat{t}} = 0$
   - Источник: **RuleApplication** с указанием применённого правила r_i и подстановки параметров p_k.
   - Список применённых правил: пустой.
   - Список заблокированных правил: правила, указанные как блокируемые в r_i.
6. **Обновление списка применённых правил**
   Если для правила r_i все допустимые подстановки параметров исчерпаны, но ни одна из них не привела к добавлению нового терма, правило помечается как применённое к терму t во избежание повторной обработки, а алгоритм переходит к следующему правилу r_{i+1}.

Правило может быть применено не только к терму t, но и к его подтерму t'. В таком случае новый терм $\hat{t}'$ получается из t методом замены подтерма t' на вывод правила r_i с применённой подстановкой p_k. Все шаги применения правил при этом проводятся аналогично.

Псевдокод алгоритма применения одного правила к терму приведён в приложении Г.

Пример 4. Рассмотрим правило для решения линейного уравнения:

$$ax + b = 0 \Rightarrow x = -b/a$$

Его ограничения: $\{a \neq 0, b$ — известно$\}$.

И покажем, как оно может быть применено к результату нормализации из примера 3: $2x + 6 = 0$.

1. Подстановка параметров: $\{a \mapsto 2, b \mapsto 6, x \mapsto x\}$. Подстановка $x \mapsto x$ может показаться бессмысленной, однако стоит помнить, что идентификатор слева является параметром, а идентификатор справа — переменной (см. 4.3.2).
2. Гипотеза: $x = -3$, требования: $2 \neq 0$, 6 — известно.
3. Принятие гипотезы, поскольку все требования — истинные утверждения.

4. Новый терм: $x = -3$.

Заметим, что возможна альтернативная подстановка:

$$\{a \mapsto x, b \mapsto 6, x \mapsto 2\},$$

однако требование $x \neq 0$ удастся доказать только после решения исходного уравнения, поэтому такая гипотеза будет отвергнута системой.

6. Языки системы

В данном разделе описываются языки, используемые для описания правил и задач в экспериментальной системе. Приведённые ниже примеры синтаксиса служат для иллюстрации описанного подхода — язык находится в разработке и может существенно изменяться.

6.1. Язык описания правил

Архитектура системы не зависит от конкретного синтаксиса записи правил, что позволяет в будущем адаптировать или полностью заменить язык описания правил без изменения ядра системы. Однако для формирования более конкретного представления о процессе обучения системы и лучшего понимания описанных ранее механизмов ниже приводятся примеры описания правил, используемых в текущей реализации.

6.1.1. Структура правила

Правила преобразования термов описываются с использованием специального синтаксиса, который позволяет формально задать шаблоны преобразований и условия их применения. Новое правило создаётся с помощью ключевого слова `rule`, далее в фигурных скобках перечисляется содержимое правила.

```
rule {
    attr <атрибуты_правила>;

    <шаблон> => <результат>;

    <ограничение_1>,
    <ограничение_2>,
    ...;
}
```

Листинг 1: Общая структура правила

6.1.2. Основная часть правила

Первая строка в теле правила, не выделенная ключевым словом, считается действующей частью правила и должна иметь вид:

1. `pattern => resolution;` — одностороннее преобразование
2. `pattern <=> resolution;` — равносильное преобразование

Отличие между двумя приведёнными записями состоит в том, что переход (2) считается равносильным. Это позволяет применять правило к цели задачи на доказательство, чтобы получать альтернативную цель, эквивалентную исходной. По возможности следует использовать именно форму (2) там, где это возможно.

6.1.3. Ограничения правила

Последующие строки считаются ограничениями (на наборы параметров) правила; они перечислены через запятую и оканчиваются точкой с запятой. Ограничения проверяются перед применением правила.

6.1.4. Атрибуты правил

Ключевое слово `attr` позволяет задавать атрибуты правила:

- `level(<num>)` — уровень срабатывания;
- `goal(<target>)` — ограничение по целевой установке;
- `id(<id>)` — уникальный идентификатор правила;
- `replace` — в случае успешного применения исключает исходный терм из рассмотрения в процессе дальнейшего решения (см. раздел 4.3.1);
- `block(<rule_id>)` — блокировка применения указанного правила к результату;
- `zero(<param>)` — требование рассмотрения случая, когда параметр равен нулю;
- `one(<param>)` — требование рассмотрения случая, когда параметр равен единице;
- `normalize(<level>)` — уровень нормализации результата применения правила. Каждый уровень включает операции предыдущих уровней. 0 — только раскрытие скобок ассоциативных операций, 1 — устранение тривиальных операций (добавление и вычитание

нуля, умножение на единицу и т. д.), упорядочивание аргументов коммутативных операций, 2 — проведение арифметических вычислений, 3 — приведение подобных, сворачивание степеней (например, $x \cdot x \to x^2$).

6.1.5. Генератор правил

Атрибуты `zero(<param>)` и `one(<param>)` позволяют описать несколько логически связанных случаев преобразований в рамках единого правила. Это полезно в случаях, когда шаблонное выражение может сильно поменять свой вид при обращении параметров в 0 или 1 из-за отсутствия соответствующих слагаемых или множителей.

6.1.6. Примеры

Рассмотрим пример простого правила из школьной алгебры, переносящего правую часть уравнения влево с противоположным знаком (Листинг 2). Система применяет это правило при решении уравнений, чтобы привести одно из условий к виду, узнаваемому другими правилами. Отсюда целевая установка `find(x)`. Заметим, что `x` не используется в терме правила или его ограничениях; это означает, что `x` может быть каким угодно, лишь бы сама целевая установка относилась к задаче на поиск значений переменных.

Атрибут `level(0)` устанавливает для правила максимальный приоритет. Таким образом, правило будет применено к новым термам раньше, чем правила с уровнем выше. Однако возможны случаи, когда это правило потребуется заблокировать. Чтобы иметь возможность указать это правило в качестве блокируемого, ему присваивается текстовый идентификатор `id(move_left)`. Пример блокировки правила будет рассмотрен далее.

Единственное ограничение проверяет, что подтерм `b` не равен терму `0` синтаксически, что отличается от ограничения `b != 0`, которое подразумевает семантическое равенство (в таком случае системе бы пришлось доказать, что выражение b не обращается в ноль в контексте задачи). Это ограничение носит оптимизационный характер и говорит о том, что константу 0 не нужно переносить влево. Конечно, можно было и не указывать этот момент, потому что при нормализации лишний ноль в левой части будет удалён, и новый терм совпадёт со старым и не будет добавлен к списку термов, однако проверить такое ограничение быстрее.

```
rule {
    attr replace, level(0), goal(find(x)), id(move_left);

    a == b <=> a - b == 0;

    !symbolic_eq(b, 0);
}
```

Листинг 2: Правило переноса в левую часть

Рассмотрим ещё один пример правила, применяемого при решении линейных уравнений (Листинг 3). Стоит отметить применяемый генератор правил. При обращении параметра `a` в единицу, либо при обращении в ноль параметра `b` терм, семантически являющийся линейным уравнением, перестаёт синтаксически подходить под общий шаблон. Данная языковая конструкция требует от системы подставить указанные значения параметров для получения альтернативных вариантов шаблона и провести их нормализацию.

Конструкция `block(move_left)` блокирует применение правила переноса в левую часть, которое могло бы помешать усмотрению ответа, либо сделать замену переменной `x` на правую часть выражения в другом терме.

Требования в данном правиле лишь уточняют определение линейного уравнения. Конструкция `b is known` требует отсутствия вхождения неизвестных в подтерм `b`, в частности отсутствие вхождения `x`.

```
rule {
    attr level(1), goal(find(x)), one(a), zero(b), block(
    move_left);

    a*x + b == 0 => x == - b/a;

    a != 0,
    b is known;
}
```

Листинг 3: Правило решения линейного уравнения

Последний пример приведён с целью продемонстрировать ограничения работы нормализации (Листинг 4). В некоторых задачах может потребоваться явное раскрытие скобок. Данный пример осуществляет понижение степени методом отделения множителя, когда выражение под степенью является суммой нескольких известных аргументов, например, когда одно из слагаемых является иррациональным числом, записанным с помощью радикалов. Однако при нормализации может быть обнаружено произведение одинаковых выражений, и множитель вернётся под знак

степени до того, как другое правило раскроет скобки в операции умножения. Здесь используется понижение уровня нормализации до второго, где такое преобразование уже не происходит.

```
rule {
   attr replace, level(4), goal(transform(x)), normalize(2);

   (a+b)^n <=> (a+b)^(n - 1)*(a + b);

   a is known,
   b is known;
}
```

Листинг 4: Правило разложения степени суммы

6.2. Язык описания задач

Синтаксис задач позволяет задать условия и целевую установку. Новая задача вводится с помощью ключевого слова `task`, далее в фигурных скобках вводятся данные задачи. Ключевое слово `goal` обозначает целевую установку задачи.

```
task {
    goal <целевая_установка>;
    text "Текст задачи для пользователя";
    answer Терм ответа для проверки;

    <условие_1>;
    <условие_2>;
    ...
}
```

Листинг 5: Общая структура задачи

6.2.1. Целевые установки

Целевые установки полностью соответствуют описанным ранее:

1. `find(x)` — выражение значения переменной x через известные параметры.
2. `prove(<statement>)` — доказать истинность утверждения (вывести его из условий задачи).
3. `transform(<statement>)` — преобразовать (упростить) выражение.

Целевая установка `goal` задачи обязательна, все прочие элементы опциональны.

6.2.2. Дополнительные элементы

- Ключевое слово `text` позволяет ввести текст задачи на любом языке. Этот текст никак не используется системой при решении задачи, он просто выводится пользователю.
- Ключевое слово `answer` позволяет задать ожидаемый ответ на задачу. При решении система не использует этот ответ, однако проводит сверку в конце процедуры. Таким образом, накопленная база задач выступает в роли набора интеграционных тестов для системы.
- Все прочие утверждения считаются условиями задачи.

6.2.3. Примеры

Приведем несколько примеров вводимых задач.

```
task {
    goal find(x);

    2*(x+4) + 3*(x-5) == 0;
}
```

Листинг 6: Линейное уравнение

```
task {
    goal find(x);

    2*x-5+t == 0;
    t is known;
}
```

Листинг 7: Линейное уравнение с параметром

```
task {
    goal find(x);

    3*x^2+2*x-5 == 0;
}
```

Листинг 8: Квадратное уравнение

```
task {
    goal prove(x > 0);

    x == 2;
}
```

Листинг 9: Простейшая задача на доказательство требующая подстановки

Список литературы

[1] А. С. Подколзин, *Компьютерное моделирование логических процессов*. Т. 1: *Архитектура и языки решателя задач*, Физматлит, Москва, 2008.

[2] Дж. Малпас, *Реляционный язык пролог и его применение*, Наука, Москва, 1990.

[3] Э. Мендельсон, *Введение в математическую логику*, Наука, Москва, 1971.

[4] F. Baader, T. Nipkow, *Term Rewriting and All That*, Cambridge University Press, Cambridge, 1998. DOI: 10.1017/CBO9781139172752.

Приложение А. Исходный код программы

Исходный код программы доступен по ссылке:
`https://github.com/Quotique/laffie/tree/v0.5`

Приложение Б. Пример работы алгоритма

Данный пример демонстрирует работу алгоритма на примере простой задачи: целевая установка **Find**(x), условия: $\{4x + 7 = 2x + 1\}$.

Используемые правила

1. r_1: $a = b \Rightarrow a - b = 0; L_{r_1} = 0$; $\{$`!symbolic_eq`$(b, 0)\}$ — правило переноса в левую часть.
2. r_2: $a \cdot u + b = 0 \Rightarrow u = -b/a; L_{r_2} = 1$; $\{a \neq 0,$ b `is known`$\}$ — правило решения линейного уравнения.

Шаги решения

0. Инициализация:
   - goal : Find(x).
   - t_1: $4x + 7 = 2x + 1$, $L_{t_1} = 0$.

- t_2: x, $L_{t_2} = 0$, IsAlternativeTarget = true.

1. Выбран $t = t_1, L_t = 0$.
   - Производится упрощение терма через подзадачу Transform($4x + 7 = 2x + 1$), однако в системе отсутствуют правила, применимые для упрощения этого терма, поэтому терм помечается как упрощённый: t_1.Simplified $\leftarrow$ true.
   - Терм t_1 не имеет вид $x = \ldots$ или $x \in \ldots$, поэтому не является ответом на задачу.
   - Выбирается правило r_1 уровня $L_r = L_t$: $a = b \Rightarrow a - b = 0$.
   - Выбирается подстановка: $P = \{a \mapsto 4x + 7, b \mapsto 2x + 1\}$.
   - Формирование гипотезы: $(4x + 7) - (2x + 1) = 0$.
   - Нормализация: $2x + 6 = 0$.
   - Проверка ограничения: `!symbolic_eq`$(2x + 1, 0)$ — истина.
   - Добавление нового терма: $t_3 : 2x + 6 = 0$, $L_{t_3} = 0$.
2. Выбран $t = t_1, L_t = 0$.
   - Упрощение терма и проверка на ответ повторно не производятся.
   - Правило r_1 не допускает альтернативных подстановок, дубликат терма t_3 пропускается.
   - Других правил уровня 0 не найдено, уровень терма увеличивается: $L_{t_1} \leftarrow 1$.
3. Выбран $t = t_2, L_t = 0$.
   - Аналогично t_1, t_2 не может быть упрощён (t_2.Simplified $\leftarrow$ true) и не является ответом на задачу.
   - Правил для преобразования t_2 не найдено, уровень увеличивается: $L_{t_2} \leftarrow 1$.
4. Выбран $t = t_3, L_t = 0$.
   - Аналогично, терм не может быть упрощён и помечается как упрощённый: t_3.Simplified $\leftarrow$ true.
   - Терм не является ответом.
   - Выбирается правило r_1 уровня $L_r = L_t$.
   - Выбирается подстановка: $P = \{a \mapsto 2x + 6, b \mapsto 0\}$.
   - Формирование гипотезы: $2x + 6 - 0 = 0$.
   - Нормализация: $2x + 6 = 0$.

- Данный терм уже есть в списке термов, обоснование не производится.
- Откат к выбору подстановки. Других подстановок не найдено.
- Переход к следующему правилу.
- Других правил не найдено, уровень увеличивается: $L_{t_3} \leftarrow 1$.

5. Выбран $t = t_1, L_t = 1$.
   - Упрощение терма и проверка на ответ повторно не производятся.
   - Правил для преобразования t_1 не найдено, уровень увеличивается: $L_{t_1} \leftarrow 2$.
6. Выбран $t = t_2, L_t = 1$.
   - Упрощение терма и проверка на ответ повторно не производятся.
   - Правил для преобразования t_2 не найдено, уровень увеличивается: $L_{t_2} \leftarrow 2$.
7. Выбран $t = t_3, L_t = 1$.
   - Упрощение терма и проверка на ответ повторно не производятся.
   - Выбирается правило r_2 уровня $L_r = L_t$: $a \cdot u + b = 0 \Rightarrow u = -b/a$.
   - Выбирается подстановка: $P = \{a \mapsto x, b \mapsto 6, u \mapsto 2\}$.
   - Формулируется гипотеза: $2 = -6/x, \{x \neq 0,\ 6\ \texttt{is known}\}$.
   - Для обоснования гипотезы вызывается подзадача Prove($x \neq 0$). В рамках этой подзадачи не будет выведено новых термов, ввиду отсутствия подходящих правил, поэтому задача так и останется нерешённой.
   - Гипотеза отвергается, происходит откат к выбору подстановки.
   - Выбирается подстановка: $P = \{a \mapsto 2, b \mapsto 6, u \mapsto x\}$.
   - Формулируется гипотеза: $x = -6/2$, $\{2 \neq 0,\ 6\ \texttt{is known}\}$.
   - Нормализация: $x = -3$, $\{2 \neq 0,\ 6\ \texttt{is known}\}$.
   - Ограничения выполнены тривиально.
   - Добавление нового терма: t_4: $x = -3$, $L_{t_4} = 0$.
8. Выбран $t = t_4, L_t = 0$.
   - Терм помечается как упрощённый: t_4.Simplified $\leftarrow$ true.
   - Проверка на ответ: терм имеет форму $x = v$, где $v = -3$ — известное значение.
   - **Ответ найден**: $x = -3$.

Приложение В. Алгоритм поиска правил в базе правил

Алгоритм 1: Поиск правил для терма в базе правил

Вход : Терм t, L_t, база правил $\mathcal{R}$
Выход: $\mathcal{R}_2 \subseteq \mathcal{R}$

$\mathcal{R}_1 \leftarrow \emptyset$
for each $s \in \text{Symbols}(t)$ **do**
    $\mathcal{R}_1 \leftarrow \mathcal{R}_1 \cup \mathcal{R}_s[L_t]$
$\mathcal{R}_2 \leftarrow \emptyset$
for each $r \in \mathcal{R}_1$ **do**
    if $\text{Symbols}(t) \supseteq \text{Symbols}(r.\text{pattern})$ **then**
        $\mathcal{R}_2 \leftarrow \mathcal{R}_2 \cup \{r\}$
return $\mathcal{R}_2$

Приложение Г. Алгоритм применения правил к терму

Алгоритм 2: Применение правил к терму

Вход : t, $\mathcal{R}_3$, $\mathbb{T}$, Goal
Выход : $(\text{found}, t_{\text{new}})$

for each $r \in \mathcal{R}_3$ **do**
  if Goal.type = Prove **and** t.IsAlternativeTarget **and** r — *неэквивалентное преобразование* **then**
    continue
  for each $m \in \text{matches}(r.\text{pattern}, t) \cap \text{matches}(r.\text{goal}, \text{Goal})$ **do**
    for each $\sigma \in \text{substitutions}(m)$ **do**
      $\hat{t}_{\text{raw}} \leftarrow \sigma(r.\text{conclusion})$
      $\hat{t} \leftarrow \text{normalize}(\hat{t}_{\text{raw}})$
      if $\hat{t} \in \{s \in \mathbb{T} \mid s.\text{IsAlternativeTarget} = t.\text{IsAlternativeTarget}\}$ **then**
        continue
      ok $\leftarrow$ true
      for each $c \in r.\text{constraints}$ **do**
        $c_{\text{norm}} \leftarrow \text{normalize}(\sigma(c))$
        if $\neg\text{prove}(c_{norm})$ **then**
          ok $\leftarrow$ false
          break
      if ok **then**
        $\hat{t}$.IsAlternativeTarget $\leftarrow t$.IsAlternativeTarget
        $\hat{t}$.IsReplaced $\leftarrow$ false
        $\hat{t}$.Simplified $\leftarrow$ false
        $\mathbb{T} \leftarrow \mathbb{T} \cup \{\hat{t}\}$
        $L_{\hat{t}} \leftarrow 0$
        if r.replace **then**
          t.IsReplaced $\leftarrow$ **true**
        $\mathcal{R}_{\text{applied}}(\hat{t}) \leftarrow \emptyset$
        $\mathcal{R}_{\text{blocked}}(\hat{t}) \leftarrow r.\text{blocks}$
        Inference$(\hat{t}) \leftarrow (r, \sigma)$ + доказательства условий r.constraints
        return $(\text{true}, \hat{t})$
  $\mathcal{R}_{\text{applied}}(t) \leftarrow \mathcal{R}_{\text{applied}}(t) \cup \{r\}$
return $(\text{false}, 0)$

Приложение Д. Основной алгоритм работы системы

В данном приложении представлен псевдокод основного цикла работы алгоритма, детальное описание которого приведено в разделе 5.

Алгоритм 3: Основной алгоритм работы системы

Вход : $\mathbb{T}$, Goal, $\mathcal{R}$, $I_{\max}$ — лимит итераций
Выход : (found, ans)

$L_{\max} \leftarrow \max_{r \in \mathcal{R}} L_r$
found $\leftarrow$ false
for $i \leftarrow 1$ **to** I_{max} **do**
  if $\mathbb{T}_{\text{active}} = \varnothing$ **then**
    break
  $t \leftarrow \arg\min_{t' \in \mathbb{T}_{\text{active}}} L_{t'}$
  if $L_t > L_{\max}$ **then**
    break
  if $\neg t$.Simplified $\wedge$ Goal.type $\neq$ Transform **then**
    $t' \leftarrow \text{transform}(t)$
    if $t' \neq t$ **then**
      t'.IsAlternativeTarget $\leftarrow$ t.IsAlternativeTarget
      Inference(t') $\leftarrow$ контекст подзадачи **Transform**
      $L_{t'} \leftarrow 0$
      t.IsReplaced $\leftarrow$ **true**
      t'.Simplified $\leftarrow$ **true**
      $\mathbb{T} \leftarrow \mathbb{T} \cup \{t'\}$
      $t \leftarrow t'$
      continue
    else
      t.Simplified $\leftarrow$ **true**
  if is_answer(t, Goal) **then**
    found $\leftarrow$ **true**
    ans $\leftarrow t$
    break
  if check_substitution(t) **then**
    t.IsSubstitution $\leftarrow$ **true**
  $R \leftarrow \text{query\_rules}(\mathcal{R}, L_t, t) \setminus \mathcal{R}_{\text{applied}}(t) \setminus \mathcal{R}_{\text{blocked}}(t)$
  (ok, t) $\leftarrow$ apply_rules($t, R, \mathbb{T}$, Goal)
  if $\neg$ok **then**
    $L_t \leftarrow L_t + 1$
if Goal.type = Transform $\wedge \neg$found **then**
  found $\leftarrow$ **true**
  ans $\leftarrow$ last_from $\{t \in \mathbb{T} \mid t.\text{IsAlternativeTarget}\}$
return (found, ans)

Статья поступила 29 января 2026 г.

Architecture of a Simplified Automated Reasoning System Based on A. S. Podkolzin's Approach

A. P. Annenkov

Solving formalizable mathematical problems using computers remains a relevant challenge with several fundamentally different approaches. A significant contribution to this field was made by A. S. Podkolzin, who proposed an original solver architecture based on logical inference using term rewriting rules. This article describes the architecture of a simplified system that implements the key ideas of this approach: an attention-switching mechanism via levels, organization of a rule base, and context representation with metadata.

Keywords: mathematical problem solver, automated reasoning, logical processes, logical language, logical formalization of problems.

References

[1] A. S. Podkolzin, *Computer modeling of logical processes*. V. 1: *Architecture and languages of the problem solver*, Fizmatlit, Moscow, 2008.

[2] J. Malpas, *Prolog: A Relational Language and Its Applications*, Prentice-Hall, Englewood Cliffs, N. J., 1987.

[3] E. Mendelson, *Introduction to Mathematical Logic*, Van Nostrand, Princeton, N. J., 1964.

[4] F. Baader, T. Nipkow, *Term Rewriting and All That*, Cambridge University Press, Cambridge, 1998. DOI: 10.1017/CBO9781139172752.

Received on January 29, 2026

О методах идентификации и оценки причинно-следственных связей в неэкспериментальных данных

А. М. Ченцов* , Н. И. Торопов‡

Начало 21-го века для наук о данных характеризуется появлением новых междисциплинарных областей, а также расширением общей теоретической базы и появлением статистических методов для решения новых задач. В данной статье описываются известные подходы к решению задачи статистической идентификации односторонних (причинно-следственных) связей между переменными для оценки с использованием неэкспериментальных данных и приводятся отличительные особенности статистических моделей, подходящих для этих целей.

Основной результат работы заключается в сравнении методов оценивания эффектов воздействия в случае, когда зависимости в данных являются существенно нелинейными. Для этого предложен алгоритм генерации данных при помощи свёрточных нейронных сетей, и исследуются два разных подхода к оценке — методом байесовских сетей и методом двойного машинного обучения. Показано, что оба таких подхода в рассмотренном случае дают неточные оценки индивидуальных эффектов, и приводятся рекомендации по оценке агрегированных эффектов воздействия.

Ключевые слова: статистическая идентификация, эффекты воздействия, DAG-модели, двойное машинное обучение, CATE.

Введение

Выявление и оценка причинно-следственных зависимостей является важнейшей составляющей большинства задач в области здравоохранения, общественных и поведенческих наук. Эта тема долгое время являлась периферийной для математической статистики [1] и развивалась, преимущественно, экономистами: так, например, в 2021 году Г. Имбенс и Дж. Ан-

* *Ченцов Александр Михайлович* — ассистент каф. дискретной математики ФПМИ МФТИ, e-mail: achentsov@nes.ru, ORCID: 0000-0003-0429-2664.

Chentsov Aleksandr Mikhailovich — assistant, Faculty of Applied Mathematics and Informatics of MIPT.

‡ *Торопов Никита Игоревич* — ассистент каф. дискретной математики ФПМИ МФТИ, e-mail: ntoropov@nes.ru, ORCID: 0000-0002-4274-2665.

Toropov Nikita Igorevich — assistant, Faculty of Applied Mathematics and Informatics of MIPT.

грист получили Нобелевскую премию по экономике за развитие методов причинно-следственного анализа. Эта ситуация начала меняться в начале XXI-го века с появлением работ, исследующих теоретические основы причинно-следственной идентификации, в частности, с использованием графических моделей. В настоящее время происходит постепенный синтез подходов к оценке эффектов воздействия, разработанных в различных областях науки – эконометрики, машинного обучения и математической статистики [2]. Количество исследований, связанных с разработкой методологии и статистической оценкой эффектов воздействия с использованием неэкспериментальных данных существенно выросло за последние 10 лет, причем некоторые исследователи подчеркивают важность этой задачи для построения моделей сильного искусственного интеллекта [1].

Понятию причинно-следственной связи, несмотря на его распространённость и естественность, достаточно сложно дать формальное определение. Райхенбах [3] связывает это понятие с естественным направлением времени. Другие авторы определяют причинно-следственность как идею о том, что могло бы произойти или какие (контрфактуальные) значения могли бы быть у числовых характеристик в гипотетическом сценарии, отличном от наблюдаемого на практике. В работе Льюиса [4] такие сценарии называются возможными мирами (англ.: possible worlds), и используются для следующей формализации:

Утверждение вида "если бы A, то выполнялось бы B" верно в мире w только тогда, когда B верно в ближайшем к w мире, в котором A верно.

Вид метрики не специфицируется автором, хотя отмечается, что ближайший для некоторого заданного утверждения A мир к w можно получить, применяя соображения общего здравого смысла. Галлес и Перл [5] используют построение Льюиса для описания некоторых формальных ограничений на свойства контрфактуальных утверждений.

Большинство ранних работ по статистической оценке эффектов воздействия связаны с рандомизированными контролируемыми испытаниями, в которых объекты испытаний случайно разделяются на две или более группы, отличающиеся наличием или уровнем воздействия – что позволяет минимизировать влияние посторонних факторов и более точно оценить влияние воздействия на интересующие исследователя характеристики. В то же время, для многих практически важных задач проведение рандомизированных экспериментов недоступно, а любой набор неэкспериментальных данных (англ.: observational data) является неполным – поскольку для каждого исследуемого объекта наблюдаемым является лишь одно состояние, тогда как другие возможные лишь постулируются. Невозможность сравнения контрфактуальных характеристик напрямую

приводит к тому, что исследователям требуется дополнять данные посредством теоретических предположений, которые позволяли бы извлечь информацию из имеющихся наблюдений; в работе Холланда [6] это заключение было названо фундаментальной проблемой выводов (инференции) о причинно-следственных связях.

Далее в представленной работе приводится краткий обзор теоретических методов идентификации эффектов воздействия: модель потенциальных исходов Рубина и модели на ориентированных ациклических графах. Показан механизм, как от общих соображений о связях между переменными в исследуемой задаче во многих случаях можно прийти к статистической модели, позволяющей оценить эффекты воздействия без проведения экспериментов. В третьей главе приводится основной результат работы, заключающийся в сравнении разных методов оценивания эффектов воздействия с использованием симулированных данных с нелинейными зависимостями, и приводятся рекомендации по выбору оптимальной процедуры оценивания. Для этого предложен новый алгоритм генерации данных с помощью свёрточных нейронных сетей, а также алгоритм дискретизации таких данных с последующей оценкой условного распределения.

1. Модель потенциальных исходов Рубина

Исторически, первой из статистических моделей, применимых для идентификации и оценки эффектов воздействия при анализе неэкспериментальных данных, является модель потенциальных исходов Рубина – в честь автора, предложившего эту модель в публикации [7], – однако, некоторые элементы использованного подхода применительно к случаю рандомизированных контролируемых экспериментов были предложены ещё в 1923 году Нейманом [8].

В модели для каждого объекта рассматриваемой популяции постулируется существование спектра различных состояний, описываемых переменной воздействия. Это позволяет говорить о потенциальных исходах зависимой переменной и в этих терминах определять причинно-следственные зависимости. Когда состояний всего два, то их традиционно называют контрольным состоянием и воздействием. Если состояний больше двух, то одно из них назначается контрольным, а остальные расцениваются как различные уровни воздействия, либо как альтернативные варианты воздействий – если сравнивать их интенсивность некорректно.

На практике не всегда существует возможность специфицировать состояния объектов достаточно точно – в силу ограничений, непосред-

ственно связанных с аккуратной формулировкой самих состояний, либо из-за сложности последующего сбора данных о них. В то же время, от этого этапа построения модели зависит интерпретация оценок, а также то, насколько получаемые результаты могут быть распространены с выборки на всю моделируемую популяцию.

Под потенциальными исходами в модели Рубина понимаются числовые характеристики, возможно случайные – причём по одной величине приписывается каждой паре из объекта наблюдения и его состояния. В простейшем случае, когда состояний всего два, их можно обозначить за Y_{i1} и Y_{i0}. Иногда представляется целесообразным, особенно для рассмотрения общих понятий, сопоставить потенциальные исходы со случайным процессом $Y = Y(\omega, d)$, где $d \in \{0, 1\}$ соответствует отсутствию/наличию воздействия, а $\omega \in \Omega$ – состояние мира, отражающее случайность в выборе конкретного объекта, и случайность, связанную с любыми другими возможными причинами. Если потенциальных исходов всего два, то можно ввести индикатор воздействия $D_i : \Omega \to \{0, 1\}$, который определяет, в каком из состояний находится интересующий нас объект наблюдения. Тогда индивидуальный эффект воздействия можно определить, как разность между потенциальными исходами

$$\delta_i = Y_{i1} - Y_{i0}, \tag{1}$$

и, поскольку δ_i в такой постановке может быть случайной величиной, то практически всегда рассматривается лишь её усреднённое значение

$$\delta_i^E = \mathbb{E}Y_{i1} - \mathbb{E}Y_{i0}. \tag{2}$$

Заметим, что нет оснований определять эффект воздействия исключительно как разность двух величин. Например, иногда исследователя может интересовать процентное различие между двумя величинами, и тогда можно рассматривать эффект воздействия как частное $\left(\frac{Y_{i1}}{Y_{i0}} - 1\right) \times 100\%$; также возможны и другие функциональные формы.

Если ввести переменную Y_i, обозначающую фактически наблюдаемый потенциальный исход для i-го объекта, то для контрольной группы данная переменная будет принимать одно из возможных значений Y_{i0}, а для экспериментальной – одно из возможных значений Y_{i1}. Формально это можно записать в виде уравнения (3):

$$Y_i = D_i Y_{i1} + (1 - D_i) Y_{i0} \tag{3}$$

Одним из возможных решений фундаментальной проблемы инференции является агрегация эффектов воздействия для элементов популяции

в один средний эффект, при этом можно попытаться использовать информацию об объектах из экспериментальной группы для выводов о $\mathbb{E}Y_{i1}$ и, аналогично, объекты из контрольной группы можно использовать для оценки популяционного значения $\mathbb{E}Y_{i0}$.

Рубин в своей работе ввёл идентификационное предположение об устойчивости индивидуальной величины воздействия – набор требований, достаточных для того, чтобы считать среднее по подвыборкам в группе воздействия и в контрольной группе несмещёнными оценками соответствующих популяционных значений.

Устойчивость индивидуальной величины воздействия:

1. Эффект воздействия на один элемент популяции не зависит от значений воздействия у других элементов популяции;
2. Эффект воздействия на один элемент популяции не зависит от механизма получения этого воздействия, то есть, наблюдаемое значение случайной величины Y_i соответствует потенциальному исходу $Y(D_i)$.

Первую часть этого предположения можно интерпретировать как отсутствие экстерналий (внешних эффектов). Вторая компонента предположения близка по смыслу к свойству контрфактуальных утверждений, которое в [9] называется состоятельностью (англ.: consistency), а в работе Галлеса и Перла [5] – композицией; кроме того, в эконометрической литературе распространено близкое по смыслу понятие экзогенности. Оптимизационное поведение исследуемых объектов нередко ведёт к нарушению данного предположения в ситуациях, когда они самостоятельно отбираются для того, чтобы подвергнуться воздействию. Такие же проблемы можно наблюдать в случаях, когда воздействие осуществляется на основе тех или иных характеристик объектов изучения.

Предложенный подход можно проиллюстрировать, используя результаты исследования влияния госпитализации на здоровье людей, приведённые в [10]. В этом примере использовались данные опроса населения США The National Health Interview Survey, в котором включены вопросы, “Был ли госпитализирован опрашиваемый в течение последних 12 месяцев”, и “Как вы оцениваете своё здоровье по шкале от 1 (отличное) до 5 (плохое)”.

Из данных в таблице следует, что опрошенные после госпитализации имеют в среднем существенно худшие показатели здоровья. Однако, с другой стороны, наверняка такие люди изначально обладали существенно худшим здоровьем, чем те, кто не был госпитализирован.

Обозначая за Y_{i0} уровень здоровья человека, не подвергнутого госпитализации, а Y_{i1} – уровень здоровья того же человека после госпитализации,

Таблица 1: Описательная статистика показателя самооценки уровня здоровья, источник [10]

Группа	Размер выборки	Среднее значение	Стандартная ошибка
Воздействие	7774	2.79	0.014
Контрольная	90049	2.07	0.003

можно получить следующее представление для среднего эффекта воздействия:

$$\underbrace{\mathbb{E}\left(Y_i \mid D_i = 1\right) - \mathbb{E}\left(Y_i \mid D_i = 0\right)}_{\text{наблюдаемая разность}} = \underbrace{\mathbb{E}\left(Y_{i1} \mid D_i = 1\right) - \mathbb{E}\left(Y_{i0} \mid D_i = 1\right)}_{\text{ATT}} +$$

$$+ \underbrace{\mathbb{E}\left(Y_{i0} \mid D_i = 1\right) - \mathbb{E}\left(Y_{i0} \mid D_i = 0\right)}_{\text{смещение выборки}}$$

Нас интересует первый член в правой части равенства – ATT (от англ. average treatment effect on the treated – средний эффект воздействия в части популяции, подвергшейся воздействию) – поскольку он является наиболее подходящим доступным приближением среднего эффекта воздействия по всей популяции ATE (от англ. average treatment effect). При этом, наблюдаемая разность также содержит компоненту, интерпретируемую как смещение выборки, – то есть среднее различие здоровья между теми, кто был, и кто не был госпитализирован. В случаях, когда воздействие назначается случайно (включая рандомизированные контролируемые испытания), выполняется $\mathbb{E}\left(Y_{i0} \mid D_i = 1\right) = \mathbb{E}\left(Y_{i0} \mid D_i = 0\right)$, поэтому смещение выборки равно нулю, и разность средних значений имеет причинно-следственную интерпретацию.

Фундаментальные отличия между экспериментальной и контрольной группами в рассмотренном примере означают, что условия устойчивости индивидуальной величины воздействия нарушены, и без дополнительной информации средний эффект воздействия не идентифицирован. Заметим также, что если бы данные из таблицы 1 использовались для оценки регрессии Y_i на D_i и константу (с использованием метода наименьших квадратов), то оценка коэффициента при D_i также была бы равна разности средних показателей здоровья в двух группах. Таким образом в этом примере для того чтобы регрессионная оценка соответствовала причинно-следственному эффекту от госпитализации, требуется решить проблему смещённости выборки.

Подробный разбор модели Рубина для системы с конечным числом состояний можно найти в монографии [11].

2. Причинно-следственные диаграммы и структурные статистические модели

Другой подход к идентификации структурных параметров использует модели на орграфах (диаграммы причинно-следственных связей), в которых каждая из вершин соответствует одной из переменных, представляющих интерес в конкретной задаче; при этом никакая переменная не ставится в соответствие с более чем одной вершиной, а рёбра графа могут быть отождествлены с теми или иными зависимостями между переменными. Такие модели являются частью более общего класса графических моделей, в которых направленные связи используются для задания марковских свойств совместного распределения. В нашем случае, связанная пара вершин показывает наличие статистической связи между соответствующими переменными, а направленный путь отождествляется с причинно-следственной связью от вершин-родителей к детям. Ациклич-ность орграфов необходима, поскольку в обратном случае в переменных нельзя разделить причину и следствие.

Применение графических моделей для описания структуры зависимостей в данных впервые встречается в работах Сьюэла и Филипса Райта, [12], и было популяризовано в 2000-х годах работами Перла [13], некоторые идеи из которых представлены далее. Модели, основанные на таких графах, называются DAG-моделями (от англ. directed acyclic graph).

Для заданного ориентированного ациклического графа можно говорить о совместных распределениях, которые являются совместимыми с ним. Такие распределения можно определить, как все распределения, для которых существует такая нумерация переменных, что если вершина j является родителем вершины i в графе, то переменная j входит в число родителей переменной i в марковском смысле.

Содержательно понятие совместимости ориентированного ациклического графа с заданным совместным распределением играет ключевую роль, поскольку оно оказывается необходимым и достаточным условием для того, чтобы распределение могло быть порождено процессом, который последовательно генерирует переменные из соответствующих условных распределений, принимая во внимание значения переменных-родителей, сгенерированных на предыдущих шагах.

Одной из наиболее важных характеристик совместного распределения, которая представляет особый интерес в прикладных статистических исследованиях, является набор отношений условной независимости. Оказывается, что такую информацию можно получить, используя направленный ациклический граф, с которым заданное распределение совместимо. Для

этого требуется ввести понятие блокировки пути: ненаправленный путь S на графе DAG-модели блокирован набором вершин Z, если выполнено хотя бы одно из следующих условий:

- в S найдётся цепочка $X_1 \to X_2 \to X_3$ или вилка $X_1 \leftarrow X_2 \to X_3$, причем $X_2 \in Z$;
- в S найдётся обратная вилка (коллайдер) $X_1 \to X_2 \leftarrow X_3$, такая что $X_2 \notin Z$, и также никакой из потомков X_2 не содержится в Z.

Такие обозначения связаны со статистической зависимостью между переменными в модели. Действительно, если переменные X_1 и X_3 имеют общую причину X_2, то они статистически зависимы – в полном соответствии с принципом Райхенбаха [3]. В обратной ситуации, две безусловно независимые переменные X_1 и X_3, имеющие общий эффект X_2, становятся статистически зависимыми условно на значение X_2.

Будем также говорить, что наборы переменных X и Y разделены Z, если все пути от каждого из элементов X к каждому из элементов Y блокированы. Можно показать, что две группы случайных величин X и Y независимы условно на третью группу случайных величин Z при любом распределении, согласованном с заданным ориентированным ациклическим графом, если Z разделяет X и Y в этом графе в приведенном выше смысле. В обратную сторону, если Z не разделяет X и Y в этом графе, то существует согласованное с ним распределение, в котором X и Y не являются независимыми условно на Z.

Одна из причин удобства DAG-моделей заключается в предположении о том, что каждому ориентированному ребру в графе соответствует некий устойчивый и автономный механизм. Иными словами, можно представить ситуацию, где лишь одна из связей в графе меняется, оставляя все прочие неизменными. Организация знания в подобную модульную структуру позволяет использовать причинно-следственную диаграмму для предсказания эффекта внешних вмешательств с минимальной дополнительной информацией, поэтому подобная модель (предполагая, что она корректна) зачастую оказывается намного более информативной, чем набор уравнений.

Условия соответствия ациклического орграфа причинно-следственной модели.

Ориентированный ациклический граф G задаёт причинно-следственную модель, если выполнены следующие условия:

1. Все переменные, включенные в G, являются наблюдаемыми;
2. Все переменные, включенные в G, функционально зависят от своих родителей и идиосинкратических шоков – случайных величин,

которые независимы от других переменных в модели и друг от друга;

3. Каждая переменная X в графе гипотетически может быть подвергнута интервенции $do(X = x)$, которая
   - заменяет вероятностное распределение у X на фиксированное значение x,
   - удаляет из графа направленные рёбра, заканчивающиеся в X,
   - не производит никаких других эффектов по отношению к другим переменным в G (кроме тех эффектов, которые связаны с влиянием X на своих потомков в оставшемся графе).

DAG-модели и критерий обходных путей являются удобными инструментами для построения статистических моделей, называемых структурными, которые пригодны для оценки эффектов воздействия. Такие модели состоят из одного или нескольких уравнений, описывающих направленные связи между переменными. Они отличаются лишь набором предположений о свойствах ошибок (или, что эквивалентно, коэффициентов), которые для линейного случая можно записать в следующем виде:

$$Y := X^T\beta + \varepsilon. \tag{4}$$

Здесь Y – зависимая переменная, $X = (X_1, \ldots, X_\ell)^T$ – переменные воздействия/регрессоры, $\beta = (\beta_1, \ldots, \beta_\ell)^T$ – регрессионные коэффициенты. Знак присвоения в (4) подчеркивает направленную зависимость между переменными, и идентифицирует β, как такой параметр, при котором установка для X произвольного значения $x \in \mathbb{R}^\ell$, приводит к тому, что среднее значение Y становится равным $x^T\beta$.

Ненаблюдаемая переменная ε, называемая ошибкой модели, объединяет эффект всех переменных, являющихся причинами Y, но не включенных в модель. Параметр β из (4) также можно идентифицировать, задавая свойства ε с использованием *do*-нотации для интервенций:

$$\mathbb{E}\big(\varepsilon \mid do(X = x)\big) = 0,$$

что соответствует нулевому условному математическому ожиданию ошибки в модели, подвергнутой интервенции $do(X = x)$ и идентифицирует β через набор моментных соотношений, подробнее рассмотренных далее.

Примечание: авторы [10] рекомендуют при оценивании эффекта воздействия с использованием неэкспериментальных данных всегда представлять гипотетический рандомизированный эксперимент, соответствующий

рассматриваемой задаче. Такой подход достаточно полезен и для интерпретации результатов, и для проверки адекватности исследуемой модели поставленной задаче – хотя соответствующий рандомизированный эксперимент не всегда оказывается очевидным. Так, в работе [20] исследуется, насколько футбольные тренеры более склонны ставить автора победного гола в стартовый состав следующей игры, по сравнению с авторами голов, оказавшимися не последними. Поскольку в такой постановке важно отделить влияние именно того факта, что гол оказался победным – то есть, был забит при равном счете, и после него не было других голов, – то соответствующим рандомизированным испытанием можно считать эксперимент, в котором в матчах, выигранных командой с преимуществом в один мяч, и в которых было забито более одного гола, авторство забитых мячей меняется случайным образом среди игроков команды.

Определение. (Ациклическая) структурная модель, соответствующая графу $G = (V, E)$, – это набор случайных величин $\{X_j\}_{j \in V}$, задаваемых уравнениями

$$X_j := f_j(Pa_j, \varepsilon_j), \quad j \in V,$$

где Pa_j – родители по отношению к вершине j направленного графа, f_j – неизвестные (измеримые) функции, а ошибки ε_j являются в простейшем случае независимыми в совокупности.

Определение. Контрфактуальная структурная модель, получаемая вследствие интервенции $do(X_j = x_j)$ – это новая структурная модель, основанная на модифицированном графе $G(x_j) = (V, E^*)$ и наборе (контрфактуальных) переменных $(X_k^*)_{k \in V}$, где

- все рёбра, входящие в вершину j удалены, то есть $\forall i \in V$ если $e_{ij} \in E$, то $e_{ij}^* = 0$;
- все остальные рёбра сохранены, $\forall i, k \in V : \ k \neq j, \ e_{ik}^* = e_{ik}$;
- контрфактуальные случайные величины определяются следующим образом:

$$X_k^* := f_k(Pa_k^*, \varepsilon_k), \quad \text{при } k \neq j,$$
$$X_j^* := x_j,$$

  где Pa_k^* – родители X_k^* в E^*.

Для иллюстрации представленных понятий рассмотрим пример, основанный на так называемой “треугольной структурной модели” из [2].

Пусть структурная модель задана с помощью уравнений (5)-(7):

$$X := \varepsilon_x, \tag{5}$$
$$D := \alpha X + \varepsilon_d, \tag{6}$$
$$Y := \beta D + \gamma X + \varepsilon_y. \tag{7}$$

Такая структурная модель соответствует графу на рис. 1 слева:

Рис. 1: Слева: DAG-модель зависимости между рассматриваемыми переменными. Справа: результат интервенции $do(D = d)$

Для удобства предположим, что все переменные в модели совместно нормальны и имеют нулевое математическое ожидание, ошибки $\varepsilon_x, \varepsilon_y, \varepsilon_d$ независимы, а числовые коэффициенты α, β, γ таковы, что дисперсия у X, Y и D равна единице. Тогда легко видеть, что $\mathbb{E}(Y \mid D = d) = (\alpha\gamma+\beta)d$, но если установить для D значение d, то уравнение (6) теряет смысл, и заменяя в (7) значение D на константу, получаем, что функция регрессии примет вид $\mathbb{E}\big(Y \mid do(D = d)\big) = \beta d$, а средний эффект воздействия D на Y равен β. Полученной контрфактуальной модели соответствует граф на рис. 1 справа.

Сопоставляя свойства структурной линейной модели с моделью потенциальных исходов Рубина, легко видеть, что структурные модели являются моделями не для наблюдаемых, а для потенциальных исходов $Y(d)$, соответствующих различным возможным значениям переменной воздействия D. Тогда идентифицируемость β можно связать с условиями устойчивости индивидуальной величины воздействия. В рассмотренной модели (7) они выполнены благодаря предположению о независимости ошибок, и при оценке регрессии Y на D и X, оценка β методом наименьших квадратов (МНК) будет состоятельной (это можно проверить, применяя теорему Фриша-Ву-Ловелла). В то же время, в простой регрессии Y на D МНК-оценка будет иметь предел по вероятности, равный $\alpha\gamma+\beta$, что связано с нарушением предположения об экзогенности D: ошибка в такой модели содержит X и из-за этого имеет ненулевую корреляцию с D, что приводит к систематической зависимости между значением воздействия D и потенциальными исходами $Y(D)$.

В рассмотренном примере имела место типичная для прикладных исследований ситуация, когда для получения состоятельной оценки искомого эффекта D на Y требуется включить в модель дополнительную (контрольную) переменную X, не представляющую самостоятельного интереса. В более типичном случае совместная нормальность не предполагается и функции условного математического ожидания могут быть произвольными. Пусть, как обычно, D – переменная воздействия, Y – зависимая переменная, а $Y(d)$ – её потенциальные значения при $D = d$. Модель потенциальных исходов имеет вид

$$Y(d) := \beta_0 d + \varepsilon, \quad \mathbb{E}\varepsilon = 0,$$

где β_0 – значение параметра, интересующее исследователя, которое соответствует среднему изменению Y при увеличении d на единицу. Линейность этого эффекта может быть обоснована бинарностью переменной воздействия, или же в ситуации, когда нам важны лишь небольшие отклонения от некоторого значения d, что позволяет использовать линейное приближение. Предполагаем также, что переменная воздействия не является (безусловно) экзогенной, иначе β_0 можно было бы состоятельно оценить в простой регрессии Y на D, но у исследователя есть данные, частично объясняющие Y, то есть $\varepsilon = g(X) + u$, где X – вектор наблюдаемых случайных величин, $g : \mathbb{R}^k \to \mathbb{R}$ – неизвестная (измеримая) функция, вид которой не представляет самостоятельного интереса, а u – ошибка модели, обладающая свойством $\mathbb{E}(u \mid X) = 0$.

Предположение об условной экзогенности переменной воздействия в структурной причинно-следственной модели:

1. корректная спецификация модели (англ.: model consistency) – наблюдаемые данные Y_i сгенерированы в соответствии со структурной моделью $Y(d)$,
$$Y_i = Y(D_i);$$

2. условная некоррелированность:
$$\operatorname{cov}\big(D, Y(d) \mid X\big) = 0,$$
(наблюдаемая переменная воздействия D не коррелирует с ошибками модели условно на наблюдаемый набор контролей). В литературе на английском языке используются близкие по смыслу понятия conditional exogeneity и conditional ignorability, подразумевающие условную независимость D и $Y(d)$ при фиксированных X.

При таких предположениях в модели

$$Y = Y(D) := \beta_0 D + g(X) + u,$$

ошибка u ортогональна (в L^2-смысле) обоим регрессорам X и D, а значит параметр причинно-следственного эффекта β_0 является идентифицированным.

Оценка причинно-следственной связи на неэкспериментальных данных включает этап экспертного решения о выполнении условия экзогенности переменной воздействия в модели. Если такое условие может быть выполнено, то в оцениваемую модель следует включить контрольные переменные, обеспечивающие экзогенность переменной, то есть переменные, статистически зависимые с D и Y одновременно, причем не относящиеся к механизму воздействия D на Y. Пример использования такого подхода можно наблюдать в работе [14], в которой авторы исследовали гипотезу о влиянии открытости экономики на поведение валютных курсов.

В то же время, представленные условия дают возможность проверки полученных результатов следующим образом: в ходе дальнейшего исследования структуры зависимостей между переменными в рассматриваемой задаче могут быть обнаружены новые потенциально релевантные контрольные переменные. При этом, в случае, когда изначально предложенная структура зависимостей является верной, включение таких переменных в модель не должно приводить к существенному изменению оценок причинно-следственного эффекта. Подобная проверка может лишь увеличить убедительность полученного результата, но не окончательно доказать его (поскольку перебрать все потенциально возможные наборы контрольных переменных на практике невозможно, кроме гипотетической ситуации, когда удаётся подобрать модель, полностью объясняющую зависимую переменную), однако даёт достаточно эффективный механизм выявления ошибочных результатов; например, подобный подход использовался в работе [15].

3. Сравнение различных методов оценки эффектов воздействия с использованием симулированных данных

Для сравнения различных подходов к оценке индивидуальных эффектов воздействия использовались симулированные данные с существенно нелинейными зависимостями. Причинно-следственная диаграмма процесса порождения данных представлена на рис. 2 и при оценке зависимости

Y_i от D_i предполагалась известной. Выбор данной структуры объясняется тем, что это наиболее простой случай, требующий включения всех переменных в статистическую модель, что позволяет сфокусироваться на проблемах оценивания нелинейных зависимостей. Во многих практических задачах структура зависимостей будет сводиться к аналогичной после отбора переменных на основе критерия обходных путей. Кроме того, наличие возможных зависимостей между переменными $X_1, \ldots, X_k$ качественно не повлияет ни на одну из далее представленных процедур.

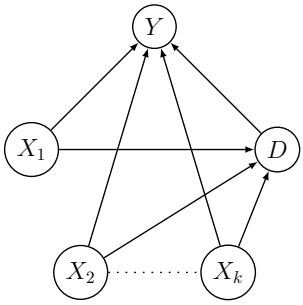

Рис. 2: Структура зависимостей между рассматриваемыми переменными

Генерация данных и оценка эффектов воздействия представлены в Алгоритмах 1-3.

Алгоритм 1. Генерация данных, соответствующих модели 2 с нелинейными зависимостями.

- Генерируется случайная выборка размером n нормальных векторов X_i размерностью $k \times 1$ с ковариационной матрицей $\sigma_x^2 I_k$. Полученная выборка подаётся на вход свёрточной нейронной сети с заданными параметрами глубины и числа слоёв, использующей активационные функции ReLU и наборы случайных центрированных весов w (использовались распределения с бесконечным вторым моментом, чтобы избежать сходимости в силу центральной предельной теоремы). На выходном слое к полученным переменным Z применялось преобразование $p = \frac{1}{1+e^{-Z^T w}}$
- Полученный вектор p используется для генерации переменных воздействия $D_i \sim \text{Bernoulli}(p_i)$.
- Переменные Y_i получались в результате применения к массиву Z, D еще одной аналогичной свёрточной нейросети, параметры весов которой сохранялись для вычисления контрфактуальных значений Y_{i0} и Y_{i1}.

Алгоритм 2. Вычисление эффектов воздействия с помощью дискретизации.

- Выбирается число узлов ℓ в сетке разбиения переменных, после чего значения X, Y округляются до ближайших выборочных квантилей уровня $\frac{1+2m}{2\ell}$, $m \in \{0, \ldots, \ell - 1\}$.
- Оценивается совместное распределение дискретизованных переменных (включая D), с использованием марковских свойств, соответствующих структуре зависимости в DAG-модели, рис. 3.
- Полученное совместное распределение используется для построения оценок функций регрессии $\mathbb{E}(Y_i \mid D_i = 1, X_i)$ и $\mathbb{E}(Y_i \mid D_i = 0, X_i)$, которые вычисляются в точках (дискретизованной) выборки
- $\hat{\mathbb{E}}(Y_i \mid D_i = 1, X_i) - \hat{\mathbb{E}}(Y_i \mid D_i = 0, X_i)$ используется в качестве оценок индивидуальных эффектов воздействия, а их усреднение по всем наблюдениям используется в качестве оценки среднего эффекта воздействия (ATE).

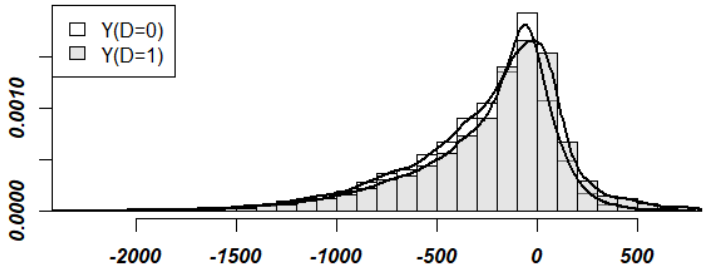

Рис. 3: Распределение Y при $D = 0$ и $D = 1$

Алгоритм 3. Вычисление эффектов воздействия с помощью метода двойного машинного обучения.

- Выборка используется для оценки структурной интерактивной регрессионной модели методом двойного машинного обучения [16]. Основное уравнение модели имеет вид:

$$Y_i := g(D_i, X_i) + \varepsilon_i, \quad \mathbb{E}(\varepsilon_i \mid X_i, D_i) = 0 \quad i \in \{1, \ldots, n\}, \tag{8}$$

где $g(\cdot)$ – неизвестная функция из $\mathbb{R}^k \times \mathbb{R}$ в $\mathbb{R}$, а ε_i – ошибки модели. Уравнение зависимости переменной воздействия D от контрольных переменных X:

$$D_i := m(X_i) + v_i, \quad \mathbb{E}(v_i \mid X_i) = 0. \tag{9}$$

Выбор метода оценки неизвестных функций осуществлялся с помощью алгоритма внутренней кросс-валидации, см. [14].

- Построение оценок индивидуальных эффектов воздействия осуществляется непараметрической оценкой $\mathbb{E}(Y_i \mid D_i = 1, X_i) - \mathbb{E}(Y_i \mid D_i = 0, X_i)$, после чего найденные значения ортогонально проецируются на базис из B-сплайнов для переменных X.
- Полученная функциональная форма позволяет вычислять оценки индивидуальных эффектов для любых наборов контрольных переменных, аналогично оценке, полученной в дискретном случае.

В таблице 2 приводятся результаты оценки среднего эффекта воздействия (ATE) с помощью линейных моделей. Из результатов видна нелинейная зависимость переменных от X: добавление контрольных переменных в модель регрессии Y на D лишь ухудшает оценку ATE (точное значение равно 70.0), но в присутствии полинома 5-й степени от X оценка становится достаточно точной, и близко совпадает с наилучшими оценками, полученными методом двойного машинного обучения.

Таблица 2: Оценка эффекта воздействия D на Y методом наименьших квадратов с различными наборами контрольных переменных

	(1)	(2)	(3)
	без контролей	линейные	полином 5й степ.
d	11.422	−20.555	53.224***
	(19.645)	(16.335)	(8.562)
x1		61.436***	...
		(3.115)	
x2		−20.779***	...
		(3.133)	
x3		200.987***	...
		(3.072)	
Число набл.	10,000	10,000	10,000
R^2	0.00003	0.320	0.822

Замечание: в скобках приведены стандартные отклонения оценок. Звёздочками помечены оценки, значимые на уровне 1%. В первом столбце оценивалась простая регрессия Y на D; во втором столбце оценивалась множественная регрессия Y на D и X_1, X_2, X_3; в третьем столбце в каче-

стве контрольных переменных использовались все компоненты полинома пятой степени от X_1, X_2, X_3.

В проведённых симуляционных экспериментах истинные значения индивидуальных эффектов сравнивались с предсказанными, и в большинстве случаев корреляция принимала значения от 0.1 до 0.2. Оценки усреднённых эффектов оказались неудовлетворительными при использовании дискретизации с числом точек p от 10 до 50. При этом достаточно хорошие оценки усреднённых эффектов можно было получить с использованием линейных моделей с полиномиальными контролями, а также при помощи двойного машинного обучения с использованием бустинга в качестве прогнозной модели. Из этого можно сделать вывод, что оценки индивидуальных эффектов в подобных процессах порождения данных могут быть существенно неточными. В частности, механизм настройки параметров моделей, предложенный в [19], заключающийся в использовании небольших выборок с заведомо экзогенными значениями переменной воздействия (получающиеся, например, путём проведения рандомизированного испытания в небольшом масштабе), в нашем случае, вероятнее всего, привёл бы к существенным ошибкам – поскольку маленькая выборка способствовала бы выбору модели с простой функциональной формой зависимости переменных от X, – тогда как в рассмотренном примере подобная оценка (столбец 2 в таблице 2) оказалась наихудшей из рассмотренных.

Заключение

В работе был проведен обзор теоретических подходов к моделированию и идентификации причинно-следственных связей и структурных параметров моделей. Рассмотрены типичные примеры, в которых оценка структурного параметра осложняется присутствием наблюдаемых и ненаблюдаемых переменных, статистически связанных одновременно с переменной воздействия и целевой переменной.

Основной результат работы заключается в исследовании оценок неоднородных эффектов воздействия в данных с нелинейными зависимостями. Для этого предложен алгоритм генерации данных при помощи свёрточных нейронных сетей, и исследуются два разных подхода к оценке – методом байесовских сетей и методом двойного машинного обучения. Показано, что оба таких подхода в рассмотренном случае дают неточные оценки индивидуальных эффектов, но при этом точность улучшается при переходе к агрегированным значениям. В частности, достаточно точными оказываются оценки средних эффектов воздействия, полученные с по-

мощью метода двойного машинного обучения, а также в множественной регрессии с полиномиальной зависимостью от контрольных переменных. Такой результат позволяет с осторожностью относиться к известному в литературе подходу к выбору модели для оценки эффектов воздействия, в котором предлагается использовать небольшую выборку, полученную с помощью рандомизированного эксперимента, для настройки модели, оцениваемой на (большой) неэкспериментальной выборке.

Список литературы

[1] Pearl J., Mackenzie D., *The Book of Why: The New Science of Cause and Effect*, Basic Books, 2018.

[2] Chernozhukov V., et. al., *Causal ML and AI*, 2025, `https://causalml-book.org/`.

[3] Reichenbach H., *The Direction of Time*, University of California Press, 1991.

[4] Lewis D., "Causation", *J. Philos.*, **70**:17 (1973), 556–567.

[5] Galles D., Pearl J., "An Axiomatic Characterization of Causal Counterfactuals", *Found. Sc.*, **3**:1 (1998), 151–182.

[6] Holland P., "Statistics and Causal Inference", *Journal of the American Statistical Association*, **81** (1986), 941–970.

[7] Rubin D., "Which Ifs Have Causal Answers?", *J. Am. Stat. Assoc.*, **81** (1986), 961–962.

[8] Neyman J., "On the application of probability theory to agricultural experiments. Essay on principles", *Statistical science*, **5** (1923), 465–480.

[9] Robins J. M., "Addendum to "a new approach to causal inference in mortality studies with a sustained exposure period—application to control of the healthy worker survivor effect"", *Computers & Mathematics with Applications.*, **14** (1987), 923–945.

[10] Angrist J., Pieschke J.-S., *Mostly Harmless Econometrics*, Princeton University Press, 2014.

[11] Imbens G., Rubin D., *Causal Inference for Statistics, Social, and Biomedical Sciences*, Cambridge University Press, 2015.

[12] Wright P., *The Tariff on Animal and Vegatable Oils*, The Macmillan company, New York, 1928.

[13] Pearl J., "A Probabilistic Calculus of Actions", *in: Uncertainty in Artificial Intelligence*, 1994, 454–462.

[14] Ченцов А. М., Торопов Н. И., "Применение подхода двойного машинного обучения для задачи анализа зависимости между отклонениями от непокрытого паритета процентных ставок и степенью открытости экономики", *Труды МФТИ*, **16**:3 (2024), 72–81.

[15] Campbell D., Chentsov, A., "Breaking Badly: The Currency Union Effect on Trade", *J. Int. Money Finance*, **136** (2023), 1–16.

[16] Chernozhukov V., et. al., "Double/debiased machine learning for treatment and structural parameters", *Econometric J.*, **21** (2018), 1–68.

[17] Frisch R., Waugh F. V., “Partial Time Regressions as Compared with Individual Trends”, *Econometrica*, **1**:4 (1933), 387–401.

[18] Imbens G. W., Angrist J. D., “Identification and Estimation of Local Average Treatment Effects”, *Econometrica*, **62**:2 (1994), 467–475.

[19] Facure, M., *Causal Inference in Python*, O’Reilly Online Learning, 2023.

[20] Smirnov, A., *Striking Success: How Goals Shape Coach Bias in Football*, Book of Abstracts European Conference on Sports Economics `2024-08-bookofabstracts`, 2024 .

Статья поступила 6 июля 2025 г.

Notes on identification and estimation of causal effects using observational data

A. M. Chentsov, N. I. Toropov

The beginning of the 21st century in data science is characterized by the emergence of new interdisciplinary fields as well as an expansion of the general theoretical framework and development of statistical methods for solving novel problems. This article describes known approaches to addressing the problem of statistical identification of unidirectional (causal) relationships between variables using non-experimental data, and highlights distinctive features of statistical models employed for this purpose. It also considers a case study on generating data with nonlinear dependencies among variables through convolutional neural networks, where two different estimation techniques are investigated — Bayesian networks and double machine learning. The results show that both these approaches yield inaccurate estimates of individual effects in the considered scenario, and recommendations are provided regarding aggregated effect evaluation.

Keywords: causal identification, treatment effects, DAG-models, double machine learning, CATE.

References

[1] Pearl J., Mackenzie D., *The Book of Why: The New Science of Cause and Effect*, Basic Books, 2018.

[2] Chernozhukov V., et al., *Causal ML and AI*, 2025, `https://causalml-book.org/`.

[3] Reichenbach H., *The Direction of Time*, University of California Press, 1991.

[4] Lewis D., “Causation”, *J. Philos.*, **70**:17 (1973), 556–567.

[5] Galles D., Pearl J., "An Axiomatic Characterization of Causal Counterfactuals", *Found. Sc.*, **3**:1 (1998), 151–182.

[6] Holland P., "Statistics and Causal Inference", *Journal of the American Statistical Association*, **81** (1986), 941–970.

[7] Rubin D., "Which Ifs Have Causal Answers?", *J. Am. Stat. Assoc.*, **81** (1986), 961–962.

[8] Neyman J., "On the application of probability theory to agricultural experiments. Essay on principles", *Statistical science*, **5** (1923), 465–480.

[9] Robins J. M., "Addendum to "a new approach to causal inference in mortality studies with a sustained exposure period—application to control of the healthy worker survivor effect"", *Computers & Mathematics with Applications.*, **14** (1987), 923–945.

[10] Angrist J., Pischke J.-S., *Mostly Harmless Econometrics*, Princeton University Press, 2014.

[11] Imbens G., Rubin D., *Causal Inference for Statistics, Social, and Biomedical Sciences*, Cambridge University Press, 2015.

[12] Wright P., *The Tariff on Animal and Vegetable Oils*, The Macmillan company, New York, 1928.

[13] Pearl J., "A Probabilistic Calculus of Actions", *in: Uncertainty in Artificial Intelligence*, 1994, 454–462.

[14] Chentsov A.M., Toropov N.I., "Application of double machine learning to estimation of the effect of country opennes of deviations from uncovered interest parit", *Trudy MIPT*, **16**:3 (2024), 72–81 (In Russian).

[15] Campbell D., Chentsov A., "Breaking Badly: The Currency Union Effect on Trade", *J. Int. Money Finance*, **136** (2023), 1–16.

[16] Chernozhukov V., et al., "Double/debiased machine learning for treatment and structural parameters", *Econometric J.*, **21** (2018), 1–68.

[17] Frisch R., Waugh F. V., "Partial Time Regressions as Compared with Individual Trends", *Econometrica*, **1**:4 (1933), 387–401.

[18] Imbens G. W., Angrist J. D., "Identification and Estimation of Local Average Treatment Effects", *Econometrica*, **62**:2 (1994), 467–475.

[19] Facure, M., *Causal Inference in Python*, O'Reilly Online Learning, 2023.

[20] Smirnov, A., *Striking Success: How Goals Shape Coach Bias in Football*, Book of Abstracts European Conference on Sports Economics `2024-08-bookofabstracts`, 2024.

Received on July 6, 2025

Часть 2
Специальные вопросы теории интеллектуальных систем

Формальная верификация и устойчивость к цензуре протоколов византийского консенсуса на примере IBFT

К. В. Зиборов* , Н. С. Бондарев‡ , Ю. А. Янович§

В работе выполнен формальный анализ семейства протоколов Istanbul BFT (IBFT), объединяющий уязвимую первую версию, исправленные промежуточные варианты и модификацию, обеспечивающую устойчивость к цензуре, в рамках единого подхода к верификации. На основе спецификаций TLA+ с помощью средства проверки моделей TLC (i) строятся контрпримеры, демонстрирующие нарушение согласованности в первой версии IBFT, (ii) подтверждаются свойства безопасности и живучести для исправленного варианта QBFT, и (iii) вводится строгая формализация устойчивости к цензуре для византийских протоколов с лидером как ограниченной гарантии качества цепочки, для которой стандартная ротация лидера может давать нарушение. В работе также предлагается и формально верифицируется правило выбора лидера *f-skip*, обеспечивающее устойчивость к цензуре без ослабления исходных гарантий безопасности и живучести.

Ключевые слова: IBFT, QBFT, BFT-консенсус, формальная верификация, TLA+, TLC, устойчивость к цензуре..

1. Введение

В системах распределённого реестра фундаментальной задачей является поддержание надежности системы в условиях наличия неисправных

* *Зиборов Кирилл Викторович* — аспирант каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: krl.ziborov@gmail.com, ORCID: 0000-0002-5676-9105.

Ziborov Kirill Viktorovich — graduate student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

‡ *Бондарев Никита Сергеевич* — студент каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: bns150101@gmail.com, ORCID: 0009-0000-5240-0178.

Bondarev Nikita Sergeevich — student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

§ *Янович Юрий Александрович* — кандидат ф.-м. наук, Сколтех, e-mail: y.yanovich@skoltech.ru, ORCID: 0000-0003-4651-7585.

Yanovich Yury Aleksandrovich — Assistant Professor, Skolkovo Institute of Science and Technology.

или враждебных процессов. Протоколы консенсуса решают эту задачу, позволяя узлам соглашаться о единой упорядоченной последовательности транзакций. В то время как ранние публичные блокчейн-сети, самая известная из которых Bitcoin [15], использовали вычислительно затратный алгоритм Proof-of-Work, современные блокчейн-системы — как публичные, так и приватные — всё чаще используют устойчивые к византийским ошибкам (BFT) алгоритмы консенсуса. Корпоративные и приватные блокчейны используют BFT-алгоритмы с детерминированной окончательностью блоков при низких затратах ресурсов, тогда как многие открытые Proof-of-Stake системы применяют BFT-консенсус только после выбора комитета валидаторов [22, 23].

Протокол Practical Byzantine Fault Tolerance (PBFT) [24] заложил основу для этого класса алгоритмов, в то время как Istanbul BFT (IBFT) адаптировал PBFT к Ethereum-подобным разрешительным сетям [25]. IBFT организует выполнение в раунды, управляемые лидером, с двухфазным голосованием, достигая окончательности посредством фиксированного набора валидаторов, и используя Proof-of-Authority для устойчивости к атакам Сивиллы. Его принятие в корпоративных блокчейнах, таких как Quorum, сделало IBFT критическим компонентом реальных распределённых систем и репрезентативным примером BFT-консенсуса в производственных средах.

Однако требования к устойчивости консенсуса выходят за рамки классических свойств безопасности и живучести. В открытых блокчейнах устойчивость к цензуре, то есть гарантия того, что ни одна сущность не может произвольно исключать транзакции, является фундаментальным условием, тесно связанным с формальным свойством качества цепочки, которое обеспечивает справедливое распределение прав создания блоков [2]. Однако это свойство не является безусловным; оно может быть нарушено из-за дефектов протокола, регуляторного давления или экономических стимулов, таких как Maximal Extractable Value (MEV) [3, 4]. Хотя существующие исследования количественно оценили цензуру и её влияние на безопасность в системах Proof-of-Work (PoW) и Proof-of-Stake (PoS) [37], анализ BFT-консенсусов, управляемых лидером, лежащих в основе многих приватных сетей и комитетных стадий открытых сетей, остаётся недостаточно исследованным. В таких протоколах злонамеренный лидер получает привилегированное положение для цензурирования транзакций. Это создаёт необходимость в формализации и обеспечении устойчивости к цензуре в тех самых BFT-протоколах, которым доверяют финальность блоков в корпоративных и развивающихся PoS-экосистемах.

Несмотря на широкое принятие, IBFT прошёл путь через несколько версий со значительными доработками, направленными на корректность

протокола. Ранние реализации содержали критические уязвимости, нарушающие как свойство безопасности (safety), так и живучести (liveness) [27]. Последующие исправления устраняли эти проблемы, однако в литературе отсутствует единый формальный анализ, связывающий развитие протокола от уязвимых к верифицированным реализациям. Более того, хотя безопасность и живучесть получили внимание, свойство устойчивости к цензуре — потенциальная уязвимость, влияющая не только на IBFT, но и на BFT-протоколы с лидером в целом — остаётся неформализованным и непроверенным, создавая риски для систем, которые полагаются на эти механизмы консенсуса.

Эта работа закрывает эти пробелы посредством трёх основных вкладов, которые могут быть расширены за пределы IBFT:

1. **Всесторонний формальный анализ развития IBFT:** Нами был проведен формальный анализ промежуточных версий IBFT и представлены как верификация исправленных реализаций, так и контрпримеры, демонстрирующие конкретные угрозы в уязвимых версиях. Таким образом, был проведен полный формальный анализ истории развития этого популярного протокола.
2. **Новое формализованное определение устойчивости к цензуре:** Мы вводим первое строгое формальное определение устойчивости к цензуре для византийских протоколов консенсуса с лидером и доказываем, что стандартный IBFT нарушает это свойство. Эта уязвимость может быть обобщена на аналогичные протоколы с ротацией лидера, где злонамеренные лидеры могут бесконечно исключать транзакции.
3. **Модификация F-Skip с формальной верификацией:** Мы предлагаем и формально верифицируем лёгкую модификацию *f-skip*, которая обеспечивает устойчивость к цензуре при сохранении всех исходных гарантий безопасности и живучести. Этот механизм применим к любому основанному на лидере BFT-протоколу и предоставляет практическое, формально верифицированное решение проблемы цензуры в приватных блокчейнах и комитетном BFT-консенсусе в Proof-of-Stake системах.

Мы использовали TLA+ [14] для спецификации и средство проверки моделей TLC для исчерпывающей формальной проверки протокола. Результаты продемонстрированы на минимальных конфигурациях, которые охватывают существенное поведение протокола и при этом поддаются формальному анализу. Наш вклад даёт как специфичные для протокола

IBFT исправления, так и общие выводы, применимые к проектированию основанного на лидере консенсуса в современных блокчейн-системах.

Остальная часть работы организована следующим образом: в Разделе 2 обсуждаются связанные работы; Раздел 3 детализирует схему работы протокола IBFT; Раздел 4 представляет наши формальные спецификации и свойства; в Разделе 5 мы представляем формальный анализ версий IBFT на основе наших спецификаций; Раздел 6 детализирует подход к верификации и результаты; и Раздел 7 завершает работу направлениями будущих исследований.

2. Связанные работы

Формальная верификация протоколов консенсуса значительно эволюционировала от классических распределённых алгоритмов до блокчейн-специфичных дизайнов. Мы группируем связанные работы по четырём направлениям: методология формальной верификации; верификация протоколов консенсуса, включая свойства за пределами классических безопасности и живучести; анализ протокола IBFT и его модификаций; а также исследования устойчивости к цензуре в консенсусах.

2.1. Методы формальной верификации

Проверка моделей и интерактивное доказательство теорем составляют два основных подхода к верификации протоколов консенсуса. Инструменты проверки моделей, такие как TLA+/TLC [14], позволяют исчерпывающе исследовать модели с конечным числом состояний, генерируя контрпримеры в случае нарушения свойств протокола. Экосистема TLA+ была расширена инструментом Apalache [32], который даёт дополнительные возможности верификации и символьной проверки моделей. Недавние разработки также включают Quint [20] — современный язык спецификации, который компилируется в TLA+. Quint использовался для верификации протокола блокчейн-консенсуса ChonkyBFT.

Подход, связанный с интерактивным доказательством теорем, представленный работами в Coq [30] и Isabelle/HOL, обеспечивает машинопроверяемые доказательства корректности при явных предположениях, но проигрывает в трудоемкости. Velisarios [29] механизирует доказательства безопасности PBFT в Coq, тогда как Li и др. [12] формально верифицируют протокол Casper для Ethereum. Наша работа использует метод проверки моделей на TLA+, который балансирует между автоматизацией и строгой верификацией, являясь достаточным для анализа эволюции протокола и позволяя генерировать конкретные контрпримеры.

2.2. Верификация протоколов консенсуса

Формальная верификация широко применялась к классическим распределённым алгоритмам, таким как Paxos [16] и Raft [28]. Многие созданные для блокчейнов протоколы, включая Tendermint [33], HotStuff [34] и TetraBFT [17], получили детальный формальный анализ. LibraBFT [35] представляет гибридную методологию верификации, сочетая проверку моделей с доказательством теорем для повышенной надежности.

Появление блокчейнов, использующих Proof-of-Stake (PoS), стимулировало усилия по верификации подобных протоколов. В частности, Ethereum 2.0 Beacon Chain был предметом нескольких исследований формальной верификации. Rashid и др. [8, 9] используют средство проверки моделей SPIN для верификации финализации чекпойнтов и процесса выхода валидаторов. Cassez и др. [10] применяют Dafny для верификации отсутствия ошибок времени выполнения в эталонной реализации Beacon Chain, тогда как Afzaal и др. [11] используют инструмент PAT для аналогичных целей. Эти работы демонстрируют растущую зрелость формальных методов для PoS-консенсуса.

Недавние работы расширили фокус за пределы базовых свойств безопасности и живучести, формализуя дополнительные гарантии консенсуса. Pass и Shi [13] вводят формальные определения справедливости в гибридных моделях консенсуса. Однако *устойчивость к цензуре*, в частности, для BFT-протоколов с лидером, используемых как в корпоративных блокчейнах, так и в комитетных PoS-системах, остаётся недостаточно формализованной. Существующие определения справедливости обычно связаны с упорядочиванием транзакций, а не гарантиями включения транзакций, оставляя разрыв в формальных моделях устойчивости к цензуре как отдельного свойства консенсуса.

2.3. Анализ и верификация IBFT

Существующие работы по анализу IBFT в основном состоят из аудитов безопасности, выявляющих уязвимости в ранних версиях [27]. Saltini и Hyland-Wood предоставляют наиболее полный анализ на сегодняшний день, выявляя проблемы безопасности и живучести в раннем IBFT и предлагая исправления. Однако их анализ ограничен первой версией IBFT и не предоставляет единого рассмотрения по промежуточным версиям, а также не изучает устойчивость к цензуре — критическое свойство для корпоративных блокчейн-приложений, где исключение транзакций создаёт бизнес-риски. В последующей работе [19] они также представили IBFT 2.0, пересмотренный вариант, который адресует проблемы безопасности и живучести. Модификация QBFT на основе варианта IBFT, описанного

Moniz[21], была специфицирована и частично верифицирована в Dafny. Доступные артефакты [18] сосредоточены на безопасности и не включают спецификацию или доказательство живучести, они также не связывают верифицированный дизайн QBFT с более ранними версиями IBFT.

Насколько нам известно, ни одна предшествующая работа не предоставляет: (1) полный путь формальной верификации через эволюцию IBFT от уязвимых к исправленным реализациям, (2) строгое формальное определение устойчивости к цензуре, применимое к основанным на лидере BFT-протоколам, или (3) формально верифицированную модификацию, такую как *f-skip*, которая обеспечивает устойчивость к цензуре при сохранении основных свойств протокола. Наша работа закрывает эти пробелы, предоставляя при этом выводы, применимые за пределами IBFT к более широкому классу основанных на ротации лидера BFT-протоколов консенсуса, используемых в современных блокчейн-системах.

2.4. Устойчивость к цензуре и безопасность блокчейнов

Изучение устойчивости к цензуре эволюционировало от концептуальной цели публичных систем к количественно определяемому свойству безопасности с прямым воздействием на целостность распределённого реестра. Формальная основа устойчивости к цензуре часто связывается с *качеством цепочки*, метрикой, которая гарантирует минимальную долю блоков, произведённых честными участниками [36]. Zhang и др. [37] устанавливают важную структуру для оценки качества цепочки в PoW-протоколах, показывая, что идеальное качество цепочки остаётся недостижимым и что его отсутствие позволяет front running и другие атаки. Эта формальная связь между качеством цепочки и безопасностью критична для нашей работы. Переходя от теории к практике, Wahrstätter и др. [2] предоставляют эмпирическое исследование, формализуя и количественно оценивая цензуру в Ethereum. Их вывод о том, что регуляторные санкции могут существенно компрометировать нейтральность блокчейна и увеличивать задержку транзакций, подчёркивает цензуру как ощутимую угрозу безопасности, а не только теоретическую проблему.

Дизайн протоколов предлагает два основных пути усиления устойчивости к цензуре. Во-первых, *безлидерные* дизайны консенсуса, такие как DBFT [5, 38], стремятся устранить единую точку отказа (и цензуры), присущую основанным на лидере протоколам, используя децентрализованную сборку блоков. Во-вторых, механизмы вроде Proposer-Builder Separation (PBS) в Ethereum [4] пытаются ограничить привилегии одного производителя блока. Однако, как показывают Heimbach и др., такие дизайны могут непреднамеренно централизовать привилегии производителя

блоков, потенциально усугубляя, а не уменьшая цензуру. Это подчёркивает сложность компромиссов в проектировании цензуроустойчивых систем.

Угроза цензуры внутренне связана с Maximal Extractable Value (MEV). Злонамеренные участники могут цензурировать транзакции, чтобы обеспечить прибыльный front running или sandwich-атаки [3]. Эффективность аукционов сборки блоков [7] и распространённость MEV-возможностей даже на сайдчейнах [1, 6] создают мощные финансовые мотивы для исключения транзакций. Эта экономическая составляющая требует, чтобы любая формальная модель устойчивости к цензуре для современных блокчейнов учитывала рациональных, ориентированных на прибыль противников, а не только произвольные византийские отказы.

Наша работа опирается на этот фундамент, но смещает фокус на *основанный на лидере BFT-консенсус*. В отличие от PoW/PoS-систем, изученных выше, BFT-протоколы предоставляют немедленную окончательность блоков и являются ключевым механизмом в работе корпоративных блокчейнов и комитетных стадий PoS-протоколов. Мы формализуем устойчивость к цензуре в рамках этой конкретной модели, доказывая нарушение этого свойства в стандартных протоколах с ротацией лидера, таких как IBFT, и предлагаем верифицируемую модификацию (*f-skip*), которая не требует радикального архитектурного перехода к безлидерному дизайну, тем самым предлагая практический путь к более сильным гарантиям для существующих, широко развёрнутых систем.

3. Протокол консенсуса IBFT

3.1. Модель системы и предположения

IBFT работает в частично синхронной системе с N детерминированными процессами (валидаторами), из которых не более F являются византийскими, требуя $N \geq 3F + 1$ для устойчивости. Процессы обмениваются сообщениями через надёжные, аутентифицированные каналы с гарантией доставки в перспективе, предполагая неизвестную, но конечную границу доставки сообщений Δ. Криптографические примитивы обеспечивают целостность сообщений и предотвращают их подделку. Каждый процесс локально хранит текущую высоту блока h и номер раунда v внутри этой высоты.

3.2. Основная механика протокола

IBFT следует трёхфазному процессу принятия значения, адаптированному из PBFT, выполняемому на каждой высоте:

1. **Pre-Prepare:** Назначенный лидер для раунда v предлагает блок для высоты h, рассылая всем валидаторам сообщение `PRE-PREPARE`, содержащее блок.
2. **Prepare:** Валидаторы проверяют предложенный блок и рассылают сообщения `PREPARE` при принятии. Блок становится *prepared* (подготовленным), когда валидатор получает $2F + 1$ совпадающих сообщений `PREPARE`, образующих кворум.
3. **Commit:** После подготовки блока валидаторы рассылают сообщения `COMMIT`. Блок становится *committed* (финализированным) после получения $2F + 1$ совпадающих сообщений `COMMIT`, после чего он необратимо добавляется в блокчейн.

Свойство пересечения кворумов (требуются $2F + 1$ сообщений) гарантирует, что любые два кворума пересекаются хотя бы в одном корректном процессе, гарантируя согласие между корректными валидаторами. Эта граница устойчивости $3F + 1$ фундаментальна для BFT-консенсуса.

3.3. Протокол смены раунда

Когда текущий лидер не прогрессирует (что определяется локальными таймаутами), валидаторы инициируют смену раунда. Критическое различие между версиями IBFT относится к их механизмам смены раунда:

3.3.1. IBFT 1.0 (уязвимая версия) [25]

Валидаторы отправляют сообщения `ROUND-CHANGE`, содержащие только номер нового раунда. Новый лидер предлагает произвольный блок, не учитывая ранее подготовленные блоки, что приводит к нарушениям свойства безопасности, когда конфликтующие блоки могут быть зафиксированы на одной и той же высоте.

3.3.2. IBFT 2.0 (исправленная промежуточная версия) [19]

Валидаторы прикрепляют (возможно пустой) *prepared certificate* к сообщениям `ROUND-CHANGE`, добавляя блок, для которого они наблюдали кворум `PREPARE`. Новый лидер собирает кворум `ROUND-CHANGE` (сертификат смены раунда) и должен предложить блок, на который ссылается

сертификат с наибольшим значением подготовленного раунда, если он существует; иначе он может предложить любой новый валидный блок, включая сертификат, обосновывающий предложение.

3.3.3. QBFT (исправленная промежуточная версия) [21]

Валидаторы включают пары (pr, pv) в сообщения `ROUND-CHANGE`, где pr — наибольший номер подготовленного раунда, а pv — соответствующее подготовленное значение. Новый лидер должен предложить значение из наиболее подготовленного состояния, определяемого оператором `HighestPrepared`, если таковое существует. Это обеспечивает безопасность при смене раунда за счет правильной передачи состояния.

3.3.4. Современные верифицированные реализации

Последние версии IBFT включают комплексные исправления и были формально верифицированы[18]. Таким образом, было подтверждено выполнение свойств безопасности и живучести, однако все ещё отсутствует анализ дополнительных свойств, таких как устойчивость к цензуре.

3.4. Выбор лидера и ротация

IBFT использует детерминированный выбор лидера. Это гарантирует то, что каждый валидатор периодически становится лидером согласно предсказуемому расписанию. Хотя это обеспечивает простоту и справедливость в честных условиях, такая ротация позволяет византийским лидерам цензурировать транзакции во время своих периодов лидерства. Это может влиять не только на IBFT, но и на любой основанный на лидере BFT-протокол с предсказуемой ротацией. Детерминированная природа позволяет злонамеренным валидаторам предугадывать, когда они будут лидером, и стратегически исключать транзакции.

3.5. IBFT в более широком ландшафте консенсуса

Будучи производным от PBFT протоколом с немедленной окончательностью блоков, IBFT представляет класс алгоритмов консенсуса, используемых как в приватных блокчейнах, так и на уровне комитетного слоя консенсуса в Proof-of-Stake системах. Его эволюция от уязвимых к исправленным реализациям отражает вызовы в дизайне BFT-протоколов в целом, делая его ценным примером для формального анализа с более широкими последствиями для основанного на лидере византийского консенсуса.

4. Формальная спецификация и определения свойств

Этот раздел детализирует нашу методику формальной верификации, структуру наших спецификаций TLA+ и точные определения свойств, которые мы проверяем. Наш подход спроектирован так, чтобы быть одновременно специфичным для IBFT и обобщаемым на более широкий класс основанных на лидере BFT-протоколов консенсуса.

4.1. Методология формальной верификации

Мы используем язык спецификации TLA+ и средство проверки моделей TLC для нашего формального анализа. Этот подход предоставляет строгие математические рамки для моделирования конкурентных и распределённых систем, позволяя исчерпывающее исследование пространства состояний при ограниченных параметрах. Наша методика структурирована следующим образом:

Разработка спецификации: Мы создаём три отдельные TLA+ модуля, каждый моделирует критический этап эволюции IBFT:

1. `IBFT1_0.tla` моделирует уязвимый исходный протокол, опуская передачу состояния при смене раунда;
2. `QBFT.tla` моделирует исправленный промежуточный протокол, который включает полный механизм смены раунда;
3. `QBFT_FSkip.tla` моделирует исправленный протокол, усиленный нашим новым правилом выбора лидера *f-skip* для обеспечения устойчивости к цензуре.

Формализация свойств: Мы определяем базовые BFT-свойства (Agreement, Validity, Termination), а также новое формализованное определение устойчивости к цензуре как инварианты и темпоральные свойства TLA+.

Проверка моделей: Мы используем TLC, чтобы проверить эти свойства на минимальных, но репрезентативных конфигурациях системы (например, $N = 4$, $F = 1$). Эта ограниченная верификация достаточна, чтобы захватить существенные взаимодействия протокола и сгенерировать краткие контрпримеры для нарушений.

Обобщение: Хотя спецификации опираются на механику IBFT, определённые свойства, уязвимости и модификации (в частности *f-skip*) сформулированы для более широкой применимости к основанным на лидере BFT-протоколам.

4.2. Архитектура базовой спецификации

Все три спецификации TLA+ разделяют общую архитектурную основу, моделируя систему из N детерминированных процессов (валидаторов) с не более чем F византийскими отказами ($N \geq 3F + 1$). Общие элементы представлены в Таблице 1.

Таблица 1: Основные элементы спецификации TLA+

Тип элемента	**Описание**
Переменные состояния	
`view_num[p]`	Текущий номер раунда для процесса p
`height`	Текущая высота блока, в которой принимается значение
`pr[p]`, `pv[p]`	Наибольший подготовленный раунд и значение для процесса p
`messages`	Мультимножество отправленных сообщений в протоколе (`PRE-PREPARE`, `PREPARE`, `COMMIT`)
`RCmessages`	Мультимножество сообщений `ROUND-CHANGE` для смены раунда
`decision[h][p]`	Значение, принятое процессом p на высоте h
`blockLeader[h]`	Лидер, чьё предложение было принято на высоте h
Константы и параметры	
N, F	Параметры устойчивости системы ($N \geq 3F + 1$)
`QUORUM`	Минимум сообщений для прогресса ($2F + 1$)
`Corr`	Множество корректных процессов, $\subseteq \{1, \ldots, N\}$, $\lvert\texttt{Corr}\rvert \geq N - F$
`Values`	Множество валидных полезных нагрузок блоков

Спецификации действий: Каждый модуль определяет набор атомарных переходов состояния, моделирующих действия протокола (например, `UponPrePrepare`, `UponPrepared`, `UponCommit`, `UponQRC` для обработки кворума сообщений смены раунда) и спецификацию допустимого византийского поведения.

Спецификации расходятся в двух ключевых аспектах: логике обоснования и выполнения смены раунда и правиле выбора лидера для заданной высоты и раунда.

4.3. Формальные определения свойств

Мы формально определяем и проверяем следующие свойства. Каждое свойство выражается как соответствующая формула TLA+, проверяемая TLC.

4.3.1. Согласованность (Safety)

Никакие два корректных процесса не принимают разные значения на одной и той же высоте. Это обеспечивает согласованность консенсуса. В TLA+ мы проверяем инвариант[1]:

```
Agreement ==
  \A h \in 1..MaxHeight, p, q \in Corr :
    (decision[h][p] /= NilValue /\ decision[h][q] /= NilValue)
    => decision[h][p] = decision[h][q]
```

4.3.2. Корректность (Safety)

Каждое принятое значение должно быть корректным (или пустым). Это предотвращает принятие произвольных значений:

```
Validity ==
  \A h \in 1..MaxHeight, p \in Corr :
    decision[h][p] \in Values \cup {NilValue}
```

4.3.3. Завершаемость (Liveness)

Протокол в конечном итоге примет значение для каждой высоты. Формально, для каждой высоты h в конечном итоге верно, что корректный процесс принимает значение $v \in \text{Values}$ на h. В TLA+ мы проверяем темпоральное свойство:

```
Termination == <> \A h \in 1..MaxHeight :
    \E v \in Values : Committed(h, v)
```

[1] Далее мы используем стандартные обозначения TLA+. Например, \/ — это логическое И, ′ обозначает значение в следующем состоянии, а \A — это универсальный квантор $\forall$. <> F — темпоральный оператор *eventually*, означающий, что всегда найдется состояние, в котором F будет выполнено.

4.3.4. Устойчивость к цензуре: новое формализованное определение

Стандартные BFT-свойства не предусматривают исключение транзакций злонамеренным лидером. Мы вводим первое строгое определение устойчивости к цензуре для основанных на лидере BFT-протоколов консенсуса.

Введем определение *устойчивости к цензуре для основанного на лидере BFT-консенсуса*:

Основанный на лидере византийский протокол консенсуса является *устойчивым к цензуре* с размером окна $W(N, F)$, если в любой последовательности из W последовательно зафиксированных блоков по крайней мере один блок был предложен корректным лидером, где W является функцией от N и F.

Это свойство гарантирует, что злонамеренные лидеры не могут бесконечно доминировать в производстве блоков. Оно непосредственно соответствует гарантии *качества цепочки* как минимум $1/W$ в последовательности блоков. Протокол с предсказуемой ротацией лидера и без дополнительных механизмов (как стандартный IBFT) имеет неограниченный W, так как византийские лидеры могут подряд возглавлять произвольно большое число блоков, нарушая устойчивость к цензуре.

Формализация как инварианта: Для ограниченного `MaxHeight` мы проверяем инвариант, что в любом окне из $F + 1$ последовательно зафиксированных высот хотя бы один лидер корректен. Это выражено в TLA+ как:

```
HonestLeaderInFPlusOneBlocks ==
  \A h \in 1..(MaxHeight - F) :
    (\A i \in 0..F : blockLeader[h + i] /= -1)
    \* All heights in the window are committed
      => \E i \in 0..F : blockLeader[h + i] \in Corr
```

4.4. Модификация F-Skip: обобщаемый механизм

Чтобы обеспечить устойчивость к цензуре, мы предлагаем модификацию *f-skip*, лёгкое правило, применимое к любому основанному на лидере BFT-протоколу с предсказуемой ротацией.

Базовое правило: Валидатор не может быть лидером на высоте h, если он был лидером на любой из предшествующих F зафиксированных высот (высоты $h - F$ по $h - 1$).

Формальная реализация: Это реализуется через два ключевых расширения спецификации:

Отслеживание состояния: Массив `blockLeader` записывает лидера для каждой зафиксированной высоты.

Динамическая допустимость лидеров:

```
AllowedLeaders(h) ==
    AllProcs \ {blockLeader[i] : i \in Max(1, h-F)..(h-1)}

Leader(h, v) ==
    LET candidates == AllowedLeaders(h) IN
    IF candidates = {} THEN DefaultLeader(h, v)
    ELSE KthMin(candidates, ((v-1) % |candidates|) + 1)
```

Лидер для (h, v) затем выбирается детерминированной ротацией из множества `AllowedLeaders`(h).

Гарантия: При не более чем F византийских процессах принцип Дирихле гарантирует, что в любой последовательности из $F + 1$ последовательно зафиксированных блоков хотя бы один лидер должен быть корректным. Это ограничивает окно цензуры как $W = F + 1$. Модификация изменяет только выбор лидера, сохраняя все исходные свойства безопасности и живучести, зависящие от базовой логики принятия значения и смены раунда.

5. Анализ эволюции протокола IBFT

Мы применяем методику из Раздела 4 для анализа трёх ключевых версий IBFT. Для каждой мы описываем ключевые характеристики модели TLA+, проверяемые свойства и результаты проверки модели.

5.1. Уязвимый IBFT 1.0: спецификация и нарушение безопасности

Обзор модели (`IBFT1_0.tla`): Эта спецификация моделирует исходный, уязвимый протокол. Причина уязвимости кроется в отсутствии передачи состояния при смене раунда. Когда таймаут вызывает смену раунда, валидаторы увеличивают свой номер раунда, но не отправляют сообщений `ROUND-CHANGE`, передающих их подготовленное состояние (pr, pv).

Ключевой дефект в спецификации: Действие смены раунда неполное:

```
Timeout(p) ==
    /\ processState' = [processState EXCEPT ![p].view_num =
       processState[p].view_num + 1]
    /\ UNCHANGED <<messages>>  \* No ROUND-CHANGE messages sent
```

Следовательно, новый лидер не имеет информации о ранее подготовленных блоках и может предложить произвольный новый блок, даже если другой блок был подготовлен в предыдущем раунде.

Результаты верификации: Наша проверка модели выявила, что:

- **Agreement нарушается**. TLC генерирует контрпример трассы, где византийский лидер отправляет конфликтующие сообщения `PRE-PREPARE` в раунде 1. Корректные процессы разделяются в ходе `PREPARE`-фазы. Последующие корректные лидеры, не зная об этом конфликте из-за отсутствия передачи состояния, предлагают разные значения, что приводит к двум различным принятым значениям на одной и той же высоте.
- **Validity** выполнялось тривиально, так как все предложенные значения принадлежат множеству `Values`.

Это подтверждаёт известную уязвимость безопасности IBFT 1.0. Контрпример конкретно демонстрирует, как протокол может нарушать свойство согласованности консенсуса.

5.2. Исправленный QBFT: безопасность и живучесть без устойчивости к цензуре

Обзор модели (`QBFT.tla`): Эта спецификация моделирует исправленную версию протокола (QBFT). Существенное исправление заключается во введении полноценного протокола смены раунда, где сообщения `ROUND-CHANGE` передают наиболее подготовленное состояние отправителя (pr, pv).

Ключевой механизм — `HighestPrepared`, безопасность протокола зависит от этого оператора, который обрабатывает кворум Q сообщений `ROUND-CHANGE`:

```
HighestPrepared(Q) ==
    IF Q = {} THEN <<NilView, NilValue>>
    ELSE
        LET prSet == {m.pr : m \in Q}
            maxPr == Maximum(prSet) IN
        IF maxPr = NilView THEN <<NilView, NilValue>>
        ELSE
            LET msgsAtMax == {m \in Q : m.pr = maxPr} IN
            LET rep == CHOOSE m \in msgsAtMax : TRUE IN
            <<maxPr, rep.pv>>
```

Новый лидер раунда v должен предложить компоненту значения (второй элемент) кортежа, возвращаемого `HighestPrepared`(Q), где Q — кворум сообщений `ROUND-CHANGE` для раунда v. Это гарантирует, что любой ранее подготовленный блок будет предложен повторно, сохраняя согласованность при смене раунда.

Результаты верификации: Наша проверка модели подтвердила, что:

- **Agreement** и **Validity выполняются**. Проверка TLC подтверждает за счёт явного перебора пространства состояний, что механизм `HighestPrepared` предотвращает принятие конфликтующих блоков на одной и той же высоте.
- **Termination выполняется**. Протокол в конечном итоге фиксирует блок на каждой высоте при условиях частичной синхронности и справедливости лидера.
- **Устойчивость к цензуре** (`HonestLeaderInFPlusOneBlocks`) **нарушается**. TLC находит контрпример, где один и тот же византийский процесс записывается как `blockLeader` для двух последовательных высот (например, высоты 1 и 2 при $F = 1$). Это возможно, потому что стандартная round-robin ротация лидера (`Leader`$(h, v) = ((h + v - 2) \bmod N) + 1$) не накладывает ограничений на то, как часто византийский валидатор может быть лидером.

Таким образом, для QBFT выполнены классические BFT-свойства безопасности и живучести, но не обеспечена устойчивость к цензуре. Один злонамеренный валидатор, являясь лидером, может бесконечно цензурировать транзакции, предлагая пустые или выборочные блоки.

5.3. QBFT с F-Skip: безопасный, живой и устойчивый к цензуре вариант

Обзор модели (`QBFT_FSkip.tla`)**:** Эта спецификация расширяет `QBFT.tla` исключительно изменением логики выбора лидера для реализации правила *f-skip* (Раздел 4.4). Базовая логика принятия значения и смены раунда остаётся неизменной.

Ключевая модификация — динамическая допустимость лидера: Лидер больше не определяется постоянной формулой. Вместо этого множество допустимых лидеров для высоты h исключает тех, кто был лидером на предыдущих F блоках:

```
AllowedLeaders(h) ==
    AllProcs \ {blockLeader[i] : i \in Max(1, h-F)..(h-1)}
```

```
Leader(h, v) ==
    LET candidates == AllowedLeaders(h) IN
    IF candidates = {} THEN DefaultLeader(h, v)
    ELSE KthMin(candidates, ((v-1) % |candidates|) + 1)
```

Результаты верификации: Наша проверка модели верифицировала, что:

- **Agreement**, **Validity** и **Termination выполняются**. Проверка TLC подтверждает, что модификация *f-skip* не затрагивает безопасность и живучесть базового протокола. Механизм `HighestPrepared` остаётся эффективным.
- **Устойчивость к цензуре (`HonestLeaderInFPlusOneBlocks`) выполняется**. Для ограниченной модели (`MaxHeight` $= 3$, $F = 1$) TLC подтверждает сохранение инварианта того, что в каждом окне из $F + 1$ (то есть 2 в проверяемой конфигурации) последовательных принятых блоков хотя бы один лидер честный. Это формально доказывает, что *f-skip* обеспечивает желаемое свойство устойчивости к цензуре.

Таким образом, модификация *f-skip* успешно преобразует протокол в устойчивый к цензуре, сохраняя при этом свойства безопасности и живучести. Она предоставляет практическое, формально верифицированное решение уязвимости цензуры, присущей стандартным основанным на лидере BFT-протоколам, таким как IBFT.

6. Формальная верификация и результаты

6.1. Настройки и методология верификации

Мы верифицировали все версии протокола с помощью средства проверки моделей TLC версии 1.7.4 в ограниченной конфигурации, обеспечивающей исчерпывающий явный перебор пространства состояний при фиксированных значениях параметров. Такая конфигурация позволяет воспроизводимо обнаруживать уязвимости, получать минимальные контрпримеры и одновременно сохранять вычислительную реализуемость экспериментов.

Аппаратно-программная среда. Эксперименты выполнялись на машине с процессором Intel Core i7-12700H (2,70 ГГц), 8 ГБ ОЗУ и SSD объёмом 1 ТБ. Для запуска использовалась среда TLA+ Toolbox 1.7.4.

Параметры проверки. Мы использовали минимальные конфигурации, которые удовлетворяют условию устойчивости $N \geq 3F + 1$. В

таблице 2 приведены значения параметров, использованные для каждого варианта протокола, включая оптимизированные настройки для различных классов свойств.

Таблица 2: Параметры верификации для каждого варианта протокола

Параметр	**IBFT 1.0**	**QBFT (без f-skip)**	**QBFT (с f-skip)**
N (валидаторы)	4	4	4
F (византийские)	1	1	1
Корректные процессы (`Corr`)	$\{2, 3, 4\}$	$\{2, 3, 4\}$	$\{2, 3, 4\}$
Византийский процесс	$\{1\}$	$\{1\}$	$\{1\}$
`Values`	`{"A "B"}`	`{"A "B"}` / `{"A"}`*	`{"A "B"}` / `{"A"}`*
`MaxHeight`	1	2	2 / 3*
`MaxView`	4	3 / 4*	3 / 5*
`MaxMessages`	50	30 / 40*	30 / 50*

*Разные конфигурации использовались для верификации разных свойств (см. ниже).

Стратегия верификации. Для каждой версии протокола мы проверяли четыре класса свойств, сформулированных в разделе 4.3: (1) корректность типов (`TypeOK`) как базовую проверку согласованности модели; (2) *Agreement* и *Validity* как инварианты безопасности; (3) *Termination* как темпоральное свойство живучести при допущениях справедливости (частичная синхронность и в конечном итоге прогрессирующий лидер); (4) инвариант устойчивости к цензуре `HonestLeaderInFPlusOneBlocks`. Для отдельных классов свойств использовались различные наборы параметров, позволяющие существенно уменьшать пространство состояний без изменения семантики проверяемого свойства (например, сокращение множества `Values` при проверке устойчивости к цензуре).

6.2. IBFT 1.0: подтверждена уязвимость безопасности

Для IBFT 1.0 применялись два набора параметров, отдельно оптимизированные под проверку `TypeOK` и поиск контрпримера для *Agreement*:

- TypeOK: `Values` $= \{"A"\, "B"\}$, `MaxView` $= 4$, `MaxHeight` $= 1$, `MaxMessages` $= 50$
- Agreement: `Values` $= \{"A"\, "B"\}$, `MaxView` $= 3$, `MaxHeight` $= 1$, `MaxMessages` $= 50$

Проверяемые свойства. Проверялись *Agreement* и *Validity* (см. раздел 4.3).

Результаты. TLC обнаружил контрпример, нарушающий *Agreement*, после перебора порядка $5{,}9 \cdot 10^5$ состояний за 7 с. Трасса контрпримера

иллюстрирует типичное нарушение безопасности: византийский процесс 1 рассылает конфликтующие сообщения `PRE-PREPARE` со значениями “A” и “B” в раунде 1; корректные процессы разделяются по подготовленным значениям; далее, из-за отсутствия передачи подготовленного состояния при смене раунда, последующие лидеры не обязаны переиспользовать уже подготовленное значение и могут предложить альтернативу. В результате на одной и той же высоте фиксируются несовместимые решения (например, процесс 2 фиксирует “A”, а процесс 3 фиксирует “B”). Таким образом, эксперименты воспроизводимо подтверждают известную уязвимость IBFT 1.0: отсутствие передачи состояния при смене раунда допускает конфликтующие принятые значения.

6.3. QBFT без f-skip: безопасность выполняется, но устойчивость к цензуре нарушается

Для QBFT без f-skip использовались два набора параметров:

- Agreement/Validity: $\texttt{Values} = \{\text{"}A\text{"}\,\text{"}B\text{"}\}$, $\texttt{MaxView} = 3$, $\texttt{MaxHeight} = 2$, $\texttt{MaxMessages} = 30$
- Устойчивость к цензуре: $\texttt{Values} = \{\text{"}A\text{"}\}$, $\texttt{MaxView} = 4$, $\texttt{MaxHeight} = 2$, $\texttt{MaxMessages} = 40$

Проверяемые свойства. Проверялись инварианты безопасности (*Agreement*, *Validity*), темпоральное свойство живучести (*Termination*) при допущениях справедливости, а также инвариант устойчивости к цензуре `HonestLeaderInFPlusOneBlocks` (см. раздел 4.3).

Результаты. Проверка TLC подтверждаёт, что *Agreement* и *Validity* выполняются (порядка $1{,}18 \cdot 10^9$ состояний за 4 ч), а *Termination* выполняется при заданных допущениях справедливости. Однако инвариант устойчивости к цензуре нарушается: TLC находит контрпример, в котором византийский процесс 1 фиксируется как `blockLeader` на высотах 1 и 2, уже после перебора порядка $5{,}5 \cdot 10^5$ состояний за 12 с. Причина нарушения заключается в том, что при стандартной ротации лидеров и отсутствии дополнительных ограничений лидера византийский валидатор может за счёт асимметричных задержек сообщений и управления моментом смены раунда добиваться последовательного лидерства на соседних высотах, тем самым демонстрируя, что одних лишь безопасности и живучести недостаточно для обеспечения устойчивости к цензуре.

6.4. QBFT с f-skip: все свойства выполняются

Для QBFT с f-skip применялись два набора параметров:

- Agreement/Validity: `Values` $= \{"A", "B"\}$, `MaxView` $= 3$, `MaxHeight` $= 2$, `MaxMessages` $= 30$
- Устойчивость к цензуре: `Values` $= \{"A"\}$, `MaxView` $= 5$, `MaxHeight` $= 2$ (и 3)*, `MaxMessages` $= 50$

Примечание. Для проверки инварианта `HonestLeaderInFPlusOneBlocks` достаточно, чтобы `MaxHeight` покрывал хотя бы одно окно длины $F + 1$; при $F = 1$ минимально достаточно `MaxHeight`$=2$.

Проверяемые свойства. Проверялись те же свойства, что и для QBFT без f-skip, с учётом модификации выбора лидера, обеспечивающей `HonestLeaderInFPlusOneBlocks`.

Результаты. Проверка TLC подтверждает выполнение всех свойств: *Agreement* и *Validity* выполняются (порядка $5{,}77 \cdot 10^8$ состояний за 3 ч); *Termination* сохраняется при тех же допущениях справедливости; инвариант устойчивости к цензуре выполняется (порядка $5{,}13 \cdot 10^8$ состояний за 6 ч). Механизм f-skip запрещает любому валидатору быть лидером более одного раза в любом окне из $F+1$ последовательно зафиксированных блоков. При $F = 1$ это означает, что в любых двух последовательных блоках по крайней мере один лидер обязан быть корректным, и, следовательно, "окно цензуры" ограничено сверху значением F последовательных блоков под византийским лидерством.

6.5. Сводка результатов верификации

В таблице 3 сведены результаты верификации для всех трёх вариантов протокола на основе фактических запусков TLC. В совокупности результаты иллюстрируют эволюцию IBFT: от небезопасной версии (IBFT 1.0) к версии, удовлетворяющей свойствам безопасности и живучести (QBFT), и далее к версии, обеспечивающей устойчивость к цензуре (QBFT с f-skip).

6.6. Ограничения и вопросы масштабируемости

Наш подход наследует типичные ограничения подхода, связанного с проверкой моделей.

Взрыв пространства состояний. Экспоненциальный рост числа состояний ограничивает анализ малыми конфигурациями ($N = 4$, $F = 1$). Тем не менее, именно минимальная конфигурация $N = 3F + 1$ соответствует "границе устойчивости" и воспроизводит наиболее сложные сценарии корректности, поэтому она является информативной для выявления уязвимостей и проверки исправлений.

Ограниченность по высотам и раундам. Верификация выполнялась для ограниченных значений высоты (1–3) и номера раунда (3–5). Это

Таблица 3: Сводка результатов верификации для вариантов IBFT

Свойство	**IBFT 1.0**	**QBFT (без f-skip)**	**QBFT (с f-skip)**
Agreement (безопасность)	Нарушено	Выполняется	Выполняется
Validity (безопасность)	Выполняется	Выполняется	Выполняется
Termination (живучесть)	Н/П	Выполняется*	Выполняется*
Устойчивость к цензуре	Н/П	Нарушено	Выполняется
Перебор состояний (TypeOK)	$\sim 4{,}6 \cdot 10^6$	$\sim 1{,}18 \cdot 10^9$	$\sim 5{,}77 \cdot 10^8$
Перебор состояний (Agreement)	$\sim 5{,}9 \cdot 10^5$	$\sim 1{,}18 \cdot 10^9$	$\sim 5{,}77 \cdot 10^8$
Перебор состояний (Censorship)	Н/П	$\sim 5{,}5 \cdot 10^5$	$\sim 5{,}13 \cdot 10^8$
Суммарное время верификации	61 с	4 ч	9 ч

*Живучесть проверялась при допущениях справедливости

не является доказательством для неограниченных исполнений, однако на практике такие границы превышают типичные шаблоны исполнения, необходимые для воспроизведения нарушений безопасности или атак цензуры.

Чувствительность к параметрам. Мы наблюдали значимую зависимость размеров пространства состояний от выбора параметров. Например, сокращение `Values` с `{"A "B"}` до `{"A"}` резко уменьшает пространство состояний при проверке устойчивости к цензуре, не влияя на корректность проверяемого инварианта.

Параметризованная верификация. Для получения гарантий при произвольных N и F в перспективе целесообразно использовать параметризованную проверку моделей или доказательство теорем. Тем не менее, для целей данной работы (демонстрация дефектов, исправлений и их эффектов) ограниченная проверка моделей особенно ценна тем, что выдаёт конкретные, интерпретируемые контрпримеры.

Несмотря на указанные ограничения, наша верификация:

1. воспроизвела нарушение безопасности в IBFT 1.0;
2. подтвердила выполнение свойств безопасности и живучести в QBFT;

3. выявила уязвимость устойчивости к цензуре при стандартной ротации лидера;
4. формально подтвердила, что модификация f-skip обеспечивает устойчивость к цензуре, не нарушая остальные свойства.

7. Заключение и направления дальнейших исследований

В данной работе представлен последовательный путь формальной верификации протокола консенсуса IBFT, который устраняет разрыв между анализом исторически уязвимых версий и современными исправленными реализациями, а также расширяет класс проверяемых гарантий за пределы традиционных безопасности и живучести. Мы внесли три ключевых вклада. Во-первых, мы дали целостный формальный анализ эволюции IBFT, подтвердив корректность промежуточных исправлений и построив воспроизводимые контрпримеры для ошибочных версий. В частности, спецификации TLA+ и эксперименты TLC демонстрируют, что отсутствие передачи состояния при смене раунда в IBFT 1.0 приводит к нарушению согласованности принятых значений, тогда как исправленные варианты (QBFT) сохраняют безопасность за счёт механизма `HighestPrepared`. Во-вторых, мы сформулировали строгое определение устойчивости к цензуре для BFT-протоколов с лидером и показали, что стандартный IBFT (и более общий класс протоколов с предсказуемой ротацией лидера) не удовлетворяет этому свойству: византийский лидер может исключать транзакции, сохраняя формальные свойства безопасности и живучести. В-третьих, мы предложили и формально верифицировали модификацию *f-skip* — лёгкий механизм, применимый к BFT-протоколам с лидером, который обеспечивает устойчивость к цензуре и при этом не нарушает исходные гарантии безопасности и живучести. Верификация подтверждает, что f-skip ограничивает “окно цензуры” сверху величиной F последовательных блоков под византийским лидерством.

Практическая значимость результатов заключается в следующем. Механизм f-skip даёт простое и обратно совместимое усиление для существующих реализаций IBFT: требуется лишь модификация выбора лидера, при этом достигается формально подтверждённая устойчивость к цензуре — критически важная гарантия для корпоративных блокчейнов, где избирательное исключение транзакций напрямую трансформируется в операционные и бизнес-риски. Более общий вывод состоит в том, что одних только безопасности и живучести недостаточно для практического консенсуса: свойства “справедливости” и противодействия цензуре долж-

ны быть явно заложены в протокол и формально проверены, особенно в архитектуре с выбранным лидером. Наконец, мы демонстрируем прикладную ценность ограниченной проверки моделей: даже минимальные конфигурации ($N = 4$, $F = 1$) способны воспроизводить существенные сценарии и порождать контрпримеры, полезные для проектирования и отладки протоколов.

В качестве направлений дальнейших исследований представляются наиболее перспективными следующие задачи. (i) Переход от ограниченной проверки к параметризованной верификации, обеспечивающей гарантии при произвольных N и F, за счёт параметризованного model checking и/или доказательства теорем. (ii) Адаптация f-skip к динамическим наборам валидаторов (вход/выход валидаторов), что критично для практических систем с изменяющимся составом участников. (iii) Перенос формализации устойчивости к цензуре и механизма f-skip на другие протоколы консенсуса с лидером (например, HotStuff, Tendermint, LibraBFT) для усиления гарантий в более широком ландшафте BFT-консенсуса. (iv) Интеграция f-skip в промышленные клиенты (например, GoQuorum, Hyperledger Besu) с последующей оценкой влияния на производительность и задержки. (v) Расширение модели противника в сторону рациональных, экономически мотивированных атак (в частности, связанных с MEV), чтобы формальная модель лучше отражала реальные стимулы цензуры и манипуляций порядком включения транзакций.

Эволюция IBFT от уязвимой версии к формально проверенному и устойчивому к цензуре варианту подчёркивает практическую роль формальных методов в проектировании блокчейн-консенсуса. Наши результаты показывают, что формальная верификация является не только академическим упражнением, но и необходимым инженерным инструментом для обеспечения надёжности распределённых систем, на которых строятся современные блокчейн-приложения. В совокупности мы связываем анализ эволюции протокола, формализацию нового свойства и практический механизм его обеспечения, тем самым внося вклад как в понимание IBFT, так и в более широкую область формальной верификации византийского консенсуса.

Доступность данных

Все спецификации TLA+, конфигурации TLC и скрипты верификации доступны по адресу:
`https://github.com/BondarevNS/ibft-tla-plus-verification`

Список литературы

[1] Vostrikov D., Madhwal Y., Seoev A., Smirnova A., Yanovich Yu., Smirnov A., Gorgadze V., "Unpacking Maximum Extractable Value on Polygon: A Study on Atomic Arbitrage", *arXiv preprint*, 2025.

[2] Wahrstätter A., Ernstberger J., Yaish A., Zhou L., Qin K., Tsuchiya T., Steinhorst S., Svetinovic D., Christin N., Barczentewicz M., Gervais A., "Blockchain Censorship", *Proceedings of the ACM Web Conference 2024*, 2024, 1632–1643. DOI: 10.1145/3589334.3645431.

[3] Daian P., Goldfeder S., Kell T., Li Y., Zhao X., Bentov I., Breidenbach L., Juels A., "Flash Boys 2.0: Frontrunning in Decentralized Exchanges, Miner Extractable Value, and Consensus Instability", *2020 IEEE Symposium on Security and Privacy (SP)*, 2020, 910–927. DOI: 10.1109/SP40000.2020.00040.

[4] Heimbach L., Kiffer L., Ferreira Torres C., Wattenhofer R., "Ethereum's Proposer-Builder Separation: Promises and Realities", *Proceedings of the 2023 ACM on Internet Measurement Conference*, 2023, 406–420. DOI: 10.1145/3618257.3624824.

[5] Afanasyeva A., Kameskiy D., Telnov S., Yanovich Yu., "Leaderless Byzantine Fault-Tolerant Consensus Protocol for Blockchains", *Proceedings of the 2023 6th International Conference on Blockchain Technology and Applications*, 2023, 78–84. DOI: 10.1145/3651655.3651665.

[6] Seoev A., Gremyachikh L., Smirnova A., Madhwal Y., Kalacheva A., Belousov D., Zubov I., Smirnov A., Fedyanin D., Gorgadze V., Yanovich Yu., "The Bidding Games: Reinforcement Learning for MEV Extraction on Polygon Blockchain", *arXiv preprint*, 2025.

[7] Öz B., Sui D., Thiery T., Matthes F., "Who Wins Ethereum Block Building Auctions and Why?", *6th Conference on Advances in Financial Technologies*, 2024, 22:1–22:25. DOI: 10.4230/LIPIcs.AFT.2024.22.

[8] Rashid M., Rasool I., Zafar N.A., Afzaal H., "Formal Modeling and Verification of Validator Voluntarily Exit in Ethereum 2.0 Beacon Chain", *2023 2nd International Conference on Emerging Trends in Electrical, Control, and Telecommunication Engineering (ETECTE)*, 2023, 1–6. DOI: 10.1109/ETECTE59617.2023.10396687.

[9] Rashid M., Rasool I., Zafar N.A., Afzaal H., "Formal Modeling and Verification of Justification and Finalization of Checkpoints in Ethereum 2.0 Beacon Chain", *2024 IEEE 1st Karachi Section Humanitarian Technology Conference (KHI-HTC)*, 2024, 1–6. DOI: 10.1109/KHI-HTC60760.2024.10481930.

[10] Cassez F., Fuller J., Asgaonkar A., "Formal Verification of the Ethereum 2.0 Beacon Chain", *Lecture Notes in Computer Science*, **13243** (2022), 167–182. DOI: 10.1007/978-3-030-99524-9_9.

[11] Afzaal H., Zafar N.A., Tehseen A., Kousar S., Imran M., "Formal Verification of Justification and Finalization in Beacon Chain", *IEEE Access*, **12** (2024), 55077–55102. DOI: 10.1109/ACCESS.2024.3389551.

[12] Li E., Serbanuta T., Diaconescu D., Zamfir V., Rosu G., "Formalizing Correct-by-Construction Casper in Coq", *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 2020, 1–3. DOI: 10.1109/ICBC48266.2020.9169468.

[13] Pass R., Shi E., “Hybrid consensus: Efficient consensus in the permissionless model”, *Leibniz International Proceedings in Informatics (LIPIcs), DISC 2017*, **91. DOI:** 10.4230/LIPIcs.DISC.2017.39 (2017).

[14] Lamport L., *Specifying Systems*, Addison–Wesley, 2002. DOI: 10.5555/579617.

[15] Nakamoto S., “Bitcoin: A peer-to-peer electronic cash system”, *White paper*, 2008.

[16] Lamport L., “TLA+ specification of Paxos”, *Online artifact*, 2008.

[17] Losa G., “TLA+ specification of TetraBFT”, *Online artifact*, 2024.

[18] Saltini R., “QBFT Formal Specification and Verification in Dafny”, *Online artifact*, 2024.

[19] Saltini R., Hyland-Wood D., “IBFT 2.0: A Safe and Live Variation of the IBFT Blockchain Consensus Protocol for Eventually Synchronous Networks”, *arXiv preprint arXiv:1909.10194*, 2019. DOI: 10.48550/arXiv:1909.10194.

[20] França B., Kolegov D., Konnov I., Prusak G., “ChonkyBFT: Consensus Protocol of ZKsync”, *arXiv preprint arXiv:2503.15380*, 2025.

[21] Moniz H., “The Istanbul BFT Consensus Algorithm”, *arXiv preprint arXiv:2002.03613*, 2020. DOI: 10.48550/arXiv.2002.03613.

[22] Buterin V., Griffith V., “Casper the Friendly Finality Gadget”, *CoRR, abs/1710.09437*, 2017.

[23] Buterin V., Hernandez D., Kamphefner T., Pham K., Qiao Z., Ryan D., Sin J., Wang Y., Zhang Y.X., “Combining GHOST and Casper”, *CoRR, abs/2003.03052*, 2020.

[24] Castro M., Liskov B., “Practical Byzantine fault tolerance”, *Proceedings of the Third Symposium on Operating Systems Design and Implementation (OSDI '99)*, 1999, 173–186. DOI: 10.1145/296806.296824.

[25] Ethereum Community, “Istanbul Byzantine Fault Tolerance (EIP-650)”, *Online proposal*.

[26] ConsenSys, “Quorum Blockchain”, *Online project*.

[27] Saltini R., “Correctness Analysis of IBFT”, *CoRR, abs/1901.07160*, 2019.

[28] Ongaro D., “Consensus: Bridging theory and practice”, *Ph.D. thesis, Stanford University*, 2014.

[29] Rahli V., Vukotic I., Völp M., Esteves-Verissimo P., “Velisarios: Byzantine Fault-Tolerant Protocols Powered by Coq”, *Programming Languages and Systems (ESOP 2018), Lecture Notes in Computer Science*, **10801** (2018), 619–650. DOI: 10.1007/978-3-319-89884-1_22.

[30] Alturki M.A., Chen J., Luchangco V., Moore B., Palmskog K., Peña L., Rosu G., “Towards a verified model of the Algorand consensus protocol in Coq”, *Formal Methods. FM 2019 International Workshops, Lecture Notes in Computer Science*, **12232** (2020), 362–367. DOI: 10.1007/978-3-030-54994-7_27.

[31] Konnov I., Kukovec J., Tran T.-H., “TLA+ model checking made symbolic”, *Proc. ACM Program. Lang.*, **3**:OOPSLA (2019), 123. DOI: 10.1145/3360549.

[32] Konnov I.V., Kuppe M.A., Merz S., “Specification and Verification with the TLA+ Trifecta: TLC, Apalache, and TLAPS”, *Leveraging Applications of Formal Methods*, 2022. DOI: 10.1007/978-3-031-19849-6_6.

[33] Braithwaite S., Buchman E., Konnov I., Milosevic Z., Stoilkovska I., Widder J., Zamfir A., “Formal Specification and Model Checking of the Tendermint Blockchain Synchronization Protocol”, *2nd Workshop on Formal Methods for*

Blockchains (FMBC 2020), OASIcs, **84** (2020), 10:1–10:8. DOI: 10.4230/OASIcs.FMBC.2020.10.

[34] Kukharenko V., Ziborov K., Sadykov R., Rezin R., "Verification of HotStuff BFT Consensus Protocol With TLA+/TLC in an Industrial Setting", *SHS Web of Conferences*, **93** (2021), 01006. DOI: 10.1051/shsconf/20219301006.

[35] Carr H., Jenkins C., Moir M., Miraldo V.C., Silva L., "Towards Formal Verification of HotStuff-Based Byzantine Fault Tolerant Consensus in Agda", *NASA Formal Methods (NFM 2022), Proceedings*, 2022, 616–635. DOI: 10.1007/978-3-031-06773-0_33.

[36] Garay J., Kiayias A., Leonardos N., "The Bitcoin Backbone Protocol: Analysis and Applications", *Advances in Cryptology – EUROCRYPT 2015, Lecture Notes in Computer Science*, **9057** (2015), 281–310.

[37] Zhang S., Lee J., "Analysis of the main consensus protocols of blockchain", *ICT Express*, 2019, 1–7. DOI: 10.1016/j.icte.2019.08.001.

[38] Crain T., Gramoli V., Larrea M., Raynal M., "DBFT: Efficient Leaderless Byzantine Consensus and its Application to Blockchains", *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*, 2018, 1–8.

Статья поступила 21 января 2026 г.

Formal Verification and Censorship Resistance of Byzantine Consensus Protocols: An IBFT Case Study

K. V. Ziborov, N. S. Bondarev, Y. A. Yanovich

We present a formal study of the Istanbul BFT (IBFT) protocol family that connects its vulnerable early design, corrected intermediate variants, and a censorship-resistant modification within a unified verification framework. Using TLA+ specifications and TLC model checking, we (i) obtain counterexample traces demonstrating safety violations in the first IBFT version, (ii) verify safety and liveness for a corrected QBFT variant, and (iii) formalize censorship resistance for leader-based Byzantine consensus as a bounded chain-quality requirement and show that standard leader rotation can violate it. We then introduce and verify the *f-skip* leader-selection rule, which enforces censorship resistance while preserving the original safety and liveness guarantees.

Keywords: IBFT, QBFT, Byzantine consensus, formal verification, TLA+, TLC, censorship resistance..

References

[1] Vostrikov D., Madhwal Y., Seoev A., Smirnova A., Yanovich Yu., Smirnov A., Gorgadze V., "Unpacking Maximum Extractable Value on Polygon: A Study on Atomic Arbitrage", *arXiv preprint*, 2025.

[2] Wahrstätter A., Ernstberger J., Yaish A., Zhou L., Qin K., Tsuchiya T., Steinhorst S., Svetinovic D., Christin N., Barczentewicz M., Gervais A., "Blockchain Censorship", *Proceedings of the ACM Web Conference 2024*, 2024, 1632–1643. DOI: 10.1145/3589334.3645431.

[3] Daian P., Goldfeder S., Kell T., Li Y., Zhao X., Bentov I., Breidenbach L., Juels A., "Flash Boys 2.0: Frontrunning in Decentralized Exchanges, Miner Extractable Value, and Consensus Instability", *2020 IEEE Symposium on Security and Privacy (SP)*, 2020, 910–927. DOI: 10.1109/SP40000.2020.00040.

[4] Heimbach L., Kiffer L., Ferreira Torres C., Wattenhofer R., "Ethereum's Proposer-Builder Separation: Promises and Realities", *Proceedings of the 2023 ACM on Internet Measurement Conference*, 2023, 406–420. DOI: 10.1145/3618257.3624824.

[5] Afanasyeva A., Kameskiy D., Telnov S., Yanovich Yu., "Leaderless Byzantine Fault-Tolerant Consensus Protocol for Blockchains", *Proceedings of the 2023 6th International Conference on Blockchain Technology and Applications*, 2023, 78–84. DOI: 10.1145/3651655.3651665.

[6] Seoev A., Gremyachikh L., Smirnova A., Madhwal Y., Kalacheva A., Belousov D., Zubov I., Smirnov A., Fedyanin D., Gorgadze V., Yanovich Yu., "The Bidding Games: Reinforcement Learning for MEV Extraction on Polygon Blockchain", *arXiv preprint*, 2025.

[7] Öz B., Sui D., Thiery T., Matthes F., "Who Wins Ethereum Block Building Auctions and Why?", *6th Conference on Advances in Financial Technologies*, 2024, 22:1–22:25. DOI: 10.4230/LIPIcs.AFT.2024.22.

[8] Rashid M., Rasool I., Zafar N.A., Afzaal H., "Formal Modeling and Verification of Validator Voluntarily Exit in Ethereum 2.0 Beacon Chain", *2023 2nd International Conference on Emerging Trends in Electrical, Control, and Telecommunication Engineering (ETECTE)*, 2023, 1–6. DOI: 10.1109/ETECTE59617.2023.10396687.

[9] Rashid M., Rasool I., Zafar N.A., Afzaal H., "Formal Modeling and Verification of Justification and Finalization of Checkpoints in Ethereum 2.0 Beacon Chain", *2024 IEEE 1st Karachi Section Humanitarian Technology Conference (KHI-HTC)*, 2024, 1–6. DOI: 10.1109/KHI-HTC60760.2024.10481930.

[10] Cassez F., Fuller J., Asgaonkar A., "Formal Verification of the Ethereum 2.0 Beacon Chain", *Lecture Notes in Computer Science*, **13243** (2022), 167–182. DOI: 10.1007/978-3-030-99524-9_9.

[11] Afzaal H., Zafar N.A., Tehseen A., Kousar S., Imran M., "Formal Verification of Justification and Finalization in Beacon Chain", *IEEE Access*, **12** (2024), 55077–55102. DOI: 10.1109/ACCESS.2024.3389551.

[12] Li E., Serbanuta T., Diaconescu D., Zamfir V., Rosu G., "Formalizing Correct-by-Construction Casper in Coq", *2020 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 2020, 1–3. DOI: 10.1109/ICBC48266.2020.9169468.

[13] Pass R., Shi E., “Hybrid consensus: Efficient consensus in the permissionless model”, *Leibniz International Proceedings in Informatics (LIPIcs), DISC 2017*, **91. DOI:** 10.4230/LIPIcs.DISC.2017.39 (2017).

[14] Lamport L., *Specifying Systems*, Addison–Wesley, 2002. DOI: 10.5555/579617.

[15] Nakamoto S., “Bitcoin: A peer-to-peer electronic cash system”, *White paper*, 2008.

[16] Lamport L., “TLA+ specification of Paxos”, *Online artifact*, 2008.

[17] Losa G., “TLA+ specification of TetraBFT”, *Online artifact*, 2024.

[18] Saltini R., “QBFT Formal Specification and Verification in Dafny”, *Online artifact*, 2024.

[19] Saltini R., Hyland-Wood D., “IBFT 2.0: A Safe and Live Variation of the IBFT Blockchain Consensus Protocol for Eventually Synchronous Networks”, *arXiv preprint arXiv:1909.10194*, 2019. DOI: 10.48550/arXiv:1909.10194.

[20] França B., Kolegov D., Konnov I., Prusak G., “ChonkyBFT: Consensus Protocol of ZKsync”, *arXiv preprint arXiv:2503.15380*, 2025.

[21] Moniz H., “The Istanbul BFT Consensus Algorithm”, *arXiv preprint arXiv:2002.03613*, 2020. DOI: 10.48550/arXiv.2002.03613.

[22] Buterin V., Griffith V., “Casper the Friendly Finality Gadget”, *CoRR, abs/1710.09437*, 2017.

[23] Buterin V., Hernandez D., Kamphefner T., Pham K., Qiao Z., Ryan D., Sin J., Wang Y., Zhang Y.X., “Combining GHOST and Casper”, *CoRR, abs/2003.03052*, 2020.

[24] Castro M., Liskov B., “Practical Byzantine fault tolerance”, *Proceedings of the Third Symposium on Operating Systems Design and Implementation (OSDI ’99)*, 1999, 173–186. DOI: 10.1145/296806.296824.

[25] Ethereum Community, “Istanbul Byzantine Fault Tolerance (EIP-650)”, *Online proposal*.

[26] ConsenSys, “Quorum Blockchain”, *Online project*.

[27] Saltini R., “Correctness Analysis of IBFT”, *CoRR, abs/1901.07160*, 2019.

[28] Ongaro D., “Consensus: Bridging theory and practice”, *Ph.D. thesis, Stanford University*, 2014.

[29] Rahli V., Vukotic I., Völp M., Esteves-Verissimo P., “Velisarios: Byzantine Fault-Tolerant Protocols Powered by Coq”, *Programming Languages and Systems (ESOP 2018), Lecture Notes in Computer Science*, **10801** (2018), 619–650. DOI: 10.1007/978-3-319-89884-1_22.

[30] Alturki M.A., Chen J., Luchangco V., Moore B., Palmskog K., Peña L., Rosu G., “Towards a verified model of the Algorand consensus protocol in Coq”, *Formal Methods. FM 2019 International Workshops, Lecture Notes in Computer Science*, **12232** (2020), 362–367. DOI: 10.1007/978-3-030-54994-7_27.

[31] Konnov I., Kukovec J., Tran T.-H., “TLA+ model checking made symbolic”, *Proc. ACM Program. Lang.*, **3**:OOPSLA (2019), 123. DOI: 10.1145/3360549.

[32] Konnov I.V., Kuppe M.A., Merz S., “Specification and Verification with the TLA+ Trifecta: TLC, Apalache, and TLAPS”, *Leveraging Applications of Formal Methods*, 2022. DOI: 10.1007/978-3-031-19849-6_6.

[33] Braithwaite S., Buchman E., Konnov I., Milosevic Z., Stoilkovska I., Widder J., Zamfir A., “Formal Specification and Model Checking of the Tendermint Blockchain Synchronization Protocol”, *2nd Workshop on Formal Methods for Blockchains (FMBC 2020), OASIcs*, **84** (2020), 10:1–10:8. DOI: 10.4230/OASIcs.FMBC.2020.10.

[34] Kukharenko V., Ziborov K., Sadykov R., Rezin R., “Verification of HotStuff BFT Consensus Protocol With TLA+/TLC in an Industrial Setting”, *SHS Web of Conferences*, **93** (2021), 01006. DOI: 10.1051/shsconf/20219301006.

[35] Carr H., Jenkins C., Moir M., Miraldo V.C., Silva L., “Towards Formal Verification of HotStuff-Based Byzantine Fault Tolerant Consensus in Agda”, *NASA Formal Methods (NFM 2022), Proceedings*, 2022, 616–635. DOI: 10.1007/978-3-031-06773-0_33.

[36] Garay J., Kiayias A., Leonardos N., “The Bitcoin Backbone Protocol: Analysis and Applications”, *Advances in Cryptology – EUROCRYPT 2015, Lecture Notes in Computer Science*, **9057** (2015), 281–310.

[37] Zhang S., Lee J., “Analysis of the main consensus protocols of blockchain”, *ICT Express*, 2019, 1–7. DOI: 10.1016/j.icte.2019.08.001.

[38] Crain T., Gramoli V., Larrea M., Raynal M., “DBFT: Efficient Leaderless Byzantine Consensus and its Application to Blockchains”, *2018 IEEE 17th International Symposium on Network Computing and Applications (NCA)*, 2018, 1–8.

Received on January 21, 2026

Улучшение качества векторных представлений слов за счёт использования нескольких источников представлений

А. М. Колосов* , А. И. Майсурадзе‡

Векторные представления слов активно используются в задачах машинного перевода, рекомендательных системах и информационном поиске. Качество таких представлений, оцениваемое как ранговая корреляция с экспертными оценками семантической близости, остаётся ограниченным. В данной работе предлагается подход к повышению качества векторных представлений слов путём слияния нескольких независимых источников первичных представлений. Вводятся понятия монотонных и антимонотонных четвёрок слов, формулируется и проверяется гипотеза о том, что информация, содержащаяся в монотонных четвёрках, позволяет восстановить истинный порядок близостей для антимонотонных четвёрок. Предложены метод отбора четвёрок слов, двухшаговая процедура коррекции с использованием полносвязного слоя и функции потерь на четвёрках (quadruplet loss), а также способ оценки качества полученных представлений. Экспериментальные результаты на моделях Word2Vec и GloVe, обученных на лемматизированной Википедии, демонстрируют возможность повышения качества представлений при оценке на экспертных наборах данных MEN, SimLex-999 и WordSim-353.

Ключевые слова: векторные представления слов, семантическая близость, слияние данных, quadruplet loss, многомерное шкалирование, Word2Vec, GloVe.

Колосов Алексей Михайлович — научный сотрудник кафедры математической теории интеллектуальных систем механико-математического факультета Московского государственного университета имени М. В. Ломоносова, e-mail: aleksei.kolosov@math.msu.ru, ORCID: 0000-0002-9474-9666.

Kolosov Alexey Mikhajlovich — Research Fellow, Department of Mathematical Theory of Intelligent Systems, Faculty of Mechanics and Mathematics, Lomonosov Moscow State University.

‡*Майсурадзе Арчил Ивериевич* — кандидат физико-математических наук, доцент кафедры математических методов прогнозирования факультета вычислительной математики и кибернетики Московского государственного университета имени М. В. Ломоносова, e-mail: maysuradze@cs.msu.ru, ORCID: 0000-0002-1757-556X.

Maysuradze Archil Iverievich — Candidate of Physical and Mathematical Sciences, Associate Professor, Department of Mathematical Methods of Forecasting, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University.

1. Введение

Задача построения векторных представлений слов (word embeddings) является одной из центральных в области обработки естественного языка (NLP). Если два слова, близких по смыслу, могут быть представлены близкими векторами, то такие представления эффективно применяются в широком классе задач: машинном переводе, информационном поиске, категоризации и суммаризации текстов, анализе тональности, разрешении лексической многозначности, генерации ответов в диалоговых системах [1].

За прошедшее десятилетие появились и получили развитие несколько семейств методов построения векторных представлений слов, позволяющих получать представления всё более высокого качества. Вначале появилось семейство вычислительно эффективных методов Skip-gram и CBOW [1], затем — FastText [2], позволяющий получать представления слов при расширении словаря за счёт использования подсловных единиц, и далее — BERT [3], дающий контекстно-зависимые представления для последовательностей слов. В результате трудоёмкость построения векторных представлений слов постоянно возрастает, однако их качество, оцениваемое как ранговая корреляция с экспертными оценками семантической близости между словами [4], не превышает 0,8 [5].

Разработка и обучение принципиально новой модели, которая бы учитывала дополнительную экспертную информацию о семантической близости между словами, сегодня оценивается в тысячи часов вычислительных экспериментов [2]. В связи с этим актуальным становится альтернативный подход: вместо создания новой модели — коррекция результатов работы уже существующих моделей с учётом дополнительной обучающей информации.

В настоящей работе рассматривается подход, при котором проводится слияние нескольких исходных (первичных) наборов векторных представлений слов в одни вторичные представления. Такой подход, с одной стороны, существенно снижает вычислительные затраты по сравнению с разработкой нового метода построения первичных представлений, а с другой — позволяет получить представления слов более высокого качества. Научная ценность работы состоит как в предложенном подходе к получению вторичных представлений из нескольких наборов первичных, так и в самих полученных вторичных представлениях, поскольку размер словаря имеет тот же порядок по количеству слов, что и естественный язык.

Ранее авторами был предложен метод коррекции одного набора векторных представлений с использованием экспертных оценок семантиче-

ской близости [7]. В данной работе этот подход обобщается на случай использования нескольких независимых источников представлений без привлечения внешней экспертной информации на этапе коррекции.

2. Постановка задачи

Пусть имеется словарь $W = \{w_1, w_2, \ldots, w_N\}$, состоящий из N слов. Пусть также имеются два набора первичных векторных представлений слов, полученных различными методами:

$$m_1 \colon W \to \mathbb{R}^d, \qquad m_2 \colon W \to \mathbb{R}^d, \tag{1}$$

где d — размерность пространства представлений.

Задача состоит в построении нового отображения $m^* \colon W \to \mathbb{R}^d$, такого, что качество представлений m^*, измеренное как ранговая корреляция (коэффициент корреляции Спирмена) с экспертными наборами оценок семантической близости, превышает качество каждого из исходных наборов m_1 и m_2.

В качестве меры близости между словами a и b в пространстве представлений используется косинусная мера:

$$\operatorname{sim}(a, b) = \frac{m(a) \cdot m(b)}{\|m(a)\| \cdot \|m(b)\|}. \tag{2}$$

Для оценки качества представлений используются стандартные экспертные наборы данных: MEN [8], SimLex-999 [9] и WordSim-353 [4], содержащие пары слов с экспертными оценками их семантической близости.

3. Монотонные и антимонотонные четвёрки

Центральным понятием предлагаемого подхода является разделение четвёрок слов на монотонные и антимонотонные.

Монотонная четвёрка. Четвёрка слов (a, b, c, d) называется *монотонной* относительно представлений m_1 и m_2, если порядок близостей в обоих наборах представлений совпадает:

$$\operatorname{sim}_{m_1}(a, b) < \operatorname{sim}_{m_1}(c, d) \quad \text{и} \quad \operatorname{sim}_{m_2}(a, b) < \operatorname{sim}_{m_2}(c, d). \tag{3}$$

Антимонотонная четвёрка. Четвёрка слов (a, b, c, d) называется *антимонотонной* относительно представлений m_1 и m_2, если порядок близостей различается:

$$\operatorname{sim}_{m_1}(a, b) < \operatorname{sim}_{m_1}(c, d) \quad \text{и} \quad \operatorname{sim}_{m_2}(a, b) > \operatorname{sim}_{m_2}(c, d). \tag{4}$$

Интуитивно, если два независимых метода согласуются в оценке порядка близостей для некоторой четвёрки слов, то этот порядок с большей вероятностью является истинным. Для антимонотонных четвёрок истинный порядок неизвестен, и задача состоит в его восстановлении.

Гипотеза. В монотонных четвёрках содержатся сведения, достаточные для коррекции порядка близостей в антимонотонных четвёрках.

Данная гипотеза основывается на следующем наблюдении: если построить новые представления слов, оптимизируя функцию потерь на монотонных четвёрках, то для значительной доли антимонотонных четвёрок порядок близостей будет скорректирован — то есть станет совпадать с порядком, определяемым лучшим из двух исходных наборов представлений.

4. Метод коррекции векторных представлений

4.1. Функция потерь на четвёрках

Предлагаемый подход использует функцию потерь, определённую на четвёрках слов (quadruplet loss). Пусть (a, b, c, d) — монотонная четвёрка, для которой выполнено $\mathrm{sim}(a, b) < \mathrm{sim}(c, d)$ в обоих наборах представлений. Тогда функция потерь для данной четвёрки определяется как:

$$\mathcal{L}(a, b, c, d) = \max\bigl(0,\ \mathrm{sim}_{m^*}(a, b) - \mathrm{sim}_{m^*}(c, d) + \alpha\bigr), \tag{5}$$

где $\alpha > 0$ — параметр зазора (margin), а m^* — обучаемое отображение. Общий функционал качества определяется как среднее потерь по всем отобранным монотонным четвёркам:

$$\mathcal{L}_{\text{total}} = \frac{1}{|\mathcal{Q}|} \sum_{(a,b,c,d)\in\mathcal{Q}} \mathcal{L}(a, b, c, d), \tag{6}$$

где $\mathcal{Q}$ — множество монотонных четвёрок.

4.2. Архитектура модели

Преобразование первичных представлений во вторичные осуществляется с помощью одного полносвязного слоя:

$$m^*(w) = \sigma(W_{\text{fc}} \cdot [m_1(w) \oplus m_2(w)] + b_{\text{fc}}), \tag{7}$$

где $[m_1(w) \oplus m_2(w)]$ — конкатенация первичных представлений слова w, $W_{\text{fc}} \in \mathbb{R}^{d\times 2d}$ и $b_{\text{fc}} \in \mathbb{R}^d$ — обучаемые параметры, σ — функция активации.

4.3. Двухшаговая процедура коррекции

Процедура коррекции состоит из двух последовательных шагов.

Шаг 1. Определение распределения коррекций. На первом шаге для каждой антимонотонной четвёрки определяется вероятность её коррекции. Для этого:

1. Из двух наборов первичных представлений формируются множества монотонных и антимонотонных четвёрок.

2. Многократно (100 раз) инициализируются случайные представления, которые затем обучаются на монотонных четвёрках с помощью полносвязного слоя и quadruplet loss.

3. Для каждой антимонотонной четвёрки подсчитывается доля запусков, в которых произошла коррекция — то есть порядок близостей в полученных представлениях стал совпадать с порядком в лучшем из исходных наборов.

Шаг 2. Коррекция представлений. На втором шаге выполняется собственно коррекция:

1. Устанавливается порог θ (например, $\theta = 55\%$): антимонотонная четвёрка считается монотонной, если доля коррекций на первом шаге превысила θ.

2. Формируется расширенное множество монотонных четвёрок, включающее как исходные монотонные, так и скорректированные антимонотонные четвёрки.

3. В качестве начального приближения берётся лучший из исходных наборов представлений, и производится обучение полносвязного слоя на расширенном множестве монотонных четвёрок.

Схематически первый шаг процедуры представлен на рис. 1, а второй шаг — на рис. 2.

Общая схема процедуры также представлена на рис. 3.

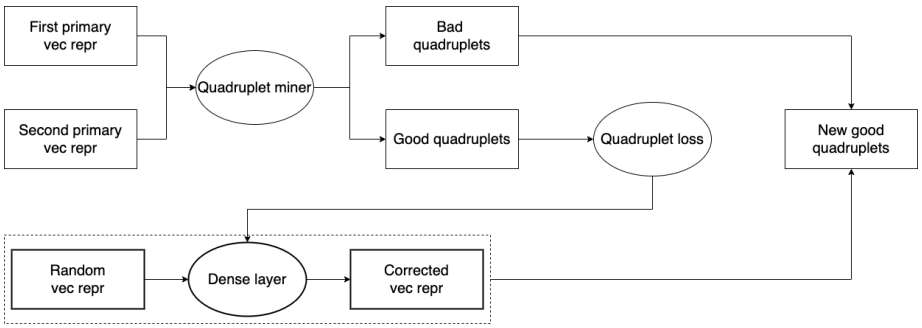

Рис. 1: Схема первого шага процедуры коррекции: определение распределения коррекций для антимонотонных четвёрок.

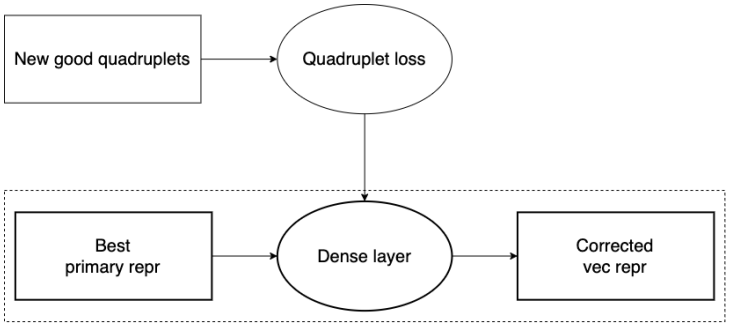

Рис. 2: Схема второго шага процедуры коррекции: обучение на расширенном множестве монотонных четвёрок.

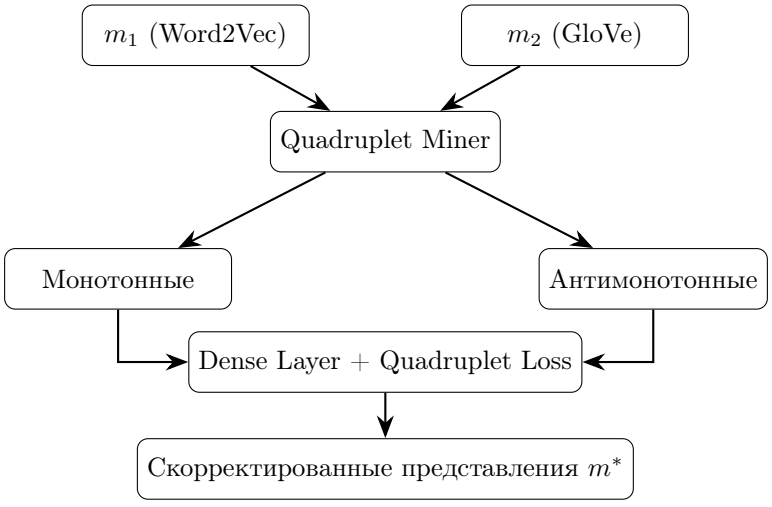

Рис. 3: Общая схема двухшаговой процедуры коррекции векторных представлений слов.

5. Связь с многомерным шкалированием

Предложенный подход тесно связан с задачами многомерного шкалирования (multidimensional scaling, MDS) [10]. В классической постановке MDS задача состоит в построении отображения $g\colon \mathrm{Obj} \to \mathbb{R}^d$, сохраняющего структуру попарных различий между объектами.

Пусть $d(i,j)$ — исходные различия между объектами i и j, а $e(x_1,x_2)$ — евклидова метрика в пространстве представлений. Отображение, сохраняющее ранги различий, называется *изотоническим*:

$$d(i,j) < d(k,l) \;\Rightarrow\; e\big(g(i),g(j)\big) < e\big(g(k),g(l)\big). \tag{8}$$

Ранг различия — это его порядковый номер в матрице попарных различий после сортировки по возрастанию. Предлагаемый метод коррекции можно рассматривать как частный случай неметрического MDS, в котором:

- в качестве исходных различий используются порядки, согласованные между несколькими источниками представлений (монотонные четвёрки);
- функция потерь (quadruplet loss) обеспечивает сохранение рангов;
- пространство представлений задаётся полносвязным слоем нейронной сети.

Таким образом, семантическая близость выступает как источник внешней информации, позволяющей уточнить структуру пространства представлений.

6. Экспериментальная оценка

6.1. Данные и настройка эксперимента

В качестве первичных представлений использовались предобученные модели Word2Vec (Skip-gram) и GloVe, построенные на лемматизированной русскоязычной Википедии и доступные в репозитории NLPL [11]. Характеристики моделей приведены в табл. 1.

Таблица 1: Характеристики первичных моделей векторных представлений.

Модель	**Размер словаря**	**Размерность**
Word2Vec (лемм.)	273 992	100
GloVe (лемм.)	273 930	100

По общему словарю из 210 186 слов были сформированы 100 000 монотонных и 100 000 антимонотонных четвёрок на одном и том же наборе слов. На первом шаге процедуры коррекции для каждой антимонотонной четвёрки было выполнено 100 запусков обучения с различной случайной инициализацией.

Для оценки качества представлений использовались экспертные наборы данных, представленные в средстве GluonNLP [6]: MEN (3 000 пар), SimLex-999 (998 пар) и WordSim-353 (351 пара). Пересечение словарей моделей с экспертными наборами составило 1 813, 586 и 190 пар соответственно.

6.2. Распределение коррекций

На первом шаге процедуры для каждой из 100 000 антимонотонных четвёрок было определено число коррекций из 100 запусков. Полученное распределение (рис. 4) имеет колоколообразную форму с центром вблизи 50 коррекций из 100 запусков. Это означает, что для значительной части антимонотонных четвёрок оба порядка близостей являются примерно равновероятными, однако существуют четвёрки, для которых один из порядков устойчиво воспроизводится.

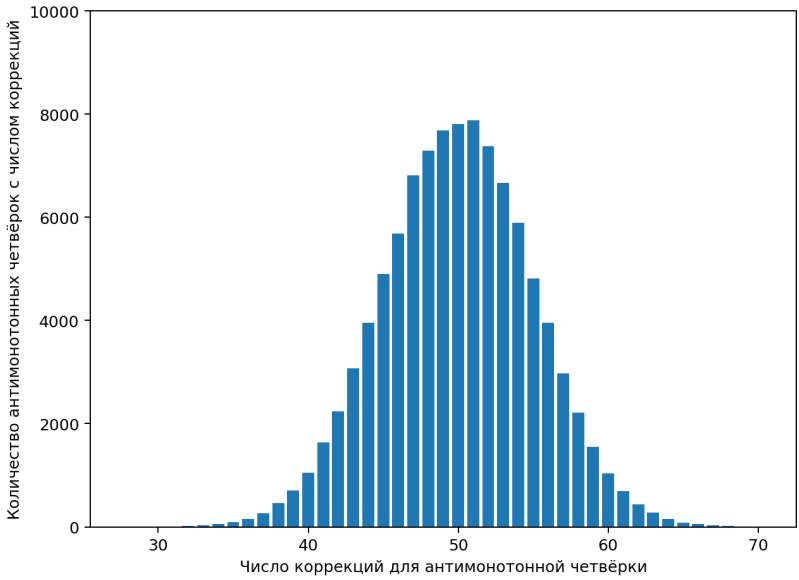

Рис. 4: Распределение числа коррекций для антимонотонных четвёрок (100 000 монотонных и 100 000 антимонотонных четвёрок, 100 запусков).

Кумулятивное распределение (рис. 5) показывает, что при пороге $\theta = 55\%$ количество четвёрок, переклассифицированных из антимонотонных в монотонные, составляет значительную долю от общего числа антимонотонных четвёрок, что позволяет существенно расширить обучающее множество на втором шаге.

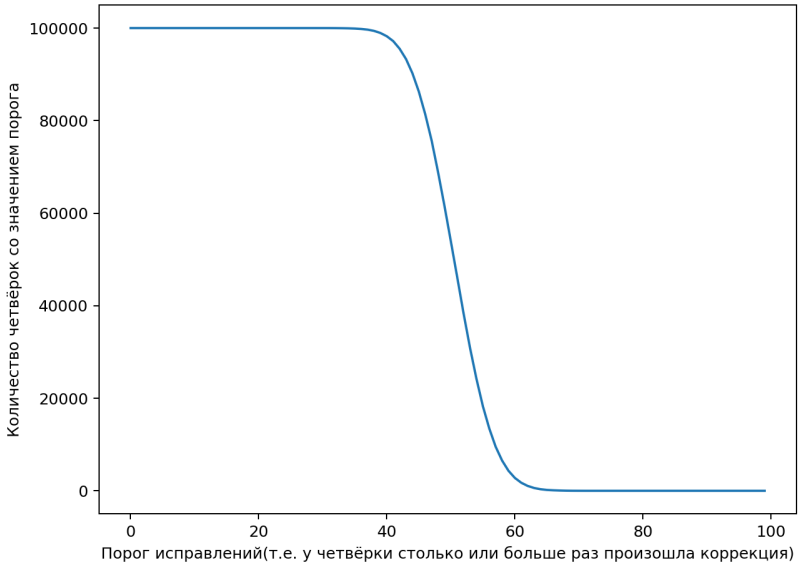

Рис. 5: Кумулятивное распределение: количество четвёрок, для которых число коррекций превышает заданный порог.

6.3. Результаты коррекции

Результаты оценки качества представлений до и после коррекции при пороге $\theta \geq 55\%$ приведены в табл. 2.

Полученные результаты показывают, что при текущей конфигурации эксперимента (100 000 четвёрок, порог 55%) наблюдается незначительное снижение коэффициента корреляции Спирмена по сравнению с лучшим из исходных наборов. Это объясняется несколькими факторами:

1. **Ограниченный размер выборки четвёрок.** При 100 000 четвёрках словарь составил 210 186 слов, однако пересечение с экспертными наборами данных существенно меньше (от 190 до 1 813 пар), что снижает статистическую значимость оценки.
2. **Выбор порога.** Порог $\theta = 55\%$ является предварительным; более тщательный подбор порога может улучшить результаты.

Таблица 2: Результаты коррекции векторных представлений (коэффициент корреляции Спирмена). Порог $\theta \geq 55\%$.

Набор	**Всего пар**	**Модель**	**Исходное**	**До корр.**	**Пар в перес.**	**После корр.**
MEN	3 000	Word2Vec	0,76	0,75	1 813	0,73
		GloVe	0,76			
SimLex	998	Word2Vec	0,41	0,40	586	0,39
		GloVe	0,39			
WordSim	351	Word2Vec	0,71	0,67	190	0,65
		GloVe	0,60			

3. **Масштаб эксперимента.** Для получения более точных оценок необходимо провести эксперимент с 1 000 000 четвёрками, что позволит увеличить покрытие словаря и повысить устойчивость результатов.

Тем не менее, значение «до коррекции» (before correction), вычисленное только на пересечении словарей, уже ниже исходного значения (0,75 против 0,76 для MEN; 0,40 против 0,41 для SimLex; 0,67 против 0,71 для WordSim), что свидетельствует о влиянии ограниченности пересечения. С учётом этого, разница между значениями «до коррекции» и «после коррекции» является менее выраженной, и подход демонстрирует потенциал для дальнейшего улучшения при масштабировании.

7. Обсуждение и направления дальнейших исследований

Предложенный подход обладает рядом преимуществ по сравнению с разработкой новых моделей первичных представлений. Во-первых, он не требует доступа к исходным обучающим корпусам и может быть применён к любым предобученным моделям. Во-вторых, вычислительные затраты на коррекцию существенно ниже, чем на обучение новой модели с нуля. В-третьих, подход является модульным: он может быть применён к произвольному числу источников представлений.

Среди направлений дальнейших исследований можно выделить следующие:

1. **Масштабирование эксперимента.** Увеличение числа четвёрок до 1 000 000 и более для повышения покрытия словаря и устойчивости результатов.
2. **Оптимизация порога.** Систематический подбор порога θ с использованием кросс-валидации.
3. **Использование более сложных архитектур.** Замена одного полносвязного слоя на многослойную нейронную сеть для более гибкого преобразования представлений.
4. **Обобщение на другие модальности.** Применение подхода к векторным представлениям объектов других типов (изображений, аудио, мультимодальных данных).
5. **Понижение размерности.** Исследование возможности одновременного улучшения качества и снижения размерности представлений.
6. **Использование тезаурусов.** Привлечение структурированных лингвистических ресурсов (тезаурусов, онтологий) в качестве дополнительного источника информации о семантической близости.

8. Заключение

В работе описан подход к улучшению качества векторных представлений слов за счёт использования нескольких источников представлений. Введены понятия монотонных и антимонотонных четвёрок слов, сформулирована гипотеза о возможности восстановления истинного порядка близостей на основе согласованной информации из нескольких источников. Предложена двухшаговая процедура коррекции, основанная на quadruplet loss и полносвязном слое нейронной сети.

Экспериментальные результаты на моделях Word2Vec и GloVe, обученных на лемматизированной Википедии, подтверждают работоспособность подхода и указывают на необходимость масштабирования эксперимента для получения статистически значимого улучшения. Описанный подход может быть обобщён для улучшения качества векторных представлений объектов других модальностей.

Список литературы

[1] T. Mikolov, K. Chen, G. Corrado, J. Dean, “Efficient Estimation of Word Representations in Vector Space”, Proceedings of Workshop at ICLR, 2013.

[2] P. Bojanowski, E. Grave, A. Joulin, T. Mikolov, "Enriching Word Vectors with Subword Information", *Transactions of the Association for Computational Linguistics*, **5** (2017), 135–146. DOI: 10.1162/tacl_a_00051.

[3] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Vol. 1), 2019, 4171–4186. DOI: 10.18653/v1/N19-1423.

[4] E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Pasca, A. Soroa, "A Study on Similarity and Relatedness Using Distributional and WordNet-based Approaches", Proceedings of NAACL-HLT, 2009, 19–27. DOI: 10.3115/1620754.1620758.

[5] *Word Embedding*, Electronic resource. Accessed 22.11.2022, https://nlp.gluon.ai/model_zoo/word_embeddings/index.html.

[6] *Word Embedding Evaluation Datasets*, Electronic resource. Accessed 22.11.2022, https://nlp.gluon.ai/api/data.html#word-embedding-evaluation-datasets.

[7] A. Kolosov, A. Maysuradze, "Correction of Vector Representations of Words to Improve the Semantic Similarity", 2021 International Conference on Information Technology and Nanotechnology (ITNT), 2021. DOI: 10.1109/ITNT52450.2021.9649102.

[8] E. Bruni, N.-K. Tran, M. Baroni, "Multimodal Distributional Semantics", *Journal of Artificial Intelligence Research*, **49** (2014), 1–47. DOI: 10.1613/jair.4135.

[9] F. Hill, R. Reichart, A. Korhonen, "SimLex-999: Evaluating Semantic Models With (Genuine) Similarity Estimation", *Computational Linguistics*, **41**:4 (2015), 665–695. DOI: 10.1162/COLI_a_00237.

[10] I. Borg, P. J. F. Groenen, *Modern Multidimensional Scaling: Theory and Applications*, 2nd ed., Springer, 2005, 614. DOI: 10.1007/0-387-28981-X pp.

[11] *NLPL Word Embeddings Repository*, Electronic resource. Accessed 22.11.2022, http://vectors.nlpl.eu/repository.

[12] F. Chung, M. Garrett, R. Graham, D. Shallcross, "Distance Realization Problems with Applications to Internet Tomography", *Journal of Computer and System Sciences*, **63**:3 (2001), 432–448. DOI: 10.1006/jcss.2001.1785.

[13] J. Pennington, R. Socher, C. D. Manning, "GloVe: Global Vectors for Word Representation", Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, 1532–1543. DOI: 10.3115/v1/D14-1162.

[14] F. Schroff, D. Kalenichenko, J. Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, 815–823. DOI: 10.1109/CVPR.2015.7298682.

Статья поступила 3 марта 2026 г.

Improving the quality of vector representations of words by using several sources of representations

A. M. Kolosov, A. I. Maysuradze

Word vector representations are widely used in machine translation, recommender systems, and information retrieval. The quality of such representations, measured as the rank correlation with expert assessments of semantic similarity, remains limited. This paper proposes an approach to improving the quality of word vector representations by merging several independent sources of primary representations. The notions of monotone and antimonotone quadruplets of words are introduced, and the hypothesis that the information contained in monotone quadruplets allows one to recover the true order of similarities for antimonotone quadruplets is formulated and verified. A method for selecting word quadruplets, a two-step correction procedure based on a fully connected layer and a quadruplet loss function, as well as a method for evaluating the quality of the resulting representations are proposed. Experimental results on Word2Vec and GloVe models trained on a lemmatised Wikipedia corpus demonstrate the feasibility of improving representation quality when evaluated on the MEN, SimLex-999, and WordSim-353 expert datasets.

Keywords: word vector representations, semantic similarity, data fusion, quadruplet loss, multidimensional scaling, Word2Vec, GloVe.

References

[1] T. Mikolov, K. Chen, G. Corrado, J. Dean, “Efficient Estimation of Word Representations in Vector Space”, Proceedings of Workshop at ICLR, 2013.

[2] P. Bojanowski, E. Grave, A. Joulin, T. Mikolov, “Enriching Word Vectors with Subword Information”, *Transactions of the Association for Computational Linguistics*, **5** (2017), 135–146. DOI: 10.1162/tacl_a_00051.

[3] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”, Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Vol. 1), 2019, 4171–4186. DOI: 10.18653/v1/N19-1423.

[4] E. Agirre, E. Alfonseca, K. Hall, J. Kravalova, M. Pasca, A. Soroa, “A Study on Similarity and Relatedness Using Distributional and WordNet-based Approaches”, Proceedings of NAACL-HLT, 2009, 19–27. DOI: 10.3115/1620754.1620758.

[5] *Word Embedding*, Electronic resource. Accessed 22.11.2022, https://nlp.gluon.ai/model_zoo/word_embeddings/index.html.

[6] *Word Embedding Evaluation Datasets*, Electronic resource. Accessed 22.11.2022, https://nlp.gluon.ai/api/data.html#word-embedding-evaluation-datasets.

[7] A. Kolosov, A. Maysuradze, "Correction of Vector Representations of Words to Improve the Semantic Similarity", 2021 International Conference on Information Technology and Nanotechnology (ITNT), 2021. DOI: 10.1109/ITNT52450.2021.9649102.

[8] E. Bruni, N.-K. Tran, M. Baroni, "Multimodal Distributional Semantics", *Journal of Artificial Intelligence Research*, **49** (2014), 1–47. DOI: 10.1613/jair.4135.

[9] F. Hill, R. Reichart, A. Korhonen, "SimLex-999: Evaluating Semantic Models With (Genuine) Similarity Estimation", *Computational Linguistics*, **41**:4 (2015), 665–695. DOI: 10.1162/COLI_a_00237.

[10] I. Borg, P. J. F. Groenen, *Modern Multidimensional Scaling: Theory and Applications*, 2nd ed., Springer, 2005, 614. DOI: 10.1007/0-387-28981-X pp.

[11] *NLPL Word Embeddings Repository*, Electronic resource. Accessed 22.11.2022, https://vectors.nlpl.eu/repository.

[12] F. Chung, M. Garrett, R. Graham, D. Shallcross, "Distance Realization Problems with Applications to Internet Tomography", *Journal of Computer and System Sciences*, **63**:3 (2001), 432–448. DOI: 10.1006/jcss.2001.1785.

[13] J. Pennington, R. Socher, C. D. Manning, "GloVe: Global Vectors for Word Representation", Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014, 1532–1543. DOI: 10.3115/v1/D14-1162.

[14] F. Schroff, D. Kalenichenko, J. Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, 815–823. DOI: 10.1109/CVPR.2015.7298682.

Received on March 3, 2026

Преодоление ограничений обучения сиамских нейронных сетей в задаче оптического распознавания символов

А. К. Мокин*

Разработка моделей глубокого обучения для задач классификации представляет особую сложность при работе с большим количеством классов в условиях ограниченных данных и вычислительных ресурсов. Метрическое обучение предлагает перспективный подход к решению этой проблемы, хотя его эффективность часто ограничивается присущими недостатками стандартных функций потерь, таких как контрастивная и триплетная потери, а также неоптимальными стратегиями выбора обучающих примеров. В данной работе представлены решения этих проблем: новый автовероятностный метод майнинга для выбора примеров и улучшенная метрическая функция потерь. Предложенный автовероятностный метод майнинга помогает выбирать наиболее информативные пары примеров для обучения сиамских нейронных сетей. В сочетании с ранее разработанным методом автокластеризации этот метод повышает эффективность обучения, максимизируя полезность данных и минимизируя вычислительные затраты. Помимо этого, вводится новая метрическая функция потерь на основе триплетов, учитывающая специфику кластеров и разработанная для преодоления конкретных недостатков традиционных контрастивной и триплетной функций потерь, тем самым улучшая процесс формирования признаковых эмбеддингов. Эффективность предложенных методов была подтверждена в ходе экспериментов по оптическому распознаванию символов на наборах данных PHD08 (корейский алфавит) и Omniglot. Для полного корейского алфавита в наборе данных PHD08 новая функция потерь со случайным майнингом достигла точности классификации $82,6\%$, установив новый эталонный показатель. Используя сокращенный алфавит, был установлен базовый уровень в $88,6\%$ на PHD08. Применение только автовероятностного метода майнинга улучшило точность до $90,6\%$ ($+2,0\%$), а его комбинация с автокластеризацией дополнительно увеличила её до $92,3\%$ ($+3,7\%$). На наборе данных Omniglot предложенный метод майнинга достиг $92,32\%$, а в

* *Мокин Арсений Кириллович* — аспирант каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: mokinak@my.msu.ru, ORCID: 0009-0005-8044-0245.

Mokin Arseniy Kirillovich — Ph.D. Student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

сочетании с автокластеризацией этот показатель вырос до 93, 17%. Результаты демонстрируют, что предложенные функция потерь и стратегия майнинга представляют собой эффективное решение для задач метрического обучения, особенно в сценариях, характеризующихся большим количеством классов и ограниченными ресурсами.

Ключевые слова: глубокое метрическое обучение, оптическое распознавание символов, сиамские нейронные сети, распознавание паттернов..

1. Введение

Методы глубокого обучения широко используются в последние годы, влияя на такие области, как обработка естественного языка и компьютерное зрение. Сверточные нейронные сети (CNN), способные эффективно строить иерархические представления из исходных данных, стали краеугольным камнем во многих приложениях. Оптическое распознавание символов (OCR) — одно из таких приложений, где CNN показали высокую производительность. Системы OCR необходимы для оцифровки и извлечения текста из документов, изображений и других отсканированных материалов, поскольку они упрощают такие задачи, как текстовый анализ и автоматический ввод данных [1].

Способность CNN автоматически обучаться дискриминативным признакам непосредственно из данных пикселей объясняет их эффективность в задачах OCR [2]. CNN способны улавливать детали, необходимые для точной идентификации символов, используя сверточные слои для извлечения иерархических структур и локальных закономерностей. Например, в простых задачах OCR часто количество нейронов в последнем слое нейронной сети соответствует количеству распознаваемых символов. В таком случае обработка огромного алфавита (Рис. 1) в OCR является нетривиальной задачей.

Рис. 1: Похожие символы Корейского алфавита.

В таких условиях количество нейронов в последнем слое резко возрастает, увеличивая объем данных и требуя большого объема вычислительных мощностей и обучающих данных. В результате обучение этих сетей становится неэффективным.

Метрическое обучение помогает избежать этой проблемы [3]. Метрическое обучение работает путем сопоставления точки в метрическом пространстве с входным вектором, гарантируя, что точка будет расположена как можно ближе к точкам того же класса и как можно дальше от точек других классов [4].

Существует метрический подход, называемый сиамскими нейронными сетями (SNN). SNN, впервые представленные Дж.Бромли и др. [5], привлекли значительное внимание благодаря своей способности обучаться устойчивым представлениям в сценариях с ограниченными размеченными данными. В отличие от традиционных архитектур CNN, которые направлены на непосредственное предсказание классов, SNN изучают метрику сходства между входами, позволяя им улавливать тонкие различия и сходства даже без достаточного количества размеченных данных.

SNN состоит из нескольких ветвей CNN с общими весами. Часто евклидова норма используется для расчета расстояний между выходами ветвей, и полученные векторы затем отправляются в функцию потерь.

Основной вклад данной работы заключается в следующем:

- Разработка метода майнинга для обучения SNN, который улучшает представление данных и снижает вычислительные затраты.
- Предложение новой метрической функции потерь, которая сочетает преимущества контрастивной и триплетной функций потерь, вводя новый механизм для обеспечения компактности кластеров.
- Экспериментальная оценка эффективности предложенных методов в задачах OCR с использованием наборов данных PHD08 и Omniglot, демонстрирующая значительное улучшение точности классификации и вычислительной эффективности.

2. Связанные работы

OCR включает классификацию символов и их сопоставление с соответствующими классами. В то время как для некоторых языков, таких как английский, достигнуты высокие показатели распознавания [6], задача становится все более сложной, когда размер алфавита превышает 10000, требуя новых методологий обучения и распознавания.

2.1. Контрастивная функция потерь

Контрастивная функция потерь [7, 8] (1), первоначально предложенная С.Чопра и др. для задач верификации лиц, стала популярна в метрическом обучении, как показано в исследованиях [9–11].

Контрастивная функция потерь (1) использует попарное расстояние и приближает положительные пары как можно ближе друг к другу и отталкивает отрицательные пары на расстояние друг от друга не меньше выбранного порога (α).

$$L^{\alpha}(x_i, x_j; f) = y_{ij} d_{ij}^2 + (1 - y_{ij}) \cdot max(0, \alpha - d_{ij})^2, \tag{1}$$

где $d_{ij} = \|f(x_i) - f(x_j)\|_2$ — евклидово расстояние между парой, а y_{ij} возвращает 1, если $y_i = y_j$, и 0 в противном случае. x_i, x_j представляют входы SNN (f).

Отсутствие естественного механизма остановки является одним из заметных недостатков контрастивной функции потерь. Эта функция потерь способствует минимизации расстояния между положительной парой до нуля. На практике нейронная сеть может использовать вычислительные ресурсы для уточнения пар, которые уже близки к сходимости, не позволяя ей исследовать и оптимизировать другие примеры.

В работе [12] упоминается другая интерпретация контрастивной функции потерь, которая имеет схожий смысл, но сформулирована иначе. В данной работе используется классическая версия контрастивной функции потерь, предложенная в [7, 8].

2.2. Триплетная функция потерь

Триплетная функция потерь (2), предложенная и описанная в [13], выступает альтернативным подходом к обучению эмбеддингов в задачах метрического обучения. В отличие от контрастивной функции потерь, которая работает с парами образцов, триплетная функция потерь работает с триплетами, состоящими из якоря, положительного примера из того же класса, что и якорь, и отрицательного примера из другого класса.

$$L^{\alpha}(x_a, x_p, x_n, f) = max(0, \|f(x_a) - f(x_p)\|_2 - \|f(x_a) - f(x_n)\|_2 + \alpha) \tag{2}$$

где x_a, x_p, x_n представляют якорь, положительный и отрицательный входы SNN (f).

Цель триплетной функции потерь — обучить сеть уменьшать расстояние между якорями и положительными примерами, одновременно

увеличивая расстояние между якорями и отрицательными примерами как минимум на заданный порог (α).

Однако важно отметить, что триплетная функция потерь вводит кубическую сложность [14], что означает, что возможных триплетов значительно больше, чем пар. Не все триплеты одинаково информативны для обучения, и включение всех возможных триплетов может привести к более медленной сходимости и вычислительной неэффективности. Поэтому выбор «сложных» триплетов, которые особенно информативны [15], эффективен для обучения модели и достижения более быстрой сходимости и улучшенной производительности.

Предлагаемый в этой работе метод майнинга (Раздел 4) представляет стратегию для решения проблемы «сложных» триплетов.

Другая проблема, связанная с триплетной функцией потерь, связана с отсутствием прямого требования близости между якорем и положительными экземплярами. Акцент делается на обеспечении того, чтобы расстояние между якорями и положительными примерами было меньше расстояния между якорями и отрицательными примерами на заданный порог (α).

Предлагаемая метрическая функция потерь (Раздел 4.4) устраняет ключевые ограничения классических функций потерь (контрастивная и триплетная), улучшает способность модели изучать дискриминативные признаки и достигать лучшей производительности в задачах метрического обучения.

2.3. Квадруплетная функция потерь

Существует квадруплетная функция потерь [16], которая расширяет традиционную триплетную функцию потерь за счет введения дополнительного отрицательного примера. Этот подход накладывает более строгое ограничение, гарантируя, что пара якорь-положительный пример не только ближе, чем первый отрицательный пример, но и ближе, чем второй отрицательный пример, тем самым улучшая дискриминацию признаков. Хотя этот метод улучшает обобщаемость в метрическом обучении, он также увеличивает вычислительную сложность и требует более продвинутых стратегий майнинга для эффективного отбора примеров. Несмотря на эти проблемы, квадруплетная функция потерь продемонстрировала улучшенную производительность.

$$\begin{aligned} L^{\alpha,\beta}(x_a, x_p, x_{n_1}, x_{n_2}) = &\max\left(0, \|f(x_a) - f(x_p)\|_2 - \|f(x_a) - f(x_{n_1})\|_2 + \alpha\right) \\ &+ \max\left(0, \|f(x_a) - f(x_p)\|_2 - \|f(x_{n_1}) - f(x_{n_2})\|_2 + \beta\right) \end{aligned} \tag{3}$$

2.4. Декомпозиция символов

Некоторые языки допускают декомпозицию символов, позволяя сегментировать их на отдельные компоненты для распознавания и последующей композиции конечного результата [9, 17]. Этот подход устраняет необходимость в нейронных сетях с десятками и даже сотнями тысяч выходов. Однако он обладает недостатками, такими как уязвимость к искажениям изображения и сильная зависимость от качества сегментации. Некоторые исследователи [18] выбирают глубокие нейронные сети с большим количеством обучаемых параметров, которые предлагают высокое качество, но требуют значительных вычислительных ресурсов и не подходят, например, для мобильных устройств и т.п.

2.5. Методы майнинга

Кроме того, обучение метрических сетей осложнено выбором правил генерации пар/триплетов. Хотя случайный майнинг (random-mining) является обычным явлением [19], в некоторых работах уделяют особое внимание решению этой проблемы, сосредотачиваясь на сходстве. Например, в [20] используется агрессивная стратегия жесткого майнинга (hard-mining), выбирающая пары с наибольшей ошибкой для обратного распространения, чтобы обучать сеть исключительно на сложных примерах. Недостатком метода является то, что возможный шум в данных может затруднить достижение локального минимума.

Другой интересный момент — математическое обоснование [21] эффективности жесткого майнинга в метрическом обучении с использованием теоремы изометрической аппроксимации. Авторы показывают, что жесткий майнинг эквивалентен минимизации расстояния Хаусдорфа между нейронной сетью и ее идеальной функцией, что объясняет эмпирический успех этого подхода.

В [22] авторы предлагают Easy Positive Triplet Mining (EPTM) для решения проблемы нестабильности традиционных методов жесткого майнинга. Hard Positive Mining часто выбирает выбросы или неправильно размеченные примеры, что приводит к зашумленной оптимизации и плохому обобщению. EPTM смягчает эту проблему, выбирая более простые, но информативные положительные примеры, обеспечивая лучшую кластеризацию при сохранении стабильного обучения. В сочетании с Semi-Hard Negative Mining, который избегает экстремальных отрицательных примеров, этот подход приводит к более надежным эмбеддингам и улучшенной сходимости.

Кроме того, методы майнинга пар на основе расстояний, такие как подход, предложенный в [10], демонстрируют эффективные результаты

за счет генерации образцов на основе создания вектора расстояний для всех возможных негативных пар. Хотя этот тип метода перспективен с точки зрения качества обучения, он требует значительных временных затрат.

Метод авто-кластеризации [23] частично решает проблему майнинга в SNN. Этот метод основан на создании кластеров, то есть групп, состоящих из классов, схожих с точки зрения нейросети. Использование кластеров позволяет нейросети уделять больше внимания классам, которые трудно различить. Согласно методу авто-кластеризации [23], отрицательный пример выбирается из того же кластера, что и положительный пример.

В процессе обучения нейронной сети можно выяснить, какие классы находятся далеко от своего кластера в метрическом пространстве, и попытаться вручную увеличить вероятность их выбора при генерации положительного класса. Для такой задачи автоматические методы рекомендуются как наиболее эффективные и удобные решения, не требующие тяжелой настройки.

3. Данные

3.1. Распознавание корейского алфавита

3.1.1. Набор данных PHD08

Корейский алфавит (Хангыль) известен своим большим набором символов и высоким визуальным сходством между многими из них, что делает его особенно сложным для задач OCR. Это хорошо согласуется с целями данного исследования, поскольку создается основа для экспериментов, которые проверяют устойчивость и дискриминативную способность предложенных методов. Таким образом, для оценки предложенных методов был выбран набор данных рукописных символов хангыля под названием PHD08 [24]. Более того, тот же набор данных использовался в предыдущих работах [9, 10, 23]. Набор данных содержит 2350 классов, и каждый класс имеет 2187 изображений корейских символов. Всего имеется 5139450 бинарных изображений разных размеров, с разными поворотами и искажениями. Примеры изображений набора данных показаны на Рис. 2.

Рис. 2: Примеры из набора данных PHD08.

3.1.2. Синтетические обучающие данные

Для оценки объективности предложенного подхода были сгенерированы синтетические данные с использованием того же подхода, что и в [23], для обучения нейросети таким же образом, как и в работах [9,10,23], используя в среднем восемь аналогичных шрифтов для каждого класса. Общее количество классов составило 11172 символа корейского алфавита. Более того, в проведенных экспериментах было решено не генерировать все возможные классы и сделать количество классов равным размеру набора данных для оценки. Это решение было принято для более точной оценки предложенного авто-вероятностного метода майнинга (Раздел 4.2), потому что сеть могла обучаться на некоторых классах, которых нет в наборе данных для оценки. И этот факт в сочетании с рассматриваемым подходом мог значительно увеличить вероятности появления таких ненужных для оценки символов.

3.2. Набор данных Omniglot

Набор данных Omniglot был собран Б.Лейком и его сотрудниками в MIT через Amazon Mechanical Turk для создания стандартного теста для обучения на небольшом количестве примеров в области распознавания рукописных символов [25].

Omniglot содержит 1623 символа из 50 различных алфавитов (Рис. 3). Он состоит не только из международных языков, таких как корейский и латинский, но и из менее известных местных диалектов и вымышленных наборов символов, таких как Futurama и клингонский. Каждый из них был нарисован вручную 20 разными людьми. Количество букв в каждом алфавите значительно варьируется приблизительно от 15 до 40 символов. Набор данных состоит из двух подмножеств: *images_background*, содержащий 19289 изображений, и *images_evaluation*, содержащий 13180 изображений.

3.3. Аугментация

Аугментация данных [23] применялась к изображениям в процессе обучения с вероятностью 0.7 для каждого образца и со следующими искажениями:

- Проективное преобразование — каждая точка изображения преобразуется, и значения смещения по осям x и у выбираются случайным образом в диапазоне $[0.0, 1.0]$, где минимальная и максимальная доли смещения представлены на оси x и оси y от ширины изображения.

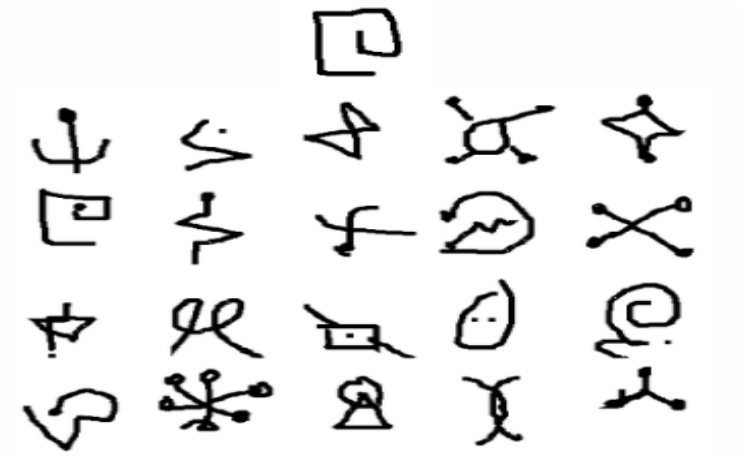

Рис. 3: Примеры набора данных Omniglot из исходной статьи [26].

- Поворот — вращение изображения на угол в диапазоне $[-5, 5]$ градусов.
- Масштабирование — масштабирование изображения до заданного размера, а затем масштабирование результата до исходного размера с коэффициентами масштабирования $min = 0.7$ и $max = 0.9$ от исходного изображения по ширине и высоте, чтобы сделать изображение пикселизированным.

3.4. Настройки для экспериментов

Для всех экспериментов в этой статье изображения изменялись до размера 37×37 пикселей и преобразовывались в оттенки серого, чтобы соответствовать стандартам предобработки предыдущих исследований для объективного сравнительного анализа.

В течение каждой эпохи выполнялось 50 итераций с генерацией 10240 элементов (пар/триплетов) на итерацию. Для контрастивной функции потерь пары динамически выбирались (3072 подлинных пары и 7168 пар-самозванцев), в то время как CATML использовала триплеты. Всего 10240 элементов генерировались как для обучающего, так и для валидационного наборов в каждую эпоху. Эти настройки применялись единообразно ко всем наборам данных, используемым в экспериментах, включая PHD08 и Omniglot, чтобы сохранить условия обучения равными и произвести объективную оценку предложенных методов.

4. Предлагаемый метод

4.1. Авто-вероятностный метод майнинга

Присвоение определенных вероятностей классам во время обучения нейронной сети может улучшить способность модели распознавать определенные классы. Приоритезация некоторых классов с большей вероятностью побуждает нейросеть выбирать их чаще, что улучшает способность нейросети распознавать эти символы.

Присвоение вероятностей вручную вызывает затруднения на практике. Во-первых, когнитивная нагрузка по поддержанию всесторонних знаний о нескольких разнообразных классах выходит за пределы человеческих возможностей, когда количество классов исчисляется тысячами. Кроме того, хотя автоматическое присвоение вероятности возможно для некоторых организованных данных, таких как языки с декомпозицией символов на основе ключей, многие реальные ситуации не имеют такой внутренней структуры, что делает присвоение вручную проблематичным. При этом, автоматизация может быть применена к более широкому кругу объектов, независимо от их типа или сложности.

В результате авто-вероятностный метод майнинга (Auto-Probabilistic Method — APM) предлагает более гибкий подход. Нейросеть приобретает способность динамически адаптироваться и расставлять приоритеты классам в соответствии с их значимостью для обучения путем автоматического вычисления вероятностей для каждого класса во время обучения. Это позволяет нейросети определять, какие классы требуют большего внимания, и выбирать их чаще, что улучшает распознавание и оптимизирует процесс обучения. Следовательно, метод APM предлагает перспективный способ повышения эффективности обучения нейронных сетей. Этот подход отличается от методов равномерного распределения классов и ищет классы с наибольшими средними расстояниями между каждым примером класса и соответствующим ему кластером.

4.2. Автоматическое вычисление вероятностей появления классов

Для улучшения процесса обучения с использованием метода APM вычисляется вектор вероятностей классов P^e на каждой эпохе e. Метод использует средние расстояния между примерами и соответствующими центроидами их классов для определения вероятностей классов.

Пусть $f(x_{i,j})$ — вектор признаков входного изображения $x_{i,j}$ (i-й класс, j-й образец), полученный из модели f. Определим d как расстояние ($L2$

норма разности) между $f(x_{i,j})$ и центром c_i i-го класса, где c_i представляет собой средний вектор признаков примеров класса i. Определим вектор вероятностей классов $\hat{P}^e$ как

$$\hat{P}_i^e = \frac{\left(\frac{1}{M_i}\sum_{j=1}^{M_i} d(f(x_{i,j}), c_i)\right)^\gamma}{\sum_{k=1}^{N}\left(\frac{1}{M_k}\sum_{j=1}^{M_k} d(f(x_{k,j}), c_k)\right)^\gamma}, \quad i = 1, \ldots, N$$

где

- M_i — количество примеров i-го класса;
- $x_{i,j}$ — j-й пример i-го класса;
- N — общее количество классов;
- e — количество эпох;
- γ — коэффициент усиления, контролирующий влияние расстояний для выделения доминирующих классов.

Тогда итоговый вектор вероятностей классов P^e на эпохе e вычисляется как

$$P^e = (1 - w) \cdot \hat{P}^e + w \cdot P^{e-1}, \tag{4}$$

где

- P^{e-1} — вектор вероятностей классов с предыдущей эпохи ($P_i^0 = \frac{1}{N}, \quad i = 1, ..., N$);
- $w \quad (0 \leqslant w \leqslant 1)$ — весовой коэффициент, контролирующий вклад вероятностей предыдущей эпохи;
- γ - степень усиления для обновленных вероятностей для выделения доминирующих классов.

Наконец, P^e нормализуется для формирования дискретного распределения вероятностей, которое используется для майнинга положительных пар во время обучения. Таким образом, это распределение гарантирует, что положительные пары выбираются более эффективно за счет приоритизации классов с большими расстояниями от текущих образцов.

4.3. Улучшения метода авто-кластеризации

Базовый метод авто-кластеризации [23] включает в себя следующие шаги: вычисление всех норм между центроидами (идеальными векторами) каждого класса, их сортировка и, поскольку их может быть много, выбор только некоторых из них.

В предыдущей версии метода авто-кластеризации [23] было много гиперпараметров, необходимых для обучения сети. И такой подход неудобен для экспериментов. На этом этапе было решено упростить, но сохранить основную суть и оставить всего два параметра:

- вероятность выбора класса из кластера для отрицательной пары/триплета (θ);
- количество наименьших норм, рассматриваемых для генерации кластеров (η).

Этот метод используется для майнинга отрицательных примеров в парах/триплетах во время обучения.

4.4. Метрическая функция потерь на основе триплетов с учетом кластеров

Предлагается новая метрическая функция потерь на основе триплетов с учетом кластеров (Cluster-aware triplets-based metric loss - CATML) (5), сочетающая преимущества контрастивной и триплетной функций потерь.

$$CATML = \rho \cdot g_1 + \tau \cdot g_2 + \xi \cdot g_3 \tag{5}$$

где

- $\rho \cdot g_1$ — вклад контрастивной функции потерь, где ρ отвечает за уменьшение расстояния между якорем и положительным примером;
- $\tau \cdot g_2$ — вклад триплетной функции потерь, где τ регулирует принцип, чтобы расстояние между якорем и положительным примером было меньше расстояния между якорем и отрицательным примером на выбранное значение, равное или большее α;
- $\xi \cdot g_3$ — вклад кластеров, где ξ отвечает за стабильность, чтобы кластеры, созданные во время обучения, не распадались и чаще получали градиент, чтобы оставаться на том же месте.

В этой работе экспериментально выбраны значения: $\rho = 0.1$, $\tau = 1.0$, $\xi = 1.0$ и $\alpha = 10$. Компоненты g_1, g_2 и g_3 определяются следующим образом:

- $g_1 = d_{AP} = \|s(f(x_a)) - s(f(x_p))\|_2$;
- $g_2 = s(d_{AP} - d_{AN} + \alpha) =$
  $= s(\|s(f(x_a)) - s(f(x_p))\|_2 - \|s(f(x_a)) - s(f(x_n))\|_2 + \alpha)$, где
  α значение порога;

- $g_3 = \frac{1}{M_i} \sum_{j=1}^{M_i} d(f(x_{i,j}), c_i),\ i = 1, \ldots, N$, где $x_{i,j} \in (x_a, x_p, x_n)$, c_i - идеальный вектор, соответствующий определенному классу (средний вектор примеров данного класса).

Здесь f обозначает модель глубокого обучения, используемую для извлечения признаков, а $s(x) = softrelu(x)$ — функция активации.

5. Эксперименты

5.1. Архитектура

Архитектура (Таб. 1) выбрана из работ [9, 10, 23] для объективности в сравнении.

Таблица 1: Список слоев метрической нейронной сети.

Layer	**Type**	**Parameters**	**Output**
1	conv	16 filters 3×3, stride 1	$35 \times 35 \times 16$
2	conv	16 filters 5×5, stride 2	$18 \times 18 \times 16$
3	conv	16 filters 3×3, stride 1	$18 \times 18 \times 16$
4	conv	24 filters 5×5, stride 2	$9 \times 9 \times 24$
5	conv	24 filters 3×3, stride 1	$9 \times 9 \times 24$
6	conv	24 filters 3×3, stride 1	$9 \times 9 \times 24$
7	fc	25 outputs	$1 \times 1 \times 25$

Функция активации *softsign* (6) подробно описана в работах [27, 28] и рекомендуется как наиболее подходящая для такого рода проблем. *Softsign* — это ограниченная функция активации, которая способствует стабильной сходимости и устраняет любые численные несоответствия. Эта функция также использовалась в работах [9, 10, 23].

$$softsign(x) = \frac{x}{1 + |x|} \tag{6}$$

5.2. Обучение

5.2.1. Синтетические данные

Проведены эксперименты, в которых сеть обучалась на синтетических изображениях корейских символов с новой функцией CATML. Были рассмотрены два размера обучающего алфавита: полный (11172 класса) и

сокращенный (2350 классов). В оцениваемом наборе данных PHD08 всего 2350 классов, поэтому сокращенный подход позволяет методу APM оказывать более объективное влияние и избегает фокусирования на ненужных для оценки классах.

Была обучена нейронная сеть с CATML на полном алфавите с методом случайного майнинга (random-mining). В следующем эксперименте нейросеть обучалась с той же функцией потерь на сокращенном алфавите со случайным майнингом, авто-кластеризацией [23] и авто-вероятностным майнингом. Более того, специфика двух последних методов позволяет использовать их вместе, где метод APM используется для майнинга положительных примеров, а авто-кластеризация [23] отвечает за отрицательные примеры.

В проведенных экспериментах метод APM показал наилучшую точность только при $\gamma = 1$, но для метода APM + Авто-кластеризация [23] наилучший результат показал $\gamma = 2$. Метод авто-кластеризации [23] продемонстрировал высокую точность при $\theta = 0.5$ и $\eta = 1000$, а метод APM достиг наилучшего результата при $w = 0$, что означает отсутствие зависимости от вероятностей с предыдущей эпохи, но этот весовой коэффициент также может быть рассмотрен для будущих экспериментов.

5.2.2. Набор данных Omniglot

Для набора данных Omniglot была обучена нейронная сеть на выделенных 60% от общего объема данных. Тестовая и валидационная части были установлены такими же, как в [2], обе равны 20%. Было зафиксировано единое количество обучающих примеров на алфавит, чтобы каждый алфавит встречался с одинаковой частотой во время обучения, хотя это не гарантируется для отдельных классов символов в каждом алфавите. Все параметры, использованные для предложенных методов, были такими же, как и для экспериментов на корейском алфавите.

5.3. Оценка

Точность (Acc) для классификации определялась как

$$Acc = \frac{N_{correct}}{N_{total}} \cdot 100\%, \tag{7}$$

где N_{total} — размер набора данных, а $N_{correct}$ — количество изображений, правильно классифицированных нейросетью.

6. Результаты

Все эксперименты следовали общему формату оценки с разделением данных на три отдельных набора: обучающий, валидационный и тестовый. Валидационный набор использовался исключительно для настройки гиперпараметров и выбора модели посредством мониторинга сходимости по эпохам, в то время как тестовый набор оставался строго изолированным для окончательной оценки. Итоговые метрики вычислялись путем однократной оценки на тестовом наборе с использованием оптимальной контрольной точки модели, выбранной на основе ошибки валидационной части. Для обеспечения воспроизводимости и предотвращения переобучения никакие обновления параметров или архитектурные решения не предпринимались под влиянием наблюдений тестового набора во время обучения.

Для каждого эксперимента предоставлена таблица результатов, содержащая как обучающую, так и тестовую точность для выбранной модели. Метрики на обучающем наборе данных включены специально для проверки поведения модели; соответствие точности между обучающей и тестовой выборками предоставляет эмпирические доказательства отсутствия переобучения.

6.1. PHD08

Результаты экспериментов с большим количеством классов (11172) представлены в Таб. 2. Результаты случайного майнинга, жесткого майнинга и майнинга на основе расстояний были взяты из работы [10]. Функция CATML показала заметную точность по сравнению со всеми предыдущими результатами.

Таблица 2: Точность классификации для набора данных PHD08 с размером алфавита 11172 классов.

Loss	**Mining Method**	**Train Acc**	**Test Acc**
Contrastive loss	Random-mining [19]	-	63.7%
Contrastive loss	Hard-mining [20]	-	64.5%
Contrastive loss	Distance-based mining [10]	-	69.7%
Contrastive loss	Auto-clustering [23]	79.3%	76.1%
CATML	**Random-mining**	86.2%	**82.6**%

В Таб. 3 видно, что предложенный метод АРМ улучшает точность классификации. Более того, предполагается, что этот метод является

эффективным и подходящим решением независимо от типа распознаваемого объекта. Кроме того, он может успешно применяться как с ранее предложенным методом авто-кластеризации [23], так и без него.

Таблица 3: Точность классификации для набора данных PHD08 с размером алфавита 2350 классов.

Loss	**Mining Method**	**Train Acc**	**Test Acc**
CATML	Random-mining	89.1%	88.6%
CATML	Auto-probabilistic	94.8%	90.6%
CATML	Auto-clustering	93.7%	91.1%
CATML	**Auto-probabilistic + Auto-clustering**	95.1%	**92.3**%

6.2. Omniglot

Таблица 4: Точность классификации для набора данных Omniglot.

Loss	**Mining Method**	**Train Acc**	**Test Acc**
CATML	Random-mining	93.33%	91.25%
CATML	Auto-clustering	95.21%	91.60%
CATML	Auto-probabilistic	94.78%	92.32%
CATML	**Auto-probabilistic + Auto-clustering**	95.89%	**93.17**%

В Таб. 4 можно увидеть, что использование предложенных методов улучшает точность классификации. Также стоит отметить, что метод авто-кластеризации [23] не улучшает точность значительно. Это можно объяснить слишком большим разнообразием набора данных, поскольку он содержит визуально различные алфавиты, в том числе включает корейский.

7. Выводы

В работе предложен авто-вероятностный метод майнинга, который повышает точность распознавания сиамских нейронных сетей, делая их более эффективными для задач классификации, таких как OCR. Этот метод может использоваться как в сочетании с методом авто-кластеризации, так и без него. Он улучшает обучение сети, достигая высокой точности, как продемонстрировано на наборах данных PHD08 и Omniglot. Разработанный метод не требует каких-либо знаний о природе объекта и может использоваться для обучения нейронных сетей для распознавания

любого типа объектов: символы, дескрипторы особых точек, лица и так далее. В дополнение к этому, предложенная метрическая функция потерь на основе триплетов с учетом кластеров сочетает в себе преимущества контрастивной и триплетной функций потерь.

В будущих исследованиях предложенные методы, включая новую функцию потерь и стратегию авто-вероятностного майнинга, могут быть рассмотрены в контексте обучения с одним примером (one-shot learning) и повторной идентификации (re-identification). Эти задачи вызывают сложности в условиях ограниченных данных и необходимости надежного определения схожести, что хорошо согласуется с сильными сторонами метрического обучения. Применяя предложенные методы к этим задачам, можно исследовать их потенциал к повышению обобщающей способности, снижению зависимости от больших наборов данных и достижению конкурентоспособной производительности.

Список литературы

[1] Khaustov P.A., “Algorithms for handwritten character recognition based on constructing structural models”, *Comput. Opt.*, **41** (2017), 67–78. DOI: 10.18287/2412-6179-2017-41-1-67-78.

[2] Koch G., Zemel R., Salakhutdinov R., “Siamese neural networks for one-shot image recognition”, *Proceedings of the 32nd International Conference on Machine Learning*, 2015, 7–9 July 2015; Volume 2.

[3] Hoffer E., Ailon N., “Deep metric learning using triplet network”, *Proceedings of the International Workshop on Similarity-Based Pattern Recognition*, 2015, 84–92.

[4] Oh Song H., Xiang Y., Jegelka S., Savarese S., “Deep metric learning via lifted structured feature embedding”, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, 4004–4012.

[5] Bromley J., Guyon I., LeCun Y., Säckinger E., Shah R., “Signature verification using a “siamese” time delay neural network”, *Adv. Neural Inf. Process. Syst.*, **6** (1993), 737–744.

[6] Tafti A.P., Baghaie A., Assefi M., Arabnia H.R., Yu Z., Peissig P., “OCR as a service: An experimental evaluation of Google Docs OCR, Tesseract, ABBYY FineReader, and Transym”, *Proceedings of the 12th International Symposium on Visual Computing*, 2016, 735–746.

[7] Chopra S., Hadsell R., LeCun Y., “Learning a similarity metric discriminatively, with application to face verification”, *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, **1** (2005), 539–546. DOI: 10.1109/CVPR.2005.202.

[8] Hadsell R., Chopra S., LeCun Y., “Dimensionality reduction by learning an invariant mapping”, *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*, **2** (2006), 1735–1742. DOI: 10.1109/CVPR.2006.100.

[9] Ilyuhin S.A., Sheshkus A.V., Arlazarov V.L., “Recognition of images of Korean characters using embedded networks”, *Proceedings of the Twelfth International Conference on Machine Vision (ICMV 2019)*, **11433** (2020), 1143311.

[10] Kondrashev I.V., Sheshkus A.V., Arlazarov V.V., “Distance-based online pairs generation method for metric networks training”, *Proceedings of the Thirteenth International Conference on Machine Vision*, **11605** (2021), 1160508.

[11] Wang X., Hua Y., Kodirov E., Hu G., Robertson N.M., “Deep metric learning by online soft mining and class-aware attention”, *Proceedings of the AAAI Conference on Artificial Intelligence*, **33** (2019), 5361–5368. DOI: 10.1609/aaai.v33i01.33015361.

[12] Wang F., Liu H., “Understanding the behaviour of contrastive loss”, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, 2495–2504.

[13] Schroff F., Kalenichenko D., Philbin J., “Facenet: A unified embedding for face recognition and clustering”, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, 815–823.

[14] Yuan Y., Chen W., Yang Y., Wang Z., “In defense of the triplet loss again: Learning robust person re-identification with fast approximated triplet loss and label distillation”, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, 354–355.

[15] Dalwadi Bijal N., Shah Vatsal, “Deep Embedding Learning for Printed Word Spotting using Triplet CNNs and Transfer Learning”, *Proceedings of the IEEE 5th International Conference on Soft Computing for Security Applications (ICSCSA)*, **73** (2025), 384–389.

[16] Chen W., Chen X., Zhang J., Huang K., “Beyond triplet loss: A deep quadruplet network for person re-identification”, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, 403–412.

[17] Franken M., van Gemert J.C., “Automatic Egyptian hieroglyph recognition by retrieving images as texts”, *Proceedings of the 21st ACM international conference on Multimedia*, 2013, 765–768. DOI: 10.1145/2502081.2502199.

[18] Kim Y.g., Cha E.y., “Learning of Large-Scale Korean Character Data through the Convolutional Neural Network”, *Proceedings of the Korean Institute of Information and Commucation Sciences Conference*, 2016, 97–100.

[19] Bell S., Bala K., “Learning visual similarity for product design with convolutional neural networks”, *ACM Trans. Graph. (TOG)*, **34** (2015), 1–10. DOI: 10.1145/2766959.

[20] Simo-Serra E., Trulls E., Ferraz L., Kokkinos I., Fua P., Moreno-Noguer F., “Discriminative learning of deep convolutional feature point descriptors”, *Proceedings of the IEEE International Conference on Computer Vision*, 2015, 118–126.

[21] Xu A., Hsieh J.Y., Vundurthy B., Cohen E., Choset H., Li L., “Mathematical Justification of Hard Negative Mining via Isometric Approximation Theorem”, *arXiv*, 2022.

[22] Xuan H., Stylianou A., Pless R., “Improved embeddings with easy positive triplet mining”, *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, 2474–2482.

[23] Mokin A.K., Gayer A.V., Sheshkus A.V., Arlazarov V.L., “Auto-clustering pairs generation method for Siamese neural networks training”, *Proceedings of the Fourteenth International Conference on Machine Vision (ICMV 2021)*, **12084** (2022), 369–376. DOI: 10.1117/12.2623139.

[24] Ham D.S., Lee D.R., Jung I.S., Oh I.S., “Construction of printed Hangul character database PHD08”, *J. Korea Contents Assoc.*, **8** (2008), 33–40.

[25] Lake B., Salakhutdinov R., Gross J., Tenenbaum J., “One shot learning of simple visual concepts”, *Proceedings of the Annual Meeting of the Cognitive Science Society*, **33** (2011).

[26] Lake B.M., Salakhutdinov R., Tenenbaum J.B., "Human-level concept learning through probabilistic program induction", *Science*, **350** (2015), 1332–1338. DOI: 10.1126/science.aab3050.

[27] Bergstra J., Desjardins G., Lamblin P., Bengio Y., "Quadratic polynomials learn better image features", *Tech. Rep.*, 2009.

[28] Lin G., Shen W., "Research on convolutional neural network based on improved Relu piecewise activation function", *Procedia Comput. Sci.*, **131** (2018), 977–984. DOI: 10.1016/j.procs.2018.04.239.

Статья поступила 9 февраля 2026 г.

Addressing the Training Challenges of Siamese Networks for Optical Character Recognition

A. K. Mokin

The development of deep learning models for classification tasks poses particular challenges when dealing with a large number of classes under conditions of limited data and computational resources. Metric learning offers a promising approach to solving this problem, although its effectiveness is often limited by inherent shortcomings of standard loss functions, such as contrastive and triplet losses, as well as suboptimal training sample selection strategies. This paper presents solutions to these issues: a novel autoprobabilistic mining method for example selection and an improved metric loss function. The proposed autoprobabilistic mining method aids in selecting the most informative example pairs for training Siamese neural networks. In combination with a previously developed autoclustering method, this approach enhances training efficiency by maximizing data utility while minimizing computational costs. Additionally, a new triplet-based metric loss function is introduced, which accounts for cluster-specific characteristics and is designed to overcome specific drawbacks of traditional contrastive and triplet loss functions, thereby improving the process of feature embedding formation. The effectiveness of the proposed methods was validated through experiments on optical character recognition using the PHD08 (Korean alphabet) and Omniglot datasets. For the full Korean alphabet in the PHD08 dataset, the novel loss function with random mining achieved a classification accuracy of 82.6%, establishing a new benchmark. Using a reduced alphabet, a baseline of 88.6% was set on PHD08. The application of the auto-probabilistic mining method alone improved accuracy to 90.6% (+2.0%), and its combination with auto-clustering further increased it to 92.3% (+3.7%). On the Omniglot dataset, the proposed mining method attained 92.32%, which rose to

93.17% when coupled with auto-clustering. The results demonstrate that the proposed loss function and mining strategy offer a robust and effective solution for complex pattern recognition tasks, especially in scenarios characterized by a high number of classes and resource limitations.

Keywords: deep metric learning, optical character recognition, siamese neural networks, pattern recognition.

References

[1] Khaustov P.A., "Algorithms for handwritten character recognition based on constructing structural models", *Comput. Opt.*, **41** (2017), 67–78. DOI: 10.18287/2412-6179-2017-41-1-67-78.

[2] Koch G., Zemel R., Salakhutdinov R., "Siamese neural networks for one-shot image recognition", *Proceedings of the 32nd International Conference on Machine Learning*, 2015, 7–9 July 2015; Volume 2.

[3] Hoffer E., Ailon N., "Deep metric learning using triplet network", *Proceedings of the International Workshop on Similarity-Based Pattern Recognition*, 2015, 84–92.

[4] Oh Song H., Xiang Y., Jegelka S., Savarese S., "Deep metric learning via lifted structured feature embedding", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, 4004–4012.

[5] Bromley J., Guyon I., LeCun Y., Säckinger E., Shah R., "Signature verification using a "siamese" time delay neural network", *Adv. Neural Inf. Process. Syst.*, **6** (1993), 737–744.

[6] Tafti A.P., Baghaie A., Assefi M., Arabnia H.R., Yu Z., Peissig P., "OCR as a service: An experimental evaluation of Google Docs OCR, Tesseract, ABBYY FineReader, and Transym", *Proceedings of the 12th International Symposium on Visual Computing*, 2016, 735–746.

[7] Chopra S., Hadsell R., LeCun Y., "Learning a similarity metric discriminatively, with application to face verification", *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, **1** (2005), 539–546. DOI: 10.1109/CVPR.2005.202.

[8] Hadsell R., Chopra S., LeCun Y., "Dimensionality reduction by learning an invariant mapping", *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, **2** (2006), 1735–1742. DOI: 10.1109/CVPR.2006.100.

[9] Ilyuhin S.A., Sheshkus A.V., Arlazarov V.L., "Recognition of images of Korean characters using embedded networks", *Proceedings of the Twelfth International Conference on Machine Vision (ICMV 2019)*, **11433** (2020), 1143311.

[10] Kondrashev I.V., Sheshkus A.V., Arlazarov V.V., "Distance-based online pairs generation method for metric networks training", *Proceedings of the Thirteenth International Conference on Machine Vision*, **11605** (2021), 1160508.

[11] Wang X., Hua Y., Kodirov E., Hu G., Robertson N.M., "Deep metric learning by online soft mining and class-aware attention", *Proceedings of the AAAI Conference on Artificial Intelligence*, **33** (2019), 5361–5368. DOI: 10.1609/aaai.v33i01.33015361.

[12] Wang F., Liu H., "Understanding the behaviour of contrastive loss", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, 2495–2504.

[13] Schroff F., Kalenichenko D., Philbin J., "Facenet: A unified embedding for face recognition and clustering", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, 815–823.

[14] Yuan Y., Chen W., Yang Y., Wang Z., "In defense of the triplet loss again: Learning robust person re-identification with fast approximated triplet loss and label distillation", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, 354–355.

[15] Dalwadi Bijal N., Shah Vatsal, "Deep Embedding Learning for Printed Word Spotting using Triplet CNNs and Transfer Learning", *Proceedings of the IEEE 5th International Conference on Soft Computing for Security Applications (ICSCSA)*, **73** (2025), 384–389.

[16] Chen W., Chen X., Zhang J., Huang K., "Beyond triplet loss: A deep quadruplet network for person re-identification", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, 403–412.

[17] Franken M., van Gemert J.C., "Automatic Egyptian hieroglyph recognition by retrieving images as texts", *Proceedings of the 21st ACM international conference on Multimedia*, 2013, 765–768. DOI: 10.1145/2502081.2502199.

[18] Kim Y.g., Cha E.y., "Learning of Large-Scale Korean Character Data through the Convolutional Neural Network", *Proceedings of the Korean Institute of Information and Commucation Sciences Conference*, 2016, 97–100.

[19] Bell S., Bala K., "Learning visual similarity for product design with convolutional neural networks", *ACM Trans. Graph. (TOG)*, **34** (2015), 1–10. DOI: 10.1145/2766959.

[20] Simo-Serra E., Trulls E., Ferraz L., Kokkinos I., Fua P., Moreno-Noguer F., "Discriminative learning of deep convolutional feature point descriptors", *Proceedings of the IEEE International Conference on Computer Vision*, 2015, 118–126.

[21] Xu A., Hsieh J.Y., Vundurthy B., Cohen E., Choset H., Li L., "Mathematical Justification of Hard Negative Mining via Isometric Approximation Theorem", *arXiv*, 2022.

[22] Xuan H., Stylianou A., Pless R., "Improved embeddings with easy positive triplet mining", *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, 2474–2482.

[23] Mokin A.K., Gayer A.V., Sheshkus A.V., Arlazarov V.L., "Auto-clustering pairs generation method for Siamese neural networks training", *Proceedings of the Fourteenth International Conference on Machine Vision (ICMV 2021)*, **12084** (2022), 369–376. DOI: 10.1117/12.2623139.

[24] Ham D.S., Lee D.R., Jung I.S., Oh I.S., "Construction of printed Hangul character database PHD08", *J. Korea Contents Assoc.*, **8** (2008), 33–40.

[25] Lake B., Salakhutdinov R., Gross J., Tenenbaum J., "One shot learning of simple visual concepts", *Proceedings of the Annual Meeting of the Cognitive Science Society*, **33** (2011).

[26] Lake B.M., Salakhutdinov R., Tenenbaum J.B., "Human-level concept learning through probabilistic program induction", *Science*, **350** (2015), 1332–1338. DOI: 10.1126/science.aab3050.

[27] Bergstra J., Desjardins G., Lamblin P., Bengio Y., "Quadratic polynomials learn better image features", *Tech. Rep.*, 2009.

[28] Lin G., Shen W., "Research on convolutional neural network based on improved Relu piecewise activation function", *Procedia Comput. Sci.*, **131** (2018), 977–984. DOI: 10.1016/j.procs.2018.04.239.

Received on February 9, 2026

Часть 3
Математические модели

Оценки доли последовательностей с небольшим отклонением от средних значений

В. В. Кочергин*

Рассматривается задача о величине ограничения в k-значном наборе на допустимое отклонение от среднего значения количества разрядов с каждым из k значений при заданной доле таких наборов. Предложен подход, позволяющий отследить на каждом этапе влияние разных параметров на окончательный ответ, а также даны оценки точности приближенного решения.

Ключевые слова: k-значный многомерный куб, формула Стирлинга, доля наборов.

1. Введение

Данная работа является результатом исследований автора, проведенных несколько лет назад по заказу одной из французских фармацевтических компаний и заключающихся в оценке доли k-значных наборов большой длины, в которых количество разрядов с каждым из k значений отличается от среднего значения не более чем на заданную величину.

После истечения срока запрета на обнародование полученных результатов были колебания по вопросу о целесообразности публикации данного материала. С одной стороны, результаты на качественном уровне теоретического значения, в общем, не имеют, так как могут быть достаточно стандартными способами получены применением центральной предельной теоремы к k-значному случайному набору с полиномиальным (или мультиномиальным) распределением (см., например, [1], [2], [3]). С другой стороны, французские заказчики, прекрасно понимая качественную картину, хотели иметь четкое представление о том, какие параметры и в какой степени влияют на точность оценок на каждом этапе. Та настойчивость, с которой заказчики неоднократно просили продолжать исследования именно в этом направлении, дает основания считать, что представленные оценки могут быть интересны при решении практических задач.

* *Кочергин Вадим Васильевич* — зав. каф. дискретной математики мех.-мат. ф-та МГУ, e-mail: vvkoch@yandex.ru, ORCID: 0000-0002-7798-9502.

Kochergin Vadim Vasil'evich — Head of Chair of Discrete Mathematics, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics.

Стоит также отметить, что в двузначном случае задача сводится к подсчету числа наборов в средних слоях булева куба, которая нередко встречается в литературе — см., например, [4], [5], [6], а здесь приводится для полноты картины. Аналогичных результатов для общего случая автору найти не удалось.

2. Постановка задачи

Обозначим множество $\{0, 1, \ldots, k-1\}$ через E_k. Для некоторого достаточно большого значения N будем рассматривать последовательности длины N с элементами из множества E_k. Обозначим множество всех таких последовательностей через E_k^N. Множество E_k^N можно также определить как множество всех слов (наборов) длины N над алфавитом E_k. Легко понять, что

$$|E_k^N| = k^N.$$

Для произвольного набора $\tilde{\alpha}$ из множества E_k^N и для каждого $i = 0, 1, \ldots, k-1$ обозначим через $N_i(\tilde{\alpha})$ число разрядов набора $\tilde{\alpha}$, равных i. Очевидно, что

$$N_0(\tilde{\alpha}) + N_1(\tilde{\alpha}) + \ldots + N_{k-1}(\tilde{\alpha}) = N.$$

Далее будем считать, что число N делится на k, т. е. $N = kn$ для некоторого натурального n.

Будем говорить, что *набор $\tilde{\alpha}$ из множества E_k^N удовлетворяет Δ-условию*, если для каждого значения i, $0 \le i \le k-1$, выполняются неравенства

$$n - \Delta \le N_i(\tilde{\alpha}) \le n + \Delta,$$

где $n = N/k$.

Множество всех наборов из множества E_k^N, удовлетворяющих Δ-условию, обозначим через $E_k^N[\Delta]$.

Введем функцию $\lambda = \lambda(N, k, \Delta)$ равенством

$$\lambda(N, k, \Delta) = \frac{|E_k^N[\Delta]|}{|E_k^N|}.$$

При фиксированных значениях N и k функция $\lambda(N, k, \Delta)$ при изменении целочисленной переменной Δ от 0 до N монотонно возрастает. Следовательно, для любого действительного значения λ из полуинтервала $(0; 1]$ можно корректно определить функцию $\Delta(N, k, \lambda)$, принимающую значения из множества $\{0, 1, \ldots, N\}$, равенством

$$\Delta(N, k, \lambda) = \max\{\nabla \mid \lambda(N, k, \nabla) \le \lambda\}.$$

Задача: по параметрам N, k и λ определить значение $\Delta(N,k,\lambda)$.

Ключевой момент для решения задачи — умение по параметрам N, k и Δ определять значение $\lambda(N,k,\Delta)$ или, что то же самое, хорошо оценивать величину $|E_k^N[\Delta]|$.

3. Случай $k=2$

В этом случае, учитывая равенство $N=2n$, имеем:

$$\lambda(N,2,\Delta)=\frac{|E_2^N[\Delta]|}{|E_2^N|}=\frac{\sum\limits_{t=-\Delta}^{\Delta}C_N^{N/2+t}}{2^N}=\frac{\sum\limits_{t=-\Delta}^{\Delta}C_{2n}^{n+t}}{2^{2n}}.$$

Используя формулу Стирлинга, оценим величину C_{2n}^{n+t} при $n\to\infty$ и $t\le n/2$:

$$\begin{aligned}
C_{2n}^{n+t}&=\frac{(2n)!}{(n+t)!(n-t)!}=\\
&=\frac{\sqrt{2\pi(2n)}\left(\frac{2n}{e}\right)^{2n}}{\sqrt{2\pi(n+t)}\left(\frac{n+t}{e}\right)^{n+t}\sqrt{2\pi(n-t)}\left(\frac{n-t}{e}\right)^{n-t}}(1+o(1))=\\
&=2^{2n}\sqrt{\frac{n}{\pi(n+t)(n-t)}}\,\frac{n^{2n}}{(n+t)^{n+t}(n-t)^{n-t}}(1+o(1))=\\
&=2^{2n}\sqrt{\frac{n}{\pi(n+t)(n-t)}}(1+o(1))\times\\
&\times\exp\left(2n\ln n-(n+t)\ln\left(n\left(1+\frac{t}{n}\right)\right)-(n-t)\ln\left(n\left(1-\frac{t}{n}\right)\right)\right)=\\
&=2^{2n}\sqrt{\frac{n}{\pi(n+t)(n-t)}}\,(1+o(1))\times\\
&\times\exp\left(-(n+t)\ln\left(1+\frac{t}{n}\right)-(n-t)\ln\left(1-\frac{t}{n}\right)\right).
\end{aligned}$$

Далее, при дополнительном условии $t=o(n)$ имеем:

$$\begin{aligned}
C_{2n}^{n+t}&=\frac{2^{2n}}{\sqrt{\pi n}}\,(1+o(1))\times\\
\times\exp&\left(-(n+t)\left(\frac{t}{n}-\frac{t^2}{2n^2}+O\left(\frac{t^3}{n^3}\right)\right)-(n-t)\left(-\frac{t}{n}-\frac{t^2}{2n^2}+O\left(\frac{t^3}{n^3}\right)\right)\right)=\\
&=\frac{2^{2n}}{\sqrt{\pi n}}\,(1+o(1))\times
\end{aligned}$$

$$\times \exp\left(-\frac{2t^2}{n}+\frac{t^2}{n}+O\left(\frac{t^3}{n^2}\right)\right)(1+o(1))=\frac{2^{2n}}{\sqrt{\pi n}}\exp\left(-\frac{t^2}{n}+O\left(\frac{t^3}{n^2}\right)\right).$$

Теперь, дополнительно считая, что выполняется условие $t = o\left(n^{2/3}\right)$, получаем:

$$C_{2n}^{n+t}=\frac{2^{2n}\,e^{-t^2/n}}{\sqrt{\pi n}}\,(1+o(1)).$$

Отметим, что величина $o(1)$ в последней формуле равномерна по t.

Возвращаясь к оценке величины $\lambda(N,2,\Delta)$, имеем:

$$\frac{\sum\limits_{t=-d}^{d}C_{2n}^{n+t}}{2^{2n}}=\frac{2\sum\limits_{t=0}^{d}C_{2n}^{n+t}-C_{2n}^{n}}{2^{2n}}=\frac{1}{\sqrt{\pi n}}\left(2\sum_{t=0}^{d}e^{-t^2/n}-1\right)(1+o(1)).$$

Функция $e^{-t^2/n}$ как функция переменной t монотонно убывает на участке $[0;+\infty)$. Поэтому

$$\sum_{t=1}^{d}e^{-t^2/n}<\int_0^d e^{-t^2/n}dt<\sum_{t=0}^{d-1}e^{-t^2/n}.$$

Следовательно,

$$\int_0^d e^{-t^2/n}dt<\sum_{t=0}^{d}e^{-t^2/n}<\int_0^d e^{-t^2/n}dt+1.$$

Поэтому,

$$\frac{\sum\limits_{t=-d}^{d}C_{2n}^{n+t}}{2^{2n}}=\frac{2}{\sqrt{\pi n}}\left(\int_0^d e^{-t^2/n}dt\right)(1+o(1)).$$

Сделав замену переменной в интеграле, получаем:

$$\int_0^d e^{-t^2/n}dt=\sqrt{n}\int_0^d e^{-(t/\sqrt{n})^2}d\left(\frac{t}{\sqrt{n}}\right)=\sqrt{n}\int_0^{d/\sqrt{n}}e^{-x^2}dx.$$

Таким образом, при $\Delta = o(N^{2/3})$ имеем:

$$\lambda(N,2,\Delta)=\frac{2}{\sqrt{\pi}}\int_0^{\frac{\Delta}{\sqrt{n}}}e^{-x^2}dx\;(1+o(1)),$$

где величина последнего интеграла ограничена сверху величиной гауссова интеграла (или интеграла Эйлера — Пуассона)

$$\int_0^{+\infty} e^{-x^2}\,dx,$$

численно равного $\sqrt{\pi}/2$.

Окончательно, в случае $k = 2$ имеем:

1) если выполняется условие

$$\frac{\Delta(N)}{\sqrt{N}} \to 0,$$

то $\lambda(N, 2, \Delta(N)) \to 0$;

2) если выполняется условие

$$\frac{\Delta(N)}{\sqrt{N}} \to \infty,$$

то $\lambda(N, 2, \Delta(N)) \to 1$;

3) если выполняется условие $\Delta(N) = a\sqrt{N}$, т. е. $\Delta(N) = a\sqrt{2n}$, то

$$\lambda(N, 2, \Delta(N)) = \frac{2}{\sqrt{\pi}}\,(1 + o(1)) \int_0^{a\sqrt{2}} e^{-x^2}\,dx.$$

4. Общий случай ($k \geq 2$)

Пусть k — некоторое фиксированное натуральное число не менее 2, $n \to \infty$, $N = kn$.

Тогда

$$\lambda(N, k, \Delta) = \frac{|E_k^N[\Delta]|}{|E_k^N|} = \frac{|E_k^{kn}[\Delta]|}{|E_k^{kn}|} = \frac{\sum P(n_1, n_2, \ldots, n_k)}{k^{kn}},$$

где $P(n_1, n_2, \ldots, n_k)$ — полиномиальный коэффициент, а суммирование ведется по всем наборам вида $(n_1, n_2, \ldots, n_k)$, состоящим из k натуральных чисел и удовлетворяющим условиям:

1) $n_1 + n_2 + \ldots + n_k = kn$;

2) $|n_i - n| \leq \Delta$ для всех i, $1 \leq i \leq k$.

Используя формулу Стирлинга, оценим при выполнении условия

$$n_1 + n_2 + \ldots + n_k \to \infty$$

величину полиномиального коэффициента $P(n_1, n_2, \ldots, n_k)$:

$$P(n_1, n_2, \ldots, n_k) = C^{n_k}_{n_1+n_2+\ldots+n_k} C^{n_{k-1}}_{n_1+n_2+\ldots+n_{k-1}} \ldots C^{n_2}_{n_1+n_2} =$$
$$= \frac{(n_1+n_2+\ldots+n_k)!}{n_1!n_2!\ldots n_k!} = \frac{(kn)!}{n_1!n_2!\ldots n_k!} =$$
$$= \frac{\sqrt{2\pi kn}\left(\frac{kn}{e}\right)^{kn}}{\sqrt{2\pi n_1}\left(\frac{n_1}{e}\right)^{n_1}\sqrt{2\pi n_2}\left(\frac{n_2}{e}\right)^{n_2}\ldots\sqrt{2\pi n_k}\left(\frac{n_k}{e}\right)^{n_k}}(1+o(1)) =$$
$$= \sqrt{\frac{kn}{(2\pi)^{k-1}n_1n_2\ldots n_k}}\,\frac{(kn)^{kn}}{n_1^{n_1}n_2^{n_2}\ldots n_k^{n_k}}(1+o(1)) =$$
$$= \sqrt{\frac{kn}{(2\pi)^{k-1}n_1n_2\ldots n_k}}\,(1+o(1))\times$$
$$\times \exp\left(kn\ln(kn) - n_1\ln n_1 - n_2\ln n_2 - \ldots - n_k\ln n_k\right).$$

Пусть дополнительно выполняется условие

$$\max(n_1, n_2, \ldots, n_k) - \min(n_1, n_2, \ldots, n_k) = o(n).$$

Тогда в силу соотношений

$$\min(n_1, n_2, \ldots, n_k) \le \frac{n_1+n_2+\ldots+n_k}{k} = n \le \max(n_1, n_2, \ldots, n_k)$$

для любого i, $1 \le i \le k$, выполняется условие

$$|n - n_i| = o(n).$$

Тогда справедливы следующие соотношения:

$$n_1n_2\ldots n_k = (n+n_1-n)(n+n_2-n)\ldots(n+n_k-n) =$$
$$= n^k\left(1+\frac{n_1-n}{n}\right)\left(1+\frac{n_2-n}{n}\right)\ldots\left(1+\frac{n_k-n}{n}\right) =$$
$$= n^k(1+o(1))(1+o(1))\ldots(1+o(1)) = n^k(1+o(1));$$

$$n_1\ln n_1 + n_2\ln n_2 + \ldots + n_k\ln n_k =$$
$$= n_1\ln\left(n\left(1+\frac{n_1-n}{n}\right)\right) + n_2\ln\left(n\left(1+\frac{n_2-n}{n}\right)\right) + \ldots$$
$$\ldots + n_k\ln\left(n\left(1+\frac{n_1-n}{n}\right)\right) =$$

$$= kn\ln n + n_1\ln\left(1+\frac{n_1-n}{n}\right) + n_2\ln\left(1+\frac{n_2-n}{n}\right) + \ldots$$
$$\ldots + n_k\ln\left(1+\frac{n_k-n}{n}\right).$$

Далее, для каждого i, $1 \le i \le k$, верны оценки

$$n_i\ln\left(1+\frac{n_i-n}{n}\right) = n_i\left(\frac{n_i-n}{n} - \frac{(n_i-n)^2}{2n^2} + O\left(\frac{(n_i-n)^3}{n^3}\right)\right) =$$
$$= (n+(n_i-n))\left(\frac{n_i-n}{n} - \frac{(n_i-n)^2}{2n^2} + O\left(\frac{(n_i-n)^3}{n^3}\right)\right) =$$
$$= (n_i-n) - \frac{(n_i-n)^2}{2n} + \frac{(n_i-n)^2}{n} - \frac{(n_i-n)^3}{2n^2} + O\left(\frac{(n_i-n)^3}{n^2}\right) =$$
$$= (n_i-n) + \frac{(n_i-n)^2}{2n} + O\left(\frac{(n_i-n)^3}{n^2}\right).$$

Таким образом,

$$n_1\ln n_1 + n_2\ln n_2 + \ldots + n_k\ln n_k =$$
$$= kn\ln n + (n_1-n) + \frac{(n_1-n)^2}{2n} + O\left(\frac{(n_1-n)^3}{n^2}\right) +$$
$$+ (n_2-n) + \frac{(n_2-n)^2}{2n} + O\left(\frac{(n_2-n)^3}{n^2}\right) + \ldots$$
$$\ldots + (n_k-n) + \frac{(n_k-n)^2}{2n} + O\left(\frac{(n_k-n)^3}{n^2}\right) =$$
$$= kn\ln n + \frac{(n_1-n)^2}{2n} + \frac{(n_2-n)^2}{2n} + \ldots + \frac{(n_k-n)^2}{2n} +$$
$$+ O\left(\frac{(\max(n_1,n_2,\ldots,n_k) - \min(n_1,n_2,\ldots,n_k))^3}{n^2}\right).$$

Далее будем считать, что выполняется условие

$$\max(n_1,n_2,\ldots,n_k) - \min(n_1,n_2,\ldots,n_k) = o\left(n^{2/3}\right).$$

Тогда, возвращаясь к оцениванию полиномиального коэффициента, имеем:

$$P(n_1,n_2,\ldots,n_k) = \sqrt{\frac{kn}{(2\pi)^{k-1}n^k(1+o(1))}} \times$$
$$\times \exp\left(kn\ln(kn) - kn\ln n - \frac{(n_1-n)^2}{2n} - \frac{(n_2-n)^2}{2n} - \ldots -\right.$$

$$-\frac{(n_k-n)^2}{2n}+o(1)\Bigg)=$$

$$=\sqrt{\frac{k}{(2\pi n)^{k-1}}}\,\exp(kn\ln k)\,(1+o(1))\times$$

$$\times\exp\left(-\frac{(n_1-n)^2}{2n}-\frac{(n_2-n)^2}{2n}-\ldots-\frac{(n_k-n)^2}{2n}\right)=$$

$$=\sqrt{\frac{k}{(2\pi n)^{k-1}}}\,k^{kn}\,\exp\left(-\frac{(n_1-n)^2}{2n}-\frac{(n_2-n)^2}{2n}-\ldots-\frac{(n_k-n)^2}{2n}\right)\cdot$$

$$\cdot(1+o(1))=$$

$$=\sqrt{\frac{k}{(2\pi n)^{k-1}}}\,k^{kn}\,e^{-((n_1-n)^2+(n_2-n)^2+\ldots+(n_k-n)^2)/(2n)}(1+o(1)).$$

Теперь нужно оценить величину

$$\sum\exp\left\{\frac{-((n_1-n)^2+(n_2-n)^2+\ldots+(n_k-n)^2)}{2n}\right\},$$

где сумма берется по всем наборам вида $(n_1,n_2,\ldots,n_k)$, состоящим из k натуральных чисел и удовлетворяющим условиям:

1) $n_1+n_2+\ldots+n_k=kn$;
2) $|n_i-n|\le\Delta$ для всех i, $1\le i\le k$.

Эту сумму можно переписать следующим образом:

$$\sum_{(t_1,\ldots,t_{k-1})\in\mathcal{D}}\exp\left\{\frac{-\left(t_1^2+t_2^2+\ldots+t_{k-1}^2+(-t_1-t_2-\ldots-t_{k-1})^2\right)}{2n}\right\},$$

где суммирование ведется по всем целочисленным наборам $(t_1,\ldots,t_{k-1})$, лежащим в области $\mathcal{D}$ пространства размерности $k-1$, определяемой следующим образом:

$$-\Delta\le t_i\le\Delta,\quad i=1,\ldots k-1;$$
$$-\Delta\le t_1+t_2+\ldots+t_{k-1}\le\Delta.$$

Последнюю сумму, в свою очередь, можно выразить через кратный интеграл

$$\sum_{(t_1,\ldots,t_{k-1})\in\mathcal{D}}\exp\left\{\frac{-\left(t_1^2+t_2^2+\ldots+t_{k-1}^2+(-t_1-t_2-\ldots-t_{k-1})^2\right)}{2n}\right\}=$$

$$= (1+o(1)) \iint\limits_{\mathcal{D}} \ldots \int e^{-\frac{t_1^2+t_2^2+\ldots+t_{k-1}^2+(-t_1-t_2-\ldots-t_{k-1})^2}{2n}}\, dt_1\, dt_2\, \ldots\, dt_{k-1}.$$

Для оценки последнего кратного интеграла путем сведения его к одномерному гауссову интегралу (или интегралу Эйлера — Пуассона) преобразуем выражение в показателе экспоненты.

Воспользуемся следующим равенством, которое проверяется непосредственно

$$\left(\frac{p+1}{2p}\right)\sum_{i=p}^{q} t_i^2 + \left(\frac{1}{p}\right)\sum_{p\le i<j\le q} t_i t_j =$$
$$= \left(\frac{p+1}{2p}\right)\left(t_p + \frac{1}{p+1}t_{p+1} + \ldots + \frac{1}{p+1}t_q\right)^2 +$$
$$+ \left(\frac{p+2}{2(p+1)}\right)\sum_{i=p+1}^{q} t_i^2 + \left(\frac{1}{p+1}\right)\sum_{p+1\le i<j\le q} t_i t_j.$$

Последовательно для $p = 1, 2, \ldots, k-2$ применяя установленное равенство (при $q = k-1$), получаем

$$\frac{1}{2}\left(t_1^2 + t_2^2 + \ldots + t_{k-1}^2 + (-t_1 - t_2 - \ldots - t_{k-1})^2\right) = \sum_{i=1}^{k-1} t_i^2 + \sum_{1\le i<j\le k-1} t_i t_j =$$
$$= \sum_{p=1}^{k-2}\left(\frac{p+1}{2p}\left(t_p + \frac{1}{p+1}t_{p+1} + \ldots + \frac{1}{p+1}t_{k-1}\right)^2\right) + \frac{k}{2(k-1)}t_{k-1}^2.$$

Оценим кратный интеграл по $(k-1)$-мерному параллелепипеду (этот параллелепипед может как содержаться в области $\mathcal{D}$, так и содержать область $\mathcal{D}$):

$$\int\limits_{-D}^{D}\int\limits_{-D}^{D}\ldots\int\limits_{-D}^{D} e^{-\frac{t_1^2+t_2^2+\ldots+t_{k-1}^2+(-t_1-t_2-\ldots-t_{k-1})^2}{2n}}\, dt_1\, dt_2\, \ldots\, dt_{k-1} =$$
$$= \int\limits_{-D}^{D}\int\limits_{-D}^{D}\ldots\int\limits_{-D}^{D} e^{-\left(\frac{1}{n}\right)\frac{t_1^2+t_2^2+\ldots+t_{k-1}^2+(-t_1-t_2-\ldots-t_{k-1})^2}{2}}\, dt_1\, dt_2\, \ldots\, dt_{k-1} =$$
$$= \int\limits_{-D}^{D}\int\limits_{-D}^{D}\ldots\int\limits_{-D}^{D} \exp\left\{-\left(\frac{1}{n}\right)\left(\sum_{p=1}^{k-2}\left(\frac{p+1}{2p}\left(t_p + \frac{1}{p+1}t_{p+1} + \ldots\right.\right.\right.\right.$$

$$\ldots + \frac{1}{p+1}t_{k-1}\Big)^2\Big) + \frac{k}{2(k-1)}t_{k-1}^2\Big)\Big\}\, dt_1\, dt_2\, \ldots\, dt_{k-1}.$$

В последнем интеграле сделаем замену переменных:

$$\begin{aligned}
x_1 &= t_1 + \frac{t_2}{2} + \ldots + \frac{t_{k-1}}{2},\\
x_2 &= t_2 + \frac{t_3}{3} + \ldots + \frac{t_{k-1}}{3},\\
&\ldots\\
x_p &= t_p + \frac{t_{p+1}}{p+1} + \ldots + \frac{t_{k-1}}{p+1},\\
&\ldots\\
x_{k-2} &= t_{k-2} + \frac{t_{k-1}}{k-1},\\
x_{k-1} &= t_{k-1}.
\end{aligned}$$

Якобиан этого преобразования равен 1, а следовательно и якобиан обратного преобразования равен 1. Поэтому

$$\int_{-D}^{D}\int_{-D}^{D}\ldots\int_{-D}^{D}\exp\left\{-\left(\frac{1}{n}\right)\left(\sum_{p=1}^{k-2}\left(\frac{p+1}{2p}\left(t_p + \frac{1}{p+1}t_{p+1} + \ldots\right.\right.\right.\right.$$
$$\left.\left.\left.\left. \ldots + \frac{1}{p+1}t_{k-1}\Big)^2\Big) + \frac{k}{2(k-1)}t_{k-1}^2\Big)\Big\}\, dt_1\, dt_2\, \ldots\, dt_{k-1} =\right.\right.$$
$$= \int_{-\frac{kD}{2}}^{\frac{kD}{2}}\int_{-\frac{kD}{3}}^{\frac{kD}{3}}\ldots\int_{-\frac{kD}{k-1}}^{\frac{kD}{k-1}}\int_{-D}^{D}\exp\left\{-\frac{1}{n}\left(\sum_{p=1}^{k-1}\frac{p+1}{2p}x_p^2\right)\right\}\, dx_1\, dx_2\, \ldots\, dx_{k-2}\, dx_{k-1}.$$

Сделаем еще одну замену:

$$y_p = \sqrt{\frac{p+1}{2pn}}x_p, \quad p = 1, 2, \ldots k-1.$$

Якобиан I обратного преобразования вычисляется следующим образом:

$$I = \prod_{p=1}^{k-1}\sqrt{\frac{2pn}{p+1}} = \sqrt{(2n)^{k-1}}\sqrt{\frac{1\cdot 2\cdot \ldots\cdot (k-2)(k-1)}{2\cdot 3\cdot \ldots\cdot (k-1)k}} = \sqrt{\frac{(2n)^{k-1}}{k}}.$$

Поэтому

$$\int\limits_{-\frac{kD}{2}}^{\frac{kD}{2}}\int\limits_{-\frac{kD}{3}}^{\frac{kD}{3}}\dots\int\limits_{-\frac{kD}{k-1}}^{\frac{kD}{k-1}}\int\limits_{-D}^{D} e^{-\left(\frac{1}{n}\right)\left(\sum_{p=1}^{k-1}\frac{p+1}{2p}x_p^2\right)}\,dx_1\,dx_2\,\dots\,dx_{k-2}\,dx_{k-1}=$$

$$=\sqrt{\frac{(2n)^{k-1}}{k}}\int\limits_{-\frac{kD}{2\sqrt{n}}}^{\frac{kD}{2\sqrt{n}}}\int\limits_{-\frac{kD}{\sqrt{12n}}}^{\frac{kD}{\sqrt{12n}}}\dots\int\limits_{-\frac{kD}{\sqrt{2(k-2)(k-1)n}}}^{\frac{kD}{\sqrt{2(k-2)(k-1)n}}}\int\limits_{-\frac{\sqrt{k}D}{\sqrt{2(k-1)n}}}^{\frac{\sqrt{k}D}{\sqrt{2(k-1)n}}}\exp\left\{-\left(y_1^2+y_2^2+\dots\right.\right.$$

$$\left.\left.\dots+\,y_{k-2}^2+y_{k-1}^2\right)\right\}dy_1\,dy_2\,\dots\,dy_{k-2}\,dy_{k-1}=$$

$$=\sqrt{\frac{(2n)^{k-1}}{k}}\left(\prod_{p=1}^{k-1}\int\limits_{-\frac{kD}{\sqrt{2p(p+1)n}}}^{\frac{kD}{\sqrt{2p(p+1)n}}}e^{-y_p^2}\,dy_p\right).$$

Таким образом, окончательно получаем такое равенство для исследуемого интеграла:

$$\int\limits_{-D}^{D}\int\limits_{-D}^{D}\dots\int\limits_{-D}^{D} e^{-\frac{t_1^2+t_2^2+\dots+t_{k-1}^2+(-t_1-t_2-\dots-t_{k-1})^2}{2n}}\,dt_1\,dt_2\,\dots\,dt_{k-1}=$$

$$=\sqrt{\frac{(2n)^{k-1}}{k}}\left(\prod_{p=1}^{k-1}\int\limits_{-\frac{kD}{\sqrt{2p(p+1)n}}}^{\frac{kD}{\sqrt{2p(p+1)n}}}e^{-y_p^2}\,dy_p\right).$$

Полученные оценки позволяют сделать следующие выводы.

Утверждение 1. *Пусть выполняется условие*

$$\frac{\Delta(N)}{N^{2/3}}\to 0.$$

Тогда

$$\lambda(N,k,\Delta(N))=\sqrt{\frac{k}{(2\pi n)^{k-1}}}\,(1+o(1))\times$$

$$\times\iint\limits_{\mathcal{D}}\dots\int e^{-\left(t_1^2+t_2^2+\dots+t_{k-1}^2+(-t_1-t_2-\dots-t_{k-1})^2\right)/(2n)}\,dt_1\,dt_2\,\dots\,dt_{k-1},$$

где область $\mathcal{D}$ пространства размерности $k-1$, по которой берется кратный интеграл, определяется следующим образом:

$$-\Delta(N) \le t_i \le \Delta(N), \quad i = 1, \dots k-1;$$
$$-\Delta(N) \le t_1 + t_2 + \dots + t_{k-1} \le \Delta(N).$$

Доказательство. В условиях утверждения 1 имеем

$$\lambda(N,k,\Delta(N)) = \frac{\sum\limits_{(n_1,\dots,n_k)\text{ удовл. }\Delta-\text{ усл.}} P(n_1,n_2,\dots,n_k)}{k^{kn}} =$$
$$= \sqrt{\frac{k}{(2\pi n)^{k-1}}}\,(1+o(1))\times$$
$$\times \sum_{(n_1,\dots,n_k)\text{ удовл. }\Delta-\text{ усл.}} e^{-\frac{((n_1-n)^2+(n_2-n)^2+\dots+(n_{k-1}-n)^2+(-(n_1-n)-\dots-(n_{k-1}-n))^2)}{(2n)}} =$$
$$= \sqrt{\frac{k}{(2\pi n)^{k-1}}}\,(1+o(1))\times$$
$$\times \iint\limits_{\mathcal{D}} \dots \int e^{-\left(t_1^2+t_2^2+\dots+t_{k-1}^2+(-t_1-t_2-\dots-t_{k-1})^2\right)/(2n)}\,dt_1\,dt_2\,\dots\,dt_{k-1}.$$

Утверждение 1 доказано. □

Утверждение 2. *Пусть выполняется условие*

$$\frac{\Delta(N)}{\sqrt{N}} \to 0.$$

Тогда

$$\lambda(N,k,\Delta(N)) \to 0.$$

Доказательство. В условиях утверждения 2 имеем

$$\lambda(N,k,\Delta(N)) = \frac{\sum\limits_{(n_1,\dots,n_k)\text{ удовл. }\Delta-\text{ усл.}} P(n_1,n_2,\dots,n_k)}{k^{kn}} =$$
$$= \frac{\sum\limits_{(n_1,\dots,n_k)\text{ удовл. }\Delta-\text{ усл.}} \sqrt{\frac{k}{(2\pi n)^{k-1}}}\,k^{kn}\,e^{-\frac{(n_1-n)^2+\dots+(n_k-n)^2}{2n}}(1+o(1))}{k^{kn}} =$$
$$= \sqrt{\frac{k}{(2\pi n)^{k-1}}}\,(1+o(1))\times$$

$$\times \sum_{(n_1,\ldots,n_k)\text{ удовл. }\Delta-\text{ усл.}} e^{-\frac{(n_1-n)^2+\ldots+(n_{k-1}-n)^2+(-(n_1-n)-\ldots-(n_{k-1}-n))^2}{2n}} \le$$

$$\le \sqrt{\frac{k}{(2\pi n)^{k-1}}}\,(1+o(1))\times$$

$$\times \int_{-\Delta}^{\Delta}\int_{-\Delta}^{\Delta}\ldots\int_{-\Delta}^{\Delta} e^{-\left(t_1^2+t_2^2+\ldots+t_{k-1}^2+(-t_1-t_2-\ldots-t_{k-1})^2\right)/(2n)}\,dt_1\,dt_2\,\ldots\,dt_{k-1} =$$

$$= \sqrt{\frac{k}{(2\pi n)^{k-1}}}\sqrt{\frac{(2n)^{k-1}}{k}}\left(\prod_{p=1}^{k-1}\int_{-\frac{k\Delta}{\sqrt{2p(p+1)n}}}^{\frac{k\Delta}{\sqrt{2p(p+1)n}}} e^{-y_p^2}\,dy_p\right)(1+o(1)) \le$$

$$\le \frac{1}{\sqrt{\pi^{k-1}}}\left(\prod_{p=1}^{k-1}\frac{2k\Delta}{\sqrt{2p(p+1)n}}\right)(1+o(1)) = o(1).$$

Утверждение 2 доказано. □

Утверждение 3. *Пусть выполняется условие*

$$\frac{\sqrt{N}}{\Delta(N)} \to 0.$$

Тогда

$$\lambda(N,k,\Delta(N)) \to 1.$$

Доказательство. Так как условие $\Delta_1 \ge \Delta_2$ влечет за собой неравенство

$$\lambda(N,k,\Delta_1) \ge \lambda(N,k,\Delta_2),$$

то достаточно доказать утверждение в случае, когда выполняется условие

$$\Delta(N) = o\left(N^{2/3}\right).$$

Тогда, используя равенство

$$\int_{-\infty}^{\infty} e^{-x^2}\,dx = \sqrt{\pi}$$

для значения гауссового интеграла, имеем

$$\lambda(N,k,\Delta(N)) =$$

$$= \sqrt{\frac{k}{(2\pi n)^{k-1}}}\,(1+o(1))\times$$

$$\times \sum_{(n_1,\ldots,n_k)\text{ удовл. }\Delta-\text{ усл.}} e^{-\frac{(n_1-n)^2+(n_2-n)^2+\ldots+(n_{k-1}-n)^2+(-(n_1-n)-\ldots-(n_{k-1}-n))^2}{2n}} \geq$$

$$\geq \sqrt{\frac{k}{(2\pi n)^{k-1}}}\,(1+o(1))\times$$

$$\times \int\limits_{-\frac{\Delta}{k}}^{\frac{\Delta}{k}} \int\limits_{-\frac{\Delta}{k}}^{\frac{\Delta}{k}} \ldots \int\limits_{-\frac{\Delta}{k}}^{\frac{\Delta}{k}} e^{-\frac{t_1^2+t_2^2+\ldots+t_{k-1}^2+(-t_1-t_2-\ldots-t_{k-1})^2}{2n}}\,dt_1\,dt_2\,\ldots\,dt_{k-1} =$$

$$= \sqrt{\frac{k}{(2\pi n)^{k-1}}}\,\sqrt{\frac{(2n)^{k-1}}{k}} \left(\prod_{p=1}^{k-1} \int\limits_{-\frac{k\Delta}{\sqrt{2p(p+1)n}}}^{\frac{k\Delta}{\sqrt{2p(p+1)n}}} e^{-y_p^2}\,dy_p \right) (1+o(1)) =$$

$$= \frac{1}{\sqrt{\pi^{k-1}}} \left(\prod_{p=1}^{k-1} \left(\int\limits_{-\infty}^{\infty} e^{-y_p^2}\,dy_p + o(1) \right) \right) (1+o(1)) =$$

$$= \frac{\left(\sqrt{\pi}\right)^{k-1}}{\sqrt{\pi^{k-1}}}(1+o(1)) = 1+o(1).$$

Утверждение 3 доказано. □

5. Грубая оценка погрешности вычислений, общий случай ($k \geq 2$)

Ко всем условиям, в рамках которых велись рассуждения в разделе 4, добавим условие

$$\Delta(N) = a\sqrt{n}.$$

Оценим погрешность вычисления значения $\lambda(N,k,\Delta(N))$ по формуле из утверждения 1, т. е. оценим величину остаточного члена вида $o(1)$ из этой формулы. Для этого асимптотические неравенства из выкладок раздела 3, использующие o-символику, заменим на обычные неравенства. При этом придется отдельно выписывать оценки снизу и сверху.

Формулу Стирлинга будем использовать в следующем виде

$$n! = \sqrt{2\pi n}\left(\frac{n}{e}\right)^n \exp\frac{\alpha(n)}{12n},$$

где при всех натуральных n выполняется неравенство $0 < \alpha(n) < 1$.

Используя указанную форму формулы Стирлинга, получаем такую оценку полиномиального коэффициента $P(n_1, n_2, \ldots, n_k)$:

$$P(n_1, n_2, \ldots, n_k) = \frac{(kn)!}{n_1! n_2! \ldots n_k!} =$$

$$= \frac{\sqrt{2\pi kn}\left(\frac{kn}{e}\right)^{kn} e^{\frac{\alpha(kn)}{12kn}}}{\sqrt{2\pi n_1}\left(\frac{n_1}{e}\right)^{n_1} e^{\frac{\alpha(n_1)}{12n_1}} \sqrt{2\pi n_2}\left(\frac{n_2}{e}\right)^{n_2} e^{\frac{\alpha(n_2)}{12n_2}} \ldots \sqrt{2\pi n_k}\left(\frac{n_k}{e}\right)^{n_k} e^{\frac{\alpha(n_k)}{12n_k}}} =$$

$$= \sqrt{\frac{kn}{(2\pi k)^{k-1} n^k}} \sqrt{\frac{n^k}{n_1 n_2 \ldots n_k}} \frac{(kn)^{kn}}{n_1^{n_1} n_2^{n_2} \ldots n_k^{n_k}} \times$$

$$\times \exp\left\{\frac{\alpha(kn)}{12kn} - \frac{\alpha(n_1)}{12n_1} - \frac{\alpha(n_2)}{12n_2} - \ldots - \frac{\alpha(n_k)}{12n_k}\right\} =$$

$$= \sqrt{\frac{k}{(2\pi kn)^{k-1}}} \sqrt{\frac{n^k}{n_1 n_2 \ldots n_k}} \times$$

$$\times \exp\left(kn \ln(kn) - n_1 \ln n_1 - n_2 \ln n_2 - \ldots - n_k \ln n_k\right) \times$$

$$\times \exp\left\{\frac{\alpha(kn)}{12kn} - \frac{\alpha(n_1)}{12n_1} - \frac{\alpha(n_2)}{12n_2} - \ldots - \frac{\alpha(n_k)}{12n_k}\right\}.$$

Отдельно оценим второй, третий и четвертый множители.

Второй множитель. В силу равенств

$$n_1 n_2 \ldots n_k = (n + n_1 - n)(n + n_2 - n) \ldots (n - n_k - n) =$$

$$= n^k \left(1 + \frac{n_1 - n}{n}\right)\left(1 + \frac{n_2 - n}{n}\right) \ldots \left(1 + \frac{n_k - n}{n}\right)$$

и учитывая, что для всех i, $1 \le i \le k$, выполняются соотношения $|n_i - n| \le \Delta = a\sqrt{n}$, имеем:

$$n^k \left(1 - \frac{a\sqrt{n}}{n}\right)^k \le n_1 n_2 \ldots n_k \le n^k \left(1 + \frac{a\sqrt{n}}{n}\right)^k.$$

Поэтому

$$\sqrt{\frac{n^k}{n_1 n_2 \ldots n_k}} \le \left(1 - \frac{a\sqrt{n}}{n}\right)^{-\frac{k}{2}} = \exp\left\{-\frac{k}{2} \ln\left(1 - \frac{a\sqrt{n}}{n}\right)\right\} \le$$

$$\le \exp\left\{-\frac{k}{2}\left(-\frac{a\sqrt{n}}{n} + \frac{(a\sqrt{n})^2}{2n^2}\right)\right\} = \exp\left\{\frac{ak}{2\sqrt{n}} - \frac{a^2 k}{4n}\right\};$$

$$\sqrt{\frac{n^k}{n_1 n_2 \dots n_k}} \geq \left(1 + \frac{a\sqrt{n}}{n}\right)^{-\frac{k}{2}} =$$

$$= \exp\left\{-\frac{k}{2}\ln\left(1 + \frac{a\sqrt{n}}{n}\right)\right\} \geq \exp\left\{-\frac{ak}{2\sqrt{n}}\right\};$$

Третий множитель. Сначала в отрицательных слагаемых показателя экспоненты этого множителя выделим слагаемое $-kn\ln n$:

$$n_1\ln n_1 + n_2\ln n_2 + \dots + n_k\ln n_k =$$
$$= n_1\ln\left(n\left(1 + \frac{n_1 - n}{n}\right)\right) + n_2\ln\left(n\left(1 + \frac{n_2 - n}{n}\right)\right) + \dots$$
$$\dots + n_k\ln\left(n\left(1 + \frac{n_1 - n}{n}\right)\right) =$$
$$= kn\ln n + n_1\ln\left(1 + \frac{n_1 - n}{n}\right) + n_2\ln\left(1 + \frac{n_2 - n}{n}\right) + \dots$$
$$\dots + n_k\ln\left(1 + \frac{n_1 - n}{n}\right).$$

Далее, для каждого i, $1 \leq i \leq k$, верны оценки

$$n_i\ln\left(1 + \frac{n_i - n}{n}\right) \leq n_i\left(\frac{n_i - n}{n} - \frac{(n_i - n)^2}{2n^2} + \frac{(n_i - n)^3}{3n^3}\right) =$$
$$= (n + (n_i - n))\left(\frac{n_i - n}{n} - \frac{(n_i - n)^2}{2n^2} + \frac{(n_i - n)^3}{3n^3}\right) =$$
$$= (n_i - n) + \frac{(n_i - n)^2}{n} - \frac{(n_i - n)^2}{2n} - \frac{(n_i - n)^3}{2n^2} + \frac{(n_i - n)^3}{3n^2} + \frac{(n_i - n)^4}{3n^3} =$$
$$= (n_i - n) + \frac{(n_i - n)^2}{2n} + \frac{(n_i - n)^3}{n^2}\left(-\frac{1}{6} + \frac{n_i - n}{3n}\right) \leq$$
$$\leq (n_i - n) + \frac{(n_i - n)^2}{2n} - \frac{(n_i - n)^3}{7n^2};$$

$$n_i\ln\left(1 + \frac{n_i - n}{n}\right) \geq n_i\left(\frac{n_i - n}{n} - \frac{(n_i - n)^2}{2n^2}\right) =$$
$$= (n + (n_i - n))\left(\frac{n_i - n}{n} - \frac{(n_i - n)^2}{2n^2}\right) =$$
$$= (n_i - n) + \frac{(n_i - n)^2}{n} - \frac{(n_i - n)^2}{2n} - \frac{(n_i - n)^3}{2n^2} =$$
$$= (n_i - n) + \frac{(n_i - n)^2}{2n} - \frac{(n_i - n)^3}{2n^2}.$$

Таким образом,

$$n_1\ln n_1+n_2\ln n_2+\ldots+n_k\ln n_k\le$$
$$\le kn\ln n+(n_1-n)+\frac{(n_1-n)^2}{2n}-\frac{(n_1-n)^3}{7n^2}+$$
$$+(n_2-n)+\frac{(n_2-n)^2}{2n}-\frac{(n_2-n)^3}{7n^2}+\ldots+(n_k-n)+\frac{(n_k-n)^2}{2n}-\frac{(n_k-n)^3}{7n^2}=$$
$$=kn\ln n+\frac{(n_1-n)^2+(n_2-n)^2+\ldots+(n_k-n)^2}{2n}-$$
$$-\frac{(n_1-n)^3+(n_2-n)^3+\ldots+(n_k-n)^3}{7n^2};$$

$$n_1\ln n_1+n_2\ln n_2+\ldots+n_k\ln n_k\ge$$
$$\ge kn\ln n+(n_1-n)+\frac{(n_1-n)^2}{2n}-\frac{(n_1-n)^3}{2n^2}+$$
$$+(n_2-n)+\frac{(n_2-n)^2}{2n}-\frac{(n_2-n)^3}{2n^2}+\ldots+(n_k-n)+\frac{(n_k-n)^2}{2n}-\frac{(n_k-n)^3}{2n^2}=$$
$$=kn\ln n+\frac{(n_1-n)^2+(n_2-n)^2+\ldots+(n_k-n)^2}{2n}-$$
$$-\frac{(n_1-n)^3+(n_2-n)^3+\ldots+(n_k-n)^3}{2n^2}.$$

Следовательно,

$$\exp\left(kn\ln(kn)-n_1\ln n_1-n_2\ln n_2-\ldots-n_k\ln n_k\right)\le$$
$$\le k^{kn}\exp\left\{-\frac{(n_1-n)^2+(n_2-n)^2+\ldots+(n_k-n)^2}{2n}\right\}\times$$
$$\times\exp\left\{\frac{(n_1-n)^3+(n_2-n)^3+\ldots+(n_k-n)^3}{2n^2}\right\}\le$$
$$\le k^{kn}\exp\left\{-\frac{(n_1-n)^2+(n_2-n)^2+\ldots+(n_k-n)^2}{2n}\right\}\exp\left\{\frac{ka^3}{2\sqrt{n}}\right\};$$

$$\exp\left(kn\ln(kn)-n_1\ln n_1-n_2\ln n_2-\ldots-n_k\ln n_k\right)\ge$$
$$\ge k^{kn}\exp\left\{-\frac{(n_1-n)^2+(n_2-n)^2+\ldots+(n_k-n)^2}{2n}\right\}\times$$
$$\times\exp\left\{\frac{(n_1-n)^3+(n_2-n)^3+\ldots+(n_k-n)^3}{7n^2}\right\}\ge$$

$$\geq k^{kn} \exp\left\{-\frac{(n_1-n)^2+(n_2-n)^2+\ldots+(n_k-n)^2}{2n}\right\} \exp\left\{\frac{ka^3}{7\sqrt{n}}\right\}.$$

Четвертый множитель. Справедливы следующие оценки:

$$\exp\left\{\frac{\alpha(kn)}{12kn}-\frac{\alpha(n_1)}{12n_1}-\frac{\alpha(n_2)}{12n_2}-\ldots-\frac{\alpha(n_k)}{12n_k}\right\} \leq \exp\left\{\frac{1}{12kn}\right\};$$

$$\exp\left\{\frac{\alpha(kn)}{12kn}-\frac{\alpha(n_1)}{12n_1}-\frac{\alpha(n_2)}{12n_2}-\ldots-\frac{\alpha(n_k)}{12n_k}\right\} \geq$$

$$\geq \exp\left\{-\frac{k}{12(n-a\sqrt{n})}\right\} \geq \exp\left\{-\frac{k}{12n}\left(1+\frac{2a}{\sqrt{n}}\right)\right\}.$$

Возвращаясь к оцениванию полиномиального коэффициента, применяем полученные оценки множителей:

$$P(n_1,n_2,\ldots,n_k) =$$

$$= \sqrt{\frac{k}{(2\pi kn)^{k-1}}} \sqrt{\frac{n^k}{n_1n_2\ldots n_k}}\, e^{kn\ln(kn)-n_1\ln n_1-n_2\ln n_2-\ldots-n_k\ln n_k} \times$$

$$\times \exp\left\{\frac{\alpha(kn)}{12kn}-\frac{\alpha(n_1)}{12n_1}-\frac{\alpha(n_2)}{12n_2}-\ldots-\frac{\alpha(n_k)}{12n_k}\right\} \leq$$

$$\leq \sqrt{\frac{k}{(2\pi kn)^{k-1}}}\, k^{kn} \exp\left\{\frac{ak}{2\sqrt{n}}-\frac{a^2k}{4n}\right\} \exp\left\{\frac{ka^3}{2\sqrt{n}}\right\} \exp\left\{\frac{1}{12kn}\right\} \times$$

$$\times \exp\left\{-\frac{(n_1-n)^2+(n_2-n)^2+\ldots+(n_k-n)^2}{2n}\right\} \leq$$

$$\leq \sqrt{\frac{k}{(2\pi kn)^{k-1}}}\, k^{kn} \exp\left\{-\frac{(n_1-n)^2+(n_2-n)^2+\ldots+(n_k-n)^2}{2n}\right\} \times$$

$$\times \exp\left\{\frac{ka(1+a^2)+\varepsilon}{2\sqrt{n}}\right\};$$

$$P(n_1,n_2,\ldots,n_k) =$$

$$= \sqrt{\frac{k}{(2\pi kn)^{k-1}}} \sqrt{\frac{n^k}{n_1n_2\ldots n_k}}\, e^{kn\ln(kn)-n_1\ln n_1-n_2\ln n_2-\ldots-n_k\ln n_k} \times$$

$$\times \exp\left\{\frac{\alpha(kn)}{12kn}-\frac{\alpha(n_1)}{12n_1}-\frac{\alpha(n_2)}{12n_2}-\ldots-\frac{\alpha(n_k)}{12n_k}\right\} \geq$$

$$\geq \sqrt{\frac{k}{(2\pi kn)^{k-1}}}\ k^{kn} \exp\left\{-\frac{ak}{2\sqrt{n}}\right\}\ \exp\left\{\frac{ka^3}{7\sqrt{n}}\right\}\ \exp\left\{-\frac{k}{n}\left(1+\frac{2a}{\sqrt{n}}\right)\right\}\ \times$$

$$\times \exp\left\{-\frac{(n_1-n)^2+(n_2-n)^2+\ldots+(n_k-n)^2}{2n}\right\} \geq$$

$$\geq \sqrt{\frac{k}{(2\pi kn)^{k-1}}}\ k^{kn} \exp\left\{-\frac{(n_1-n)^2+(n_2-n)^2+\ldots+(n_k-n)^2}{2n}\right\}\ \times$$

$$\times\ \exp\left\{-\frac{ka(7-2a^2)+\varepsilon}{14\sqrt{n}}\right\}.$$

Обозначим через $\mathcal{D}$ область пространства размерности $k-1$, определяемой следующим образом:

$$-\Delta \leq t_i \leq \Delta, \quad i = 1, \ldots k-1;$$
$$-\Delta \leq t_1 + t_2 + \ldots + t_{k-1} \leq \Delta.$$

При суммированиях нас будут интересовать точки $(t_1, t_2, \ldots, t_{k-1})$ области $\mathcal{D}$ с целочисленными координатами.

Так как

$$\lambda(N,k,\Delta) = \frac{1}{k^{kn}}\left(\sum_{(t_1,\ldots,t_{k-1})\in\mathcal{D}} P(t_1,t_2,\ldots,t_{k-1},-t_1-t_2-\ldots-t_{k-1})\right),$$

то справедливы неравенства

$$\lambda(N,k,\Delta) \leq \sqrt{\frac{k}{(2\pi kn)^{k-1}}}\ \exp\left\{\frac{ka(1+a^2)+\varepsilon}{2\sqrt{n}}\right\}\ \times$$

$$\times \sum_{(t_1,\ldots,t_{k-1})\in\mathcal{D}} \exp\left\{\frac{-\left(t_1^2+t_2^2+\ldots+t_{k-1}^2+(-t_1-t_2-\ldots-t_{k-1})^2\right)}{2n}\right\};$$

$$\lambda(N,k,\Delta) \geq \sqrt{\frac{k}{(2\pi kn)^{k-1}}}\ \exp\left\{-\frac{ka(7-2a^2)+\varepsilon}{14\sqrt{n}}\right\}\ \times$$

$$\times \sum_{(t_1,\ldots,t_{k-1})\in\mathcal{D}} \exp\left\{\frac{-\left(t_1^2+t_2^2+\ldots+t_{k-1}^2+(-t_1-t_2-\ldots-t_{k-1})^2\right)}{2n}\right\}.$$

Оценим величину

$$\sum_{(t_1,\ldots,t_{k-1})\in\mathcal{D}} \exp\left\{\frac{-\left(t_1^2+t_2^2+\ldots+t_{k-1}^2+(-t_1-t_2-\ldots-t_{k-1})^2\right)}{2n}\right\}$$

через кратный интеграл

$$\iint\limits_{\mathcal{D}} \dots \int e^{-\left(t_1^2+t_2^2+\ldots+t_{k-1}^2+(-t_1-t_2-\ldots-t_{k-1})^2\right)/(2n)}\, dt_1\, dt_2\, \dots\, dt_{k-1}.$$

Обозначим через $K(t_1, t_2, \dots, t_{k-1})$ единичный куб, задаваемый в $(k-1)$-мерном пространстве как множество точек $(x_1, x_2, \dots, x_{k-1})$, удовлетворяющих неравенствам $t_i \le x_i \le t_i+1$, $i = 1, 2, \dots, k-1$. Кроме того, для произвольного компакта K $(k-1)$-мерного пространства определим функционал $\mathrm{Var}(K)$ равенством

$$\mathrm{Var}(K) = \max_{(x_1,\dots,x_{k-1})\in K} e^{-\left(x_1^2+\ldots+x_{k-1}^2+(-x_1-\ldots-x_{k-1})^2\right)/(2n)} - \\ - \min_{(x_1,\dots,x_{k-1})\in K} e^{-\left(x_1^2+\ldots+x_{k-1}^2+(-x_1-\ldots-x_{k-1})^2\right)/(2n)}.$$

Заметим, что если $(t_1, t_2, \dots, t_{k-1}) \in \mathcal{D}$, то справедливо неравенство

$$\mathrm{Var}(K(t_1, t_2, \dots, t_{k-1})) \le 1 - \exp\left\{-\frac{2(k-1)a}{\sqrt{n}}\right\} \le \frac{2(k-1)a}{\sqrt{n}}.$$

Далее, положим

$$\mathcal{D}_1 = \bigcup_{(t_1,\dots,t_{k-1})\in\mathcal{D}} K(t_1, \dots, t_{k-1});$$

$$\mathcal{D}_2 = \bigcup_{(t_1,\dots,t_{k-1}):\ K(t_1,\dots,t_{k-1})\subset\mathcal{D}} K(t_1, \dots, t_{k-1}).$$

Отметим, что выполняются включения

$$\mathcal{D}_2 \subset \mathcal{D}, \quad \mathcal{D}_2 \subset \mathcal{D}_1.$$

Оценим объемы некоторых областей (объем области $\mathcal{D}$ будем обозначать через $|\mathcal{D}|$). При оценке объема области $\mathcal{D}_1 \setminus \mathcal{D}$ воспользуемся тем соображением, что этот объем не более произведения площади поверхности области $\mathcal{D}$ на единицу, так как область $\mathcal{D}_1 \setminus \mathcal{D}$ находится в слое толщины 1 сразу за границей области $\mathcal{D}$). В свою очередь, площади поверхности области $\mathcal{D}$ не превосходит площади поверхности $(k-1)$-мерного куба с ребром $2a\sqrt{n}$. Аналогично будем оценивать и объем области $\mathcal{D} \setminus \mathcal{D}_2$. Таким образом,

$$|\mathcal{D}| = (2a\sqrt{n})^{k-1} - 2\frac{(2a\sqrt{n})^{k-1}}{(k-1)!} = (2a\sqrt{n})^{k-1}\left(1 - \frac{2}{(k-1)!}\right);$$

$$|\mathcal{D}_1 \setminus \mathcal{D}| \le 2(k-1)(2a\sqrt{n})^{k-2};$$

$$|\mathcal{D}_1| = |\mathcal{D}| + |\mathcal{D}_1 \setminus \mathcal{D}| \le (2a\sqrt{n})^{k-1}\left(1 - \frac{2}{(k-1)!} + \frac{k-1}{a\sqrt{n}}\right) < \\ < (2a\sqrt{n})^{k-1}\left(1 + \frac{k-1}{a\sqrt{n}}\right);$$

$$|\mathcal{D} \setminus \mathcal{D}_2| \le 2(k-1)(2a\sqrt{n})^{k-2}.$$

Тогда, вводя обозначение

$$E_n(t_1, t_2, \ldots, t_{k-1}) = \exp\left\{\frac{t_1^2 + t_2^2 + \ldots + t_{k-1}^2 + (-t_1 - t_2 - \ldots - t_{k-1})^2}{2n}\right\},$$

получаем:

$$\sum_{(t_1,\ldots,t_{k-1})\in\mathcal{D}} \exp\left\{\frac{-\left(t_1^2 + t_2^2 + \ldots + t_{k-1}^2 + (-t_1 - t_2 - \ldots - t_{k-1})^2\right)}{2n}\right\} \le$$

$$\le \iint\limits_{\mathcal{D}_1} \ldots \int \left(E_n(t_1, t_2, \ldots, t_{k-1}) + \frac{2(k-1)a}{\sqrt{n}}\right) dt_1\, dt_2\, \ldots\, dt_{k-1} \le$$

$$\le \iint\limits_{\mathcal{D}_1} \ldots \int E_n(t_1, t_2, \ldots, t_{k-1})\, dt_1\, dt_2\, \ldots\, dt_{k-1} + \frac{2(k-1)a}{\sqrt{n}}|\mathcal{D}_1| \le$$

$$\le \iint\limits_{\mathcal{D}} \ldots \int E_n(t_1, t_2, \ldots, t_{k-1})\, dt_1\, dt_2\, \ldots\, dt_{k-1} +$$

$$+ \int\int\limits_{\mathcal{D}_1\setminus\mathcal{D}} \ldots \int E_n(t_1, t_2, \ldots, t_{k-1})\, dt_1\, dt_2\, \ldots\, dt_{k-1} + \frac{2(k-1)a}{\sqrt{n}}|\mathcal{D}_1| \le$$

$$\le \iint\limits_{\mathcal{D}} \ldots \int E_n(t_1, t_2, \ldots, t_{k-1})\, dt_1\, dt_2\, \ldots\, dt_{k-1} +$$

$$+ \exp\left\{\frac{-(k-1)(a\sqrt{n}/(k-1))^2 - (a\sqrt{n})^2}{2n}\right\} |\mathcal{D}_1 \setminus \mathcal{D}| + \frac{2(k-1)a}{\sqrt{n}}|\mathcal{D}_1| \le$$

$$\le \iint\limits_{\mathcal{D}} \ldots \int E_n(t_1, t_2, \ldots, t_{k-1})\, dt_1\, dt_2\, \ldots\, dt_{k-1} +$$

$$+\exp\left\{\frac{-ka^2}{2(k-1)}\right\} 2(k-1)(2a\sqrt{n})^{k-2} + \frac{2(k-1)a}{\sqrt{n}}(2a\sqrt{n})^{k-1}\left(1 + \frac{k-1}{a\sqrt{n}}\right).$$

Таким образом, при $\Delta = a\sqrt{n}$ имеем такую верхнюю оценку:

$$\lambda(N,k,\Delta) \le \sqrt{\frac{k}{(2\pi kn)^{k-1}}}\ \exp\left\{\frac{ka(1+a^2)+\varepsilon}{2\sqrt{n}}\right\}\ \times$$
$$\times\ \left(\iint\limits_{\mathcal{D}}\dots\int E_n(t_1,t_2,\dots,t_{k-1})\,dt_1\,dt_2\,\dots\,dt_{k-1}\ +\right.$$
$$+\exp\left\{\frac{-ka^2}{2(k-1)}\right\}\ 2^{k-1}a^{k-2}(k-1)(\sqrt{n})^{k-2}+$$
$$\left.+\,(2a)^k(k-1)(\sqrt{n})^{k-2}\left(1+\frac{k-1}{a\sqrt{n}}\right)\right).$$

Аналогично оцениваем снизу

$$\sum_{(t_1,\dots,t_{k-1})\in\mathcal{D}}\exp\left\{\frac{-\left(t_1^2+t_2^2+\ldots+t_{k-1}^2+(-t_1-t_2-\ldots-t_{k-1})^2\right)}{2n}\right\}\ge$$
$$\ge\iint\limits_{\mathcal{D}_2}\dots\int\left(E_n(t_1,t_2,\dots,t_{k-1})-\frac{2(k-1)a}{\sqrt{n}}\right)\,dt_1\,dt_2\,\dots\,dt_{k-1}\ge$$
$$\ge\iint\limits_{\mathcal{D}_2}\dots\int E_n(t_1,t_2,\dots,t_{k-1})\,dt_1\,dt_2\,\dots\,dt_{k-1}-\frac{2(k-1)a}{\sqrt{n}}|\mathcal{D}_2|\ge$$
$$\ge\iint\limits_{\mathcal{D}}\dots\int E_n(t_1,t_2,\dots,t_{k-1})\,dt_1\,dt_2\,\dots\,dt_{k-1}-$$
$$-\int\int\limits_{\mathcal{D}\setminus\mathcal{D}_2}\dots\int E_n(t_1,t_2,\dots,t_{k-1})\,dt_1\,dt_2\,\dots\,dt_{k-1}-\frac{2(k-1)a}{\sqrt{n}}|\mathcal{D}_2|\ge$$
$$\ge\iint\limits_{\mathcal{D}}\dots\int E_n(t_1,t_2,\dots,t_{k-1})\,dt_1\,dt_2\,\dots\,dt_{k-1}\ +$$
$$+\exp\left\{\frac{-(k-1)(a\sqrt{n}/(k-1))^2-(a\sqrt{n})^2}{2n}\right\}\ |\mathcal{D}\setminus\mathcal{D}_2|-\frac{2(k-1)a}{\sqrt{n}}|\mathcal{D}|\ge$$
$$\ge\iint\limits_{\mathcal{D}}\dots\int E_n(t_1,t_2,\dots,t_{k-1})\,dt_1\,dt_2\,\dots\,dt_{k-1}\ -$$
$$-\exp\left\{\frac{-ka^2}{2(k-1)}\right\}\ 2(k-1)(2a\sqrt{n})^{k-2}-$$
$$-\frac{2(k-1)a}{\sqrt{n}}(2a\sqrt{n})^{k-1}\left(1-\frac{2}{(k-1)!}\right).$$

При $\Delta = a\sqrt{n}$ получаем такую нижнюю оценку:

$$\lambda(N,k,\Delta) \geq \sqrt{\frac{k}{(2\pi kn)^{k-1}}}\ \exp\left\{-\frac{ka(7+2a^2)+\varepsilon}{14\sqrt{n}}\right\}\ \times$$
$$\times\ \left(\idotsint\limits_{\mathcal{D}} E_n(t_1,t_2,\ldots,t_{k-1})\,dt_1\,dt_2\,\ldots\,dt_{k-1} -\right.$$
$$-\exp\left\{\frac{-ka^2}{2(k-1)}\right\}\ 2^{k-1}a^{k-2}(k-1)(\sqrt{n})^{k-2} -$$
$$\left.-(2a)^k(k-1)(\sqrt{n})^{k-2}\left(1-\frac{2}{(k-1)!}\right)\right).$$

Теперь, положив

$$I = I(N,k,\Delta) = \sqrt{\frac{k}{(2\pi kn)^{k-1}}}\ \times$$
$$\times\ \idotsint\limits_{\mathcal{D}} E_n(t_1,t_2,\ldots,t_{k-1})\,dt_1\,dt_2\,\ldots\,dt_{k-1},$$

при $\Delta = a\sqrt{n}$ получаем, что $I = I(N,k,\Delta)$ — некоторая высчитываемая константа, для которой выполняются соотношения

$$\lambda(N,k,\Delta) \leq I\ \exp\left\{\frac{ka(1+a^2)+\varepsilon}{2\sqrt{n}}\right\}\ \left(1+\frac{1}{I}\sqrt{\frac{k}{(2\pi kn)^{k-1}}}\ \times\right.$$
$$\times\ \left(\exp\left\{\frac{-ka^2}{2(k-1)}\right\}\ 2^{k-1}a^{k-2}(k-1)(\sqrt{n})^{k-2} +\right.$$
$$\left.\left.+\ (2a)^k(k-1)(\sqrt{n})^{k-2}\left(1+\frac{k-1}{a\sqrt{n}}\right)\right)\right) \leq$$
$$\leq I\ \exp\left\{\frac{ka(1+a^2)+\varepsilon}{2\sqrt{n}}\right\}\left(1+\frac{\sqrt{2^{k-1}}a^{k-2}(k-1)(\gamma+2a^2(1+\varepsilon))}{I\sqrt{k^{k-2}n}}\right) \leq$$
$$\leq I\ \exp\left\{\frac{ka(1+a^2)+\varepsilon}{2\sqrt{n}}\right\}\exp\left\{\frac{\sqrt{2^{k-1}}a^{k-2}(k-1)(\gamma+2a^2(1+\varepsilon))}{I\sqrt{k^{k-2}n}}\right\} \leq$$
$$\leq I\ \exp\left\{\frac{1}{\sqrt{n}}\left(\frac{ka(1+a^2)+\varepsilon}{2}+\frac{\sqrt{2^{k-1}}a^{k-2}(k-1)(\gamma+2a^2(1+\varepsilon))}{I\sqrt{k^{k-2}}}\right)\right\} \leq$$
$$\leq I\ \left(1+\frac{1}{\sqrt{n}}\left(\frac{ka(1+a^2)+\varepsilon}{2}+\right.\right.$$

$$+\frac{\sqrt{2^{k-1}}a^{k-2}(k-1)(\gamma+2a^2(1+\varepsilon))}{I\sqrt{k^{k-2}}}+\varepsilon\Bigg)\Bigg);$$

$$\lambda(N,k,\Delta)\geq I\ \exp\left\{-\frac{ka(7-2a^2)+\varepsilon}{14\sqrt{n}}\right\}\ \left(1-\frac{1}{I}\sqrt{\frac{k}{(2\pi kn)^{k-1}}}\times\right.$$

$$\times\ \left(\exp\left\{\frac{-ka^2}{2(k-1)}\right\}\ 2^{k-1}a^{k-2}(k-1)(\sqrt{n})^{k-2}+\right.$$

$$\left.\left.+\ (2a)^k(k-1)(\sqrt{n})^{k-2}\left(1-\frac{2}{(k-1)!}\right)\right)\right)\geq$$

$$\geq I\ \exp\left\{-\frac{ka(7-2a^2)+\varepsilon}{14\sqrt{n}}\right\}\left(1-\frac{\sqrt{2^{k-1}}a^{k-2}(k-1)(\gamma+2a^2)}{I\sqrt{k^{k-2}n}}\right)\geq$$

$$\geq I\ \left(1-\frac{ka(7-2a^2)+\varepsilon}{14\sqrt{n}}\right)\left(1-\frac{\sqrt{2^{k-1}}a^{k-2}(k-1)(\gamma+2a^2)}{I\sqrt{k^{k-2}n}}\right)\geq$$

$$\geq I\ \left(1-\frac{1}{\sqrt{n}}\left(\frac{ka(7-2a^2)+\varepsilon}{14}+\frac{\sqrt{2^{k-1}}a^{k-2}(k-1)(\gamma+2a^2)}{I\sqrt{k^{k-2}}}+\varepsilon\right)\right),$$

где $\gamma=e^{\frac{-ka^2}{2(k-1)}}$.

Список литературы

[1] Де Гроот М., *Оптимальные статистические решения*, Мир, Москва, 1974, 491 с.

[2] Крамер Г., *Математические методы статистики*, Мир, Москва, 1975, 648 с.

[3] Ширяев А. Н., *Вероятность*, «Наука», Москва, 1980, 576 с.

[4] Яблонский С. В., *Введение в дискретную математику*, «Наука», Москва, 1986, 384 с.

[5] Чашкин А. В., *Дискретная математика*, Издательский центр «Академия», Москва, 2012, 352 с.

[6] Галатенко А. В., Галатенко В. В., "О расстоянии Хэмминга между почти всеми функциями алгебры логики", *Фундаментальная и прикладная математика*, **15**:5 (2009), 43–47.

Статья поступила 18 ноября 2025 г.

Estimates for the Proportion of Sequences with Small Deviations from the Mean

V. V. Kochergin

The problem of the magnitude of the restriction in a k-digit set on the permissible deviation from the average count of digits having each of the k values under the condition of a given proportion of such sequences. An approach is proposed that allows monitoring the influence of various parameters on the final result at each stage, and estimates of the accuracy of the approximate solution are provided.

Keywords: k-digit hypercube, Stirling's formula, proportion of sequences.

References

[1] DeGroot M. H, *Optimal statistical decisions*, McGraw-Hill, New York, 1969, 520 pp.

[2] Cramer H., *Mathematical methods of statistics*, Princeton university press, 1946, 529 pp.

[3] Shiryayev A. N., *Probability*, Springer Science+Business Media, New York, 1984, 573 pp.

[4] Yablonsky S. V., *Introduction to discrete mathematics*, Mir, Moscow, 1989, 384 pp.

[5] Chashkin A. V., *Discrete mathematics*, Publishing Center «Academy», Moscow, 2012 (In Russian), 352 pp.

[6] Galatenko A. V., Galatenko V. V., "On the Hamming distance between almost all Boolean functions", *Journal of Mathematical Sciences*, **172**:5 (2011), 650–653. DOI: 10.1007/s10958-011-0210-4

Received on November 18, 2025

О некоторых подходах к получению асимптотических оценок сложности реализации систем булевых функций в модели клеточных схем

С. А. Ложкин* , В. С. Зизов‡

В работе рассматриваются вопросы асимптотической сложности реализации систем булевых функций в модели клеточных схем из функциональных и коммутационных элементов. Основное внимание уделено получению асимптотических оценок высокой степени точности для площади клеточных схем, реализующих достаточно большие классы булевых функций. Сформулированы методы получения верхних и нижних асимптотических оценок, применимые к широким классам функций. На основе этих методов как следствие получены асимптотические оценки высокой степени точности для системы всех самодвойственных булевых функций. Для получения верхних оценок используется конструктивный подход, основанный на сведении исследуемых классов функций к универсальным многополюсникам меньшего порядка и последующей модификации реализующих их клеточных схем. В результате для класса самодвойственных булевых функций установлены согласованные верхние и нижние асимптотические оценки площади, что приводит к асимптотическому равенству порядка роста сложности их реализации в модели клеточных схем.

Ключевые слова: клеточные схемы, площадь схемы, асимптотические оценки, асимптотические оценки высокой степени точности, системы булевых функций, самодвойственные функции, универсальный многополюсник.

* *Ложкин Сергей Андреевич* — профессор, д.ф.-м.н., заведующий кафедрой математической кибернетики факультета ВМК МГУ, e-mail: `lozhkin@cs.msu.ru`, ORCID: 0000-0002-8952-6046.

Lozhkin Sergey Andreevich — Professor, Doctor of Physics and Mathematics, Head of the Chair of Mathematical Cybernetics, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University..

‡ *Зизов Вадим Сергеевич* — м.н.с. кафедры математической кибернетики факультета ВМК МГУ, e-mail: `vzs815@gmail.com`, ORCID: 0000-0002-4053-4803.

Zizov Vadim Sergeevich — Chair of Mathematical Cybernetics, Faculty of Computational Mathematics and Cybernetics, Lomonosov Moscow State University..

1. Введение

Впервые модель клеточных схем (КС) в «стандартном» базисе $Б_0$ из функциональных и коммутационных элементов, где под сложностью КС понималась ее площадь, была предложена в 1967 году С.С. Кравцовым в [1]. Она положила начало исследованиям, связанным с решением в рамках данной модели различных задач и, в первую очередь, задачи синтеза.

Будем предполагать, что базис Б является конечным функционально полным базисом в модели клеточных схем, состоящим из функциональных и коммутационных элементов, каждый из которых имеет форму единичного квадрата, причем входы и выходы каждого элемента однократно располагаются на серединах его сторон (см. рис. 1 и ср. с [1]).

Каждая КС над базисом Б представляет собой прямоугольную решетку, состоящую из элементов базиса, соединенных между собой «корректным» [9] способом через середины общих сторон. При этом входные и выходные булевы переменные (БП) данной КС, связанные с её входами и выходами, однократно приписываются расположенным на границе схемы входам и выходам её элементов соответственно. Структура соединений входов, функциональных элементов и выходов КС задает схему из функциональных элементов над функциональной частью базиса Б (см. [2]).

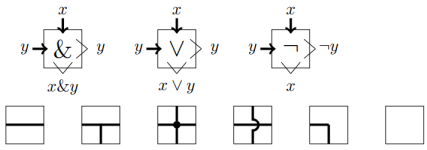

Рис. 1: Базис $Б'_0$, состоящий из 3 функциональных, 5 коммутационных элементов и изолятора. Нетождественные выходы функциональных элементов сонаправлены осям символов конъюнкции и дизъюнкции. Все элементы допускают поворот и отражение.

Задача (индивидуального) синтеза связана с изучением функционала сложности $A_Б(F)$, который определяется для системы булевых функций (БФ) $F = (f_1, \ldots, f_m) \in P_2^m(n)$, где $P_2(n)$ – множество всех БФ от булевых переменных (БП) $X(n) = \{x_1, \ldots, x_n\}$. Функционал $A(\Sigma)$ называемый, обычно, её площадью, определяется как произведение линейных размеров схемы, то есть длины $\lambda(\Sigma)$ на её ширину $h(\Sigma)$, где $\lambda(\Sigma) \geq h(\Sigma)$. При этом считается, что функционал $A_Б(F)$ (соответственно $h_Б(F)$) равен мини-

мальной площади (соответственно высоте) КС в базисе Б, реализующих систему БФ F.

Во многих случаях реализуемая система БФ F имеет вид $F = \vec{H}$, где $H \subseteq P_2(n)$, то есть состоит из всех БФ множества H, упорядоченных в соответствии с лексикографической нумерацией их столбцов значений. При этом рассматривается, обычно, последовательность множеств $H(1), H(2), \ldots, H(n), \ldots$, где $H(n) \subseteq P_2(n)$ при всех $n, n = 1, 2, \ldots$, и устанавливается (асимптотическое) поведение последовательности

$$A_{\text{Б}}(\vec{H}(n)) \text{ при } n = 1, 2, \ldots.$$

Так, в работе [1], а также в работе Н.А. Шкаликовой [3], был установлен порядок роста вида $n \cdot 2^n$ для площади $A_{\text{Б}_0}(\vec{K}(n))$, вида $n \cdot 2^{2^n}$ для площади $A_{\text{Б}_0}(\vec{P}_2(n))$ и вида $\log n \cdot 2^n$ для площади $A_{\text{Б}_0}(\vec{S}(n))$, где множество $K(n)$ состоит из всех элементарных конъюнкций ранга n от БП $\text{X}(n)$, то есть $\vec{K}(n)$ является т.н. *дешифратором* порядка n, а множество $S(n)$ состоит из всех симметрических БФ из $P_2(n)$. Напомним, что система БФ $\vec{P}_2(n)$ называется, обычно, *универсальным многополюсником порядка* n. Напомним также, что симметрической называется функция, значение которой не зависит от перестановки её аргументов.

В работах Г.В. Калачева [4] в качестве базиса клеточных элементов рассматриваются все булевы операторы, суммарное число входов и выходов которых не превышает 4. В качестве меры мощности рассматривается средний потенциал, количество выходов элементов, выдающих единицу на заданном входном наборе схемы, усреднённое по всем входным наборам. Получены нижние оценки функции Шеннона в зависимости от ограничений на расположение выходов схем. В работе [5] определяются две меры мощности схем – потенциал и переключательная мощность, причем каждая из них исследуется как в среднем, так и в худшем случае. Потенциал равен количеству выходов клеточных элементов, равных единице, и характеризует «статическую» активность КСФЭ. Переключательная мощность определяется на паре входных наборов и равна количеству выходов клеточных элементов, поменявших значение при изменении входного набора с первого на второй. Эта мера мощности характеризует «динамическую активность» КСФЭ и более приближена к реальному энергопотреблению интегральной схемы, построенной по данной «клеточной» модели. В работе [6] исследуется поведение функций Шеннона для этих мер мощности и для различных классов частичных булевых операторов, а также связь этих мер с площадью и глубиной. В частности, устанавливается оценка площади для оптимальных по порядку мощности клеточных схем, равная $m2^n$ для всюду определённых функций, где за m обозначено число выходов.

По аналогии с работой [7] последовательность $R(n), n = 1, 2, \ldots$ такую, что $R(n) \leq A_{\text{Б}}(\vec{H}(n))$ (соответственно $R(n) \geq A_{\text{Б}}(\vec{H}(n))$) будем считать нижними (соответственно верхними) **асимптотическими оценками высокой степени точности** (АОВСТ) для последовательности $A_{\text{Б}}(\vec{H}(n))$ в модели КС над базисом Б, если

$$|R(n) - A_{\text{Б}}(\vec{H}(n))| = O\left(\frac{A_{\text{Б}}(\vec{H}(n))}{h_{\text{Б}}(\vec{H}(n))}\right).$$

В работе [8] были установлены асимптотически точные верхние и нижние АОВСТ для площади схем над базисом $\text{Б}_0'$, реализующих дешифратор порядка n, которые имеют вид $n2^{n-1}(1 \pm O(\frac{1}{n}))$. В работе [9] получены асимптотически точные нижние оценки вида $A_{\text{Б}_0'}(\mu_n) \geqslant n2^{n-1}(1 - \frac{\log n}{n})$ для т.н. мультиплексорной БФ μ_n порядка n, а также её верхние АОВСТ.

В работе [10] были установлены верхние и нижние АОВСТ для сложности $A_{\text{Б}_0'}(\vec{P}_2(n))$, то есть для площади универсального многополюсника порядка n в модели клеточных схем над базисом $\text{Б}_0'$, имеющие вид

$$n \cdot 2^{2^n-1} - O(n^2) \leq A_{\text{Б}_0'}(\vec{P}_2(n)) \leq (n+6)2^{2^n-1} + \frac{3n}{2^n}2^{2^n-1}, \qquad (1)$$

Булева функция $f(x_1, \ldots, x_n)$ называется *самодвойственной*, если

$$f(x_1, \ldots, x_n) = \overline{f(\overline{x_1}, \ldots, \overline{x_n})},$$

где $\overline{x_i}$ — отрицание переменной x_i. Множество всех самодвойственных БФ от БП $X(n)$ будем обозначать $D(n)$.

В настоящей работе аналогичные (1) результаты установлены для класса $D(n)$, то есть получены верхние и нижние АОВСТ для площади $A_{\text{Б}_0'}(\vec{D}(n))$ (см. также [11]).

2. Метод получения нижних оценок сложности исследуемых систем булевых функций и его применение к классу самодвойственных БФ

Перейдем далее к получению нижних оценок сложности реализации достаточно «больших» систем БФ на основе т.н. тестового подхода, пример применения которого для системы ФАЛ $\vec{P}_2(n)$ дан в [10], а для системы $S(n)$ в [11].

Теорема 1. *Для $n \geqslant 3$ и произвольной системы БФ вида $\vec{G}$, где множество $G, G \subseteq P_2(n)$, удовлетворяет неравенству $|G| \geq 2^{2^{n/2}} + 2$, не*

содержит БП из $X(n)$, но для каждого $i, i \in [1, n]$ содержит хотя бы одну БФ, существенно зависящую от БП x_i, справедливо неравенство

$$A(\vec{G}) \geq (2h_G - n) \cdot \left(\frac{|G| + n}{2} - 2h_G + n \right), \tag{2}$$

где

$$h_G = \left\lceil \log \left\lceil \log \left\lfloor \frac{|G| - 1}{2} \right\rfloor \right\rceil \right\rceil .$$

Доказательство. Заметим, сначала, что клеточная схема Σ, реализующая указанную в условии теоремы систему ФАЛ $\vec{G}$ удовлетворяет неравенствам

$$\lambda(\Sigma) \geq h(\Sigma) \geq h(\vec{G}), \qquad 2(h(\Sigma) + \lambda(\Sigma)) \geq n + |\vec{G}|. \tag{3}$$

Решая задачу минимизации произведения $h(\Sigma) \cdot \lambda(\Sigma)$ при условиях (3), получим

$$A(\Sigma) \geq \frac{1}{2} h(\vec{G}) \cdot (|G| + n - 2h(\vec{G})). \tag{4}$$

Нижняя оценка (2) для системы функций $\vec{G}$ вытекает из (4) и неравенства $h(\vec{G}) \geq 2h_G - n$, которое мы далее докажем, заметив, что $h_G \geq 2h_G - n$.

Пусть для клеточной схемы Σ, реализующей систему ФАЛ $\vec{G}$, выполняется неравенство $h = h(\Sigma) < 2h_G - n$. Тогда из соотношений

$$N = \left\lfloor \frac{|G| - 1}{2} \right\rfloor \geq h_G, \tag{5}$$

верных при условии $|\vec{G}| \geq 5$, которое выполняется в случае $n \geqslant 3$, как это оговорено в формулировке теоремы, следует, что схему Σ можно подходящим вертикальным сечением π разделить на «левую» подсхему Σ' и «правую» подсхему Σ'', каждая из которых содержит не менее, чем N выходов этой схемы.

Действительно, пусть $\hat{\pi}$ — вертикальное сечение КС Σ, дисбаланс

$$d(\hat{\pi}) = |m' - m''|$$

которого не меньше 2, где $\hat{m}'$ и $\hat{m}''$ — число выходов КС Σ, расположенных слева и справа от $\hat{\pi}$ соответственно. Тогда в случае $\hat{m}' < \hat{m}''$ (соответственно $\hat{m}' > \hat{m}''$) сдвигая $\hat{\pi}$ на 1 вправо (соответственно влево)

мы получим сечение, дисбаланс которого не больше, чем $d(\hat{\pi})$, но и не меньше, чем $d(\hat{\pi}) - 2$.

Повторяя указанные сдвиги до тех пор, пока дисбаланс сечения не окажется в сегменте $[0, 1]$, мы либо получим искомое сечение π, либо достигнем вертикальной стороны, что влечет за собой выполнение неравенств $h(\Sigma) \geqslant \frac{|G|}{2} > h_G$, последнее из которых в силу (5) противоречит предположению о том, что $h(\Sigma) < 2h_G - n \leqslant h_G$.

Пусть, далее, подсхемы Σ' и Σ'' содержат входные переменные схемы Σ из множеств X' и X'' соответственно, а на содержащихся в Σ' и Σ'' выходах схемы Σ в ней реализуются функции из множеств G' и G'' соответственно, где

$$X' \cap X'' = \varnothing,\ X' \cup X'' = X(n), \qquad G' \cap G'' = \varnothing,\ G' \cup G'' = G.$$

Положим

$$|X'| = n', |X''| = n'', |G'| = m', |G''| = m'',$$

и заметим, что согласно вышесказанному,

$$n' + n'' = n, \quad m' + m'' = |G|, \quad m' \geq N, \quad m'' \geq N. \tag{6}$$

Пусть, наконец, на сечении π располагаются p' и p'' дополнительных входов схемы Σ' и схемы Σ'' соответственно, которые присоединены к p' и p'' дополнительным выходам схемы Σ'' и схемы Σ' соответственно. Выберем в кубе $B^n = B^{n'}(X') \times B^{n''}(X'')$ множество наборов A', которое является тупиковым диагностическим тестом для множества функций G' и для которого, очевидно, с учетом (6), верны неравенства

$$|A'| \geq \lceil \log m' \rceil \geq \lceil \log N \rceil. \tag{7}$$

Заметим, что в силу особенностей структуры схемы Σ и тупиковости теста A' из любых двух различных наборов значений переменных (X', X'') вида (α', β'') и (α', γ''), которым соответствует один и тот же вектор значений дополнительных выходов схемы Σ'', в A' может входить только один. Это означает, что

$$|A'| \leq 2^{n'+p'},$$

и, следовательно, в силу (7) справедливо неравенство

$$n' + p' \geq \lceil \log \lceil \log N \rceil \rceil.$$

Аналогичным образом доказывается неравенство

$$n'' + p'' \geq \lceil \log \lceil \log N \rceil \rceil,$$

суммируя которое с (7) и учитывая (4), получим неравенство

$$h(\Sigma) \geq 2h_G - n,$$

приводящее к противоречию со сделанным ранее предположением. □

Из теоремы 1 вытекают нижние АОВСТ для сложности системы $\vec{D}(n)$.

Следствие 1. *При $n = 3, 4, \ldots$ выполнено:*

$$A(\vec{D}(n)) \geqslant \frac{n-2}{2}(2^{2^{n-1}} - n + 2) \geqslant n2^{2^{n-1}-1} - 2^{2^{n-1}} - \frac{n^2}{2}.$$

3. Методы получения верхних оценок сложности исследуемых систем булевых функций. Верхние оценки системы всех самодвойственных функций

Для получения верхних оценок сложности реализации последовательности систем булевых функций $\vec{H}(n), n = 1, 2, \ldots$, где $H(n) \subseteq P_2(n)$ в модели клеточных схем над базисом $Б_0'$ можно использовать следующий подход, который был разработан в [10] для универсальных многополюсников.

Покроем множество $H(n)$, рассматриваемое как множество наборов куба B^{2^n}, непересекающимися единичными сферами $S_1, \ldots, S_p$ данного куба с «центрами» $g_1, \ldots, g_p$. Построим для каждого $i, i = 1, \ldots, p$, (двустороннюю) КС Σ_i в базисе $Б_0'$, которая реализует все БФ из $H(n) \cap S_i$, имея длину $\frac{1}{2}|H(n) \cap S_i|(1 + \bar{o}(1))$ и высоту $n + \underline{O}(1)$.

Эта КС, пропуская через себя по горизонтали БП из $X(n)$, реализует сначала БФ g_i, а затем осуществляет коррекцию ее столбца значений для получения каждой БФ из $H(n) \cap S_i$, используя для этого один вертикальный ряд. Последовательное горизонтальное соединение КС $\Sigma_1, \ldots, \Sigma_p$ дает искомую КС для системы $\vec{H}(n)$.

В настоящей работе мы будем использовать также метод модификации последовательности КС $\Sigma_n', n = 1, 2, \ldots$, реализующих последовательность систем БФ $\vec{H}(n)$, где $H(n) \subseteq P_2(n)$, для реализации «близкой» к ней последовательности систем БФ $\vec{F}(n), F(n) \subseteq P_2(n), n = 1, 2, \ldots$.

Указанная модификация возможна, если $|F(n)| = |H(n)|$ и каждая ФАЛ $f, f \in F(n)$, может быть представлена в виде $f = g(h, x_1, \ldots, x_k)$, где $k = k(n) = \bar{o}(n), h \in H(n)$, а $g \in P_2(k+1)$, причем данное представление

задает взаимно-однозначное соответствие между множествами $F(n)$ и $H(n)$. При этом «переделка» КС Σ'_n, реализующей систему БФ $\vec{H}(n)$, в схему Σ''_n, реализующую систему БФ $F(n)$, заключается в следующем.

1. Приведем как выше, так и ниже КС Σ'_n по k горизонтальных проводников, передающих БП $x_1, \ldots, x_k$ а затем выделим в КС Σ'_n непересекающиеся группы, состоящие не более, чем из t соседних расположенных на одной из ее горизонтальных сторон выходов.

2. Для каждой такой группы G, расположенной на нижней границе Σ'_n, и каждого ее выхода y, на которой реализуется БФ h, построим такую КС S, реализующую БФ вида $g(y, x_1, \ldots, x_k)$ из представления $f = g(h, x_1, \ldots, x_k)$, которая пропускает БП $x_1, \ldots, x_k$ сверху вниз, получает вход y с правой вертикальной стороны и отправляет с нее же, но «ниже», чем y, выход z.

3. Освободим справа от группы G вертикальную полосу, в которой друг над другом разместим указанные схемы S для всех ее выходов.

4. Каждый выход y схемы Σ'_n из группы G «опустим» по вертикали до уровня соответствующего ему входа связанной с $y = h$ схемы S и соединим их между собой горизонтальным проводником, после чего выведем на соответствующую вертикаль выход схемы S и объявим его выходом схемы Σ''_n, на котором реализуется БФ $f = g(h, x_1, \ldots, x_k)$.

5. Аналогичные преобразования проведем для каждой группы, расположенной на верхней границе Σ'_n.

Полученная описанным выше способом КС Σ''_n будет реализовывать систему БФ $\vec{F}(n)$, а для ее длины и высоты будут выполняться неравенства:

$$\lambda(\Sigma''_n) \leqslant \lambda(\Sigma'_n) + \left\lceil \frac{|H(n)|}{t} \right\rceil \cdot \lambda_s$$
$$h(\Sigma''_n) \leqslant h(\Sigma'_n) + 2th_s + 2k,$$

где λ_s и h_s — максимальная длина и высота схем S описанного выше вида.

Применение этого метода дает асимптотически точные верхние оценки для ряда классов БФ, включая класс самодвойственных БФ, который рассмотрен в следующем утверждении.

Теорема 2. *Для класса самодвойственных функций $D(n)$ справедливо неравенство:*

$$A_{Б'_0}(\vec{D}(n)) \leqslant 2^{2^{n-1}-1}(n + 9\sqrt{n} + O(1)).$$

Доказательство. Построим схему Σ_n от n переменных, реализующую систему $\vec{D}(n)$. При этом будем использовать схему $S_{\oplus}$, реализующую сумму по модулю 2 двух переменных так, как показано на рис. 2.

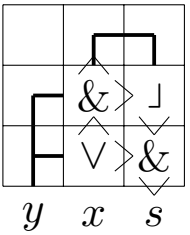

Рис. 2: Пример схемы $S_\oplus$, реализующей сумму по модулю два $s(x, y) = x \oplus y$.

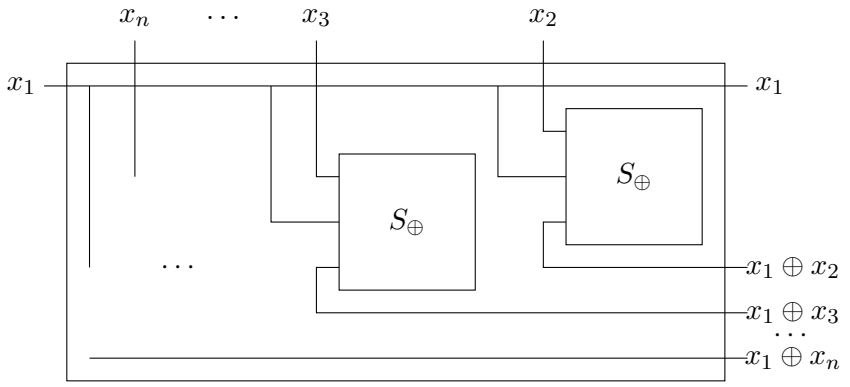

Рис. 3: Вид схемы Σ'_n, реализующей систему БФ $\{x_2 \oplus x_1, \ldots, x_n \oplus x_1\}$.

Применяя описанный выше подход, сведем задачу синтеза КС для системы БФ $\vec{D}(n)$, к задаче синтеза КС, реализующей систему $\vec{P}_2(n-1)$. Для этого сопоставим функции $f, f(x_1, x_2, \ldots, x_n) \in D(n)$, функцию g из $P_2(n-1)$ так что $f(x_1, x_2, \ldots, x_n) = g(x_1 \oplus x_2, \ldots, x_1 \oplus x_n) \oplus x_1$.

Построим сначала КС, реализующую систему БФ

$$\vec{P}_2(x_2 \oplus x_1, \ldots, x_n \oplus x_1)$$

в результате горизонтальной стыковки ее подсхем Σ'_n и Σ''_n. Схема Σ'_n состоит из ряда схем $S_\oplus$, каждая из которых принимает на вход переменные x_1 с левой и x_i с верхней стороны, а возвращает $x_i \oplus x_1$, формируя вертикальный ряд из $n-1$ переменных функции g. Длина схемы Σ'_n равна $(n-1)\lambda(S_\oplus)$, а высота равна $n - 1 + h(S_\oplus)$.

Функции g из $P_2(n-1)$ реализуем универсальным многополюсником Σ''_n из [10], имеющим длину $2^{2^{n-1}-1} + \underline{O}(\frac{2^{2^{n-1}}}{n})$ и высоту $n + C$. Разобьём её выходы на части $G_1, G_2, \ldots, G_m$, в каждой из которых содержится не

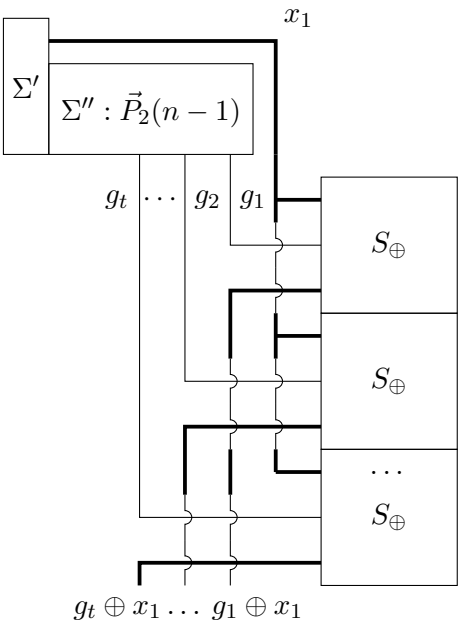

Рис. 4: Вид схемы Σ_n, реализующей систему БФ $\vec{D}(n)$.

более t «соседних» выходов, расположенных на одной из горизонтальных сторон схемы. При этом $m = \lceil 2^{2^{n-1}}/t \rceil$.

Опишем теперь переход от полученной схемы к схеме Σ_n, реализующей $\vec{D}(n)$ в соответствии с изложенным ранее подходом. Продлим все выходы КС Σ''_n вниз, добавив вертикальный ряд, состоящий из схем $S_\oplus$. Каждая схема $S_\oplus$ принимает на вход выход y_j схемы G_i по горизонтальному ряду, и x_1 по вертикальному. Выход схемы $S_\oplus$ направлен горизонтально, и возвращает результат $y_j \oplus x_1$ в вертикаль y_j. Такая реализация требует дополнительной высоты $h(S_\oplus)t$ и длины $\lambda(S_\oplus)$.

Получим размеры схемы Σ_n:

$$\lambda(\Sigma_n) = m(t+\lambda(S_\oplus)) + n\lambda(S_\oplus) = 2^{2^{n-1}-1} + 2^{2^{n-1}-1}\frac{\lambda(S_\oplus)}{t} + n\lambda(S_\oplus) + O(t)$$
$$h(\Sigma_n) = n + 12 + 2th(S_\oplus).$$

Возьмем $t = \lfloor \sqrt{n} \rfloor$, вспомним, что $\lambda(S_\oplus) = 3, h(S_\oplus) = 3$, и оценим площадь схемы Σ_n следующим образом

$$A(\Sigma_n) = \left(2^{2^{n-1}-1} + 2^{2^{n-1}-1}\frac{3}{\sqrt{n}} + n\right)(n + 12 + 6\sqrt{n}) + O(n\sqrt{n}),$$

то есть выполняется равенство

$$A(\Sigma_n) = n2^{2^{n-1}-1} + 9 \cdot 2^{2^{n-1}-1}\sqrt{n} + 30 \cdot 2^{2^{n-1}-1} + O\left(2^{2^{n-1}-1} \cdot \frac{\sqrt{n}}{n}\right),$$

из которого следует утверждение настоящей теоремы

$$A(\vec{D}(n)) \leqslant 2^{2^{n-1}-1}\left(n + 9\sqrt{n} + O(1)\right).$$

□

Таким образом, следствие 1 из теоремы 1 и теорема 2 устанавливают асимптотическое равенство $A(\vec{D}(n)) \sim n2^{2^{n-1}-1}$, которое было объявлено в [11].

Список литературы

[1] С. С. Кравцов, "О реализации функций алгебры логики в одном классе схем из функциональных и коммутационных элементов", *Проблемы кибернетики*, 1967, № 19, 285–292.

[2] С. А. Ложкин, "Лекции по основам кибернетики. Учебное пособие", *М.: МГУ*, 2004.

[3] Н. А. Шкаликова, "О реализации булевых функций схемами из клеточных элементов", *Математические вопросы кибернетики*, 1989, № 2, 177–197.

[4] Г. В. Калачев, "Нижние оценки мощности плоских схем, реализующих частичные булевы операторы", *Интеллектуальные системы. Теория и приложения*, **18** (2014), 279–322.

[5] Г. В. Калачев, "Порядок мощности плоских схем, реализующих булевы функции", *Дискретная математика*, **26**:1 (2014), 49–74.

[6] Г. В. Калачев, "Об одновременной оптимизации площади, мощности и глубины плоских схем, реализующих частичные булевы операторы", *Интеллектуальные системы. Теория и приложения*, **20** (2016), 203–266.

[7] С. А. Ложкин, "Оценки высокой степени точности для сложности управляющих систем из некоторых классов", *Математические вопросы кибернетики*, 1996, № 6, 189–214.

[8] С. А. Ложкин, В. С. Зизов, "Уточненные оценки сложности дешифратора в модели клеточных схем из функциональных и коммутационных элементов", *Учен. зап. Казан. ун-та. Сер. Физ.-матем. науки*, **162**:3 (2020), 322–334. DOI: 10.26907/2541-7746.2020.3.322-334.

[9] С. А. Ложкин, В. С. Зизов, "Асимптотически точные оценки для площади мультиплексоров в модели клеточных схем", *Дискретная математика*, **34**:4 (2022), 52–68. DOI: 10.4213/dm1712.

[10] С. А. Ложкин, В. С. Зизов, "Асимптотические оценки высокой степени точности для сложности универсального многополюсника в модели клеточных схем", *Вестн. Моск. ун-та. Сер. 1. Матем., мех.*, 2025, № 4, 61–65. DOI: 10.55959/MSU0579-9368-1-66-4-9.

[11] С. А. Ложкин, В. С. Зизов, “О сложности реализации универсального клеточного многополюсника для класса самодвойственных функций”, *Проблемы теоретической кибернетики: материалы XX Международной научной конференции (Москва, 5-8 декабря 2024)*, ООО МАКС Пресс, Москва, 2025, 79–80. DOI: 10.29003/m4678.978-5-317-07402-9.

Статья поступила 12 февраля 2026 г.

On Certain Approaches to Obtaining Asymptotic Complexity Estimates for the Implementation of Systems of Boolean Functions in the Cellular Circuit Model

S. A. Lozhkin, V. S. Zizov

The paper addresses the problem of asymptotic complexity of implementing systems of Boolean functions in the model of cellular circuits composed of functional and switching elements. The main focus is on deriving high-accuracy asymptotic estimates for the area of cellular circuits realizing sufficiently large classes of Boolean functions. Methods for obtaining upper and lower asymptotic bounds applicable to wide classes of functions are formulated. As a consequence of these methods, high-accuracy asymptotic estimates are obtained for the system of all self-dual Boolean functions. Upper bounds are derived using a constructive approach based on reducing the considered classes of functions to universal multipoles of smaller order and subsequent modification of the corresponding cellular circuits. As a result, matching upper and lower asymptotic bounds on the circuit area are established for the class of self-dual Boolean functions, leading to an asymptotic equality for the order of growth of their implementation complexity in the cellular circuit model.

Keywords: cellular circuits, circuit area, asymptotic bounds, high-accuracy asymptotic bounds, systems of Boolean functions, self-dual functions, universal multipole.

References

[1] S. S. Kravtsov, “On implementation of Boolean functios in one class of circuits of functional and switching elements”, *Problemy Kibernetiki*, 1967, № 19, 285–292 (In Russian).

[2] S. A. Lozhkin, *Lectures on the Foundations of Cybernetics. Textbook.*, MSU, Moscow, 2004 (In Russian).

[3] N. A. Skhalikova, “On implementation of Boolean functions by cellular circuits”, *Matematicheskie Voprosy Kibernetiki*, 1989, № 2, 177–197 (In Russian).

[4] G. V. Kalachev, "Lower estimates of the power of planar circuits implementing partial Boolean operators", *Intellektualnye Systemy. Teoria i Prilojenia*, **18** (2014), 279–322 (In Russian).

[5] G. V. Kalachev, "The power order of planar circuits implementing Boolean functions", *Discrete Mathematics and Applications*, **26** (2014), 49–74 (In Russian).

[6] G. V. Kalachev, "On the simultaneous optimization of area, power and depth of planar circuits implementing partial Boolean operators", *Intellektualnye Systemy. Teoria i Prilojenia*, **20** (2016), 203–266 (In Russian).

[7] S. A. Lozhkin, "High-precision estimates for the complexity of control systems from some classes", *Matematicheskie Voprosy Kibernetiki*, 1996, №6, 189–214 (In Russian).

[8] S. A. Lozhkin, V. S. Zizov, "Refined estimates of the decoder complexity in the model of cellular circuits with functional and switching elements", *Uchenye Zapiski Kazanskogo Universiteta. Seriya Fiziko-Matematicheskie Nauki*, **162**:3 (2020), 322–334. DOI: 10.26907/2541-7746.2020.3.322-334 (In Russian).

[9] S. A. Lozhkin, V. S. Zizov, "Asymptotically sharp estimates for the area of multiplexers in the cellular circuit model", *Discrete Mathematics and Applications*, **34**:2 (2024), 103–115. DOI: 10.1515/dma-2024-0009.

[10] S. A. Lozhkin, V. S. Zizov, "Asymptotic estimates of a high degree of accuracy for the complexity of a universal multipole in a model of cellular circuits", *Moscow University Mathematics Bulletin*, **80** (2025), 255–260. DOI: 10.3103/S0027132225700500.

[11] S. A. Lozhkin, V. S. Zizov, "On the complexity of implementing a universal cellular multipole for a class of self-dual functions", *Problemy Teoreticheskoy Kibernetiki: Proceedings of the XX International Scientific Conference (Moscow, December 5–8, 2024)*, OOO Maks Press, Moscow, 2025, 79–80. DOI: 10.29003/m4678.978-5-317-07402-9 (In Russian).

Received on February 12, 2026

Сложность базовых булевых операторов для цифровой схемотехники

И. С. Сергеев*

Статья содержит обзор оценок сложности схем для базовых булевых преобразований, применяемых в цифровой схемотехнике, и эффективных методов синтеза таких схем. Изложение охватывает структурно простые функции и операторы, такие как счетчики, сумматоры, шифраторы, мультиплексоры, и исключает более сложные алгебраические операции с числами, многочленами и матрицами. Дополнительно рассмотрено несколько приложений к построению схем более узкого назначения.

Ключевые слова: булевы схемы, сложность, глубина, параллельные схемы, префиксные схемы, инкрементор, двусторонний счетчик, счетчик Грея, сумматор, компаратор, дешифратор, мультиплексор, шифратор, компрессор, пороговые симметрические функции, приоритетный шифратор, унарная кодировка, сортировка.

Введение

В работе предпринимается попытка собрать сведения о сложности самых простых и фундаментальных булевых преобразований, которые широко используются как в практической схемотехнике, так и в теоретических задачах синтеза схем. Речь идёт о шифраторах, мультиплексорах, компараторах и прочих операторах, структурно не более сложных, чем оператор сложения чисел. Рассматриваемые преобразования часто не имеют выраженного самостоятельного значения, а играют роль «строительных блоков» при решении содержательных задач. С этих преобразований обычно начинается изучение основ проектирования цифровых функциональных узлов в популярных учебниках по цифровой схемотехнике, см., например, [9, 11]. Поэтому в рамках настоящего обзора мы (неформально) называем их *базовыми*.

Теория алгоритмов умножения чисел или матриц, дискретных преобразований Фурье настолько развита, что требует для изложения многотомных изданий. При этом «рабочие лошадки» электроники, такие как оператор сдвига, приоритетный шифратор или унарный преобразователь обычно остаются в тени. В популярных монографиях по сложности вычислений, таких как [2, 3, 28], даются лишь фрагментарные сведения о

* *Сергеев Игорь Сергеевич* — д.ф.-м.н., нач. лаб., ФГУП «НИИ «Квант», Москва, e-mail: isserg@gmail.com, ORCID: 0000-0002-0622-0843.

Sergeev Igor Sergeevich — Dr.Habil., head of laboratory, FSUE "RDI "Kvant", Moscow.

базовых операторах. Замысел настоящего обзора состоит в том, чтобы представить по возможности полную картину.

Значительную часть излагаемых далее результатов и методов следует отнести к фольклору ввиду простоты и достаточной известности. Поэтому здесь они приводятся без указания литературных источников. В остальных случаях, когда речь идет о результатах, потребовавших определенных усилий, особенно это касается нижних оценок, ссылки на известные автору работы проставлены, как это принято. Некоторые пробелы автору пришлось заполнить самостоятельно, но при этом не прибегая к нетривиальным построениям.

Рассматривается реализация булевых преобразований в модели схем из функциональных элементов, т. е. булевых или логических схем, см., например, [13, 28]. Из стандартных математических моделей она наиболее точно соответствует реальным электронным схемам. Напомним, что *схема над базисом* (множеством функций) $\mathcal{B}$ — это ациклический ориентированный граф, в котором вершины, не имеющие входящих ребер, отмечены как входы, а некоторые вершины отмечены как выходы. Входам приписаны символы переменных или констант базиса $\mathcal{B}$, остальным вершинам (эти вершины называются функциональными элементами) — символы функций из базиса $\mathcal{B}$. Функционирование схемы определяется естественным образом, от входов к выходам: в каждой вершине вычисляется сопоставленная ей функция, аргументами которой служат функции, поступающие по входящим в вершину ребрам. Схема реализует оператор (т. е. систему функций) F, если на выходах схемы вычисляются все компоненты оператора. *Сложность* схемы определяется как число вершин в ее графе без учета входов. Сложность оператора F при реализации схемами над базисом $\mathcal{B}$ определяется как сложность минимальной реализующей его схемы. *Глубина* схемы — это длина (измеряемая в ребрах или функциональных элементах) максимального ориентированного пути, соединяющего вход и выход схемы. По аналогии, глубина оператора F определяется как минимальная глубина реализующей его схемы. Физически сложность примерно соответствует площади схемы, а глубина — задержке распространения сигнала от входов к выходам.

Мы ограничиваем рассмотрение базисом из всех двуместных булевых функций. Сложность схемы Φ над этим базисом обозначим через $\mathsf{C}(\Phi)$, а сложность оператора F — через $\mathsf{C}(F)$. Дополнительно введем функционал $\mathsf{C}_{\log}(F_n)$, обозначающий сложность реализации последовательности операторов F_n схемами глубины $O(\log n)$, где n — число входных переменных. Неформально говоря, это сложность параллельного вычисления оператора. В практической схемотехнике параллельным схемам отдается

предпочтение. Стоит отметить, что рассматриваемые далее операторы достаточно просты и допускают параллельную реализацию.

При проектировании электронных схем обычно доступны более широкие базисы, включающие многовходовые элементы, и даже элементы с несколькими выходами. Однако эффективные методы синтеза, как правило, достаточно универсальны, и могут быть адаптированы к произвольному базису. Напомним, что порядок сложности/глубины оператора в любом полном конечном базисе — один и тот же.

Приводимые оценки сложности характеризуют сложность вычислений в асимптотическом смысле, т.е. при числе входов $n \to \infty$. Однако хорошо известно, что асимптотически эффективные методы синтеза не обязательно дают хороший результат при практически значимых величинах n. Впрочем, упоминаемые далее простые методы обычно неплохо работают уже при самых малых n. Кроме того, при таких значениях n параллельные методы синтеза приводят к схемам с компромиссными характеристиками глубины и сложности.

Далее излагаются сведения о сложности и эффективных способах реализации базовых преобразований. К базовым операциям мы относим префиксные и суффиксные суммы, числовой инкремент/декремент, двусторонний счетчик, счетчик Грея, вычисление переносов, сложение и сравнение двух чисел, максимум/минимум двух чисел, дешифратор, мультиплексор, прямой и циклический сдвиг, шифратор, выделение первой единицы и номера ее позиции, суммирование битов и сравнение суммы с пороговым значением, вычисление ширины единичного блока, перевод в унарную кодировку, обрезка, сортировка битового массива. Оценки сложности перечисленных операций сведены в таблицу 1.

Обзор дополняется примерами применения базовых операций и полезных для синтеза идей на материале параллельных схем 2-селектора, счетчика с сохранением веса, множественного выбора и перестановки пары битов.

При изложении материала используются следующие обозначения:

$\mathbb{B} = \{0, 1\}$;

$\mathbb{B}^n$ — множество булевых строк или векторов длины n; по умолчанию, нумерация битов в строке — с нуля слева направо;

$[\![n]\!]$ — множество целых чисел от 0 до $n-1$, заданных двоичной записью длины $\lceil \log n \rceil$, нумерация разрядов — с нуля справа налево;

$|s|$ — длина булева вектора или строки;

$\nu(X)$ — двоичный вес (число единичных битов или разрядов) булевой строки или числа X;

$\overline{x}$, $x \vee y$, $x \wedge y$ или $x \cdot y$, $x \oplus y$, $x \sim y$ — булевы операции отрицания, дизъюнкции, конъюнкции, сложения по модулю 2, эквивалентности;

$X^\alpha = \bigwedge x_i^{\alpha_i}$ — элементарная конъюнкция вектора переменных $X = [x_1, x_2, \ldots]$, где $\alpha = [\alpha_1, \alpha_2, \ldots] \in \mathbb{B}^{|X|}$; по определению, $x^1 = x$ и $x^0 = \overline{x}$;

« $\|$ » — операция конкатенации строк;

$[\sigma]^m$ — вектор или строка длины m из булевых величин σ.

Все логарифмы далее по тексту имеют основание 2.

Сложность базовых операторов

Префиксные суммы. Оператор $\mathtt{PREF}_n^* : \mathbb{S}^n \to \mathbb{S}^n$ вычисляет семейство префиксных сумм n переменных из полугруппы $(\mathbb{S}, *)$:

$$p_i = x_1 * x_2 * \ldots * x_i, \quad 1 \le i \le n. \tag{1}$$

Дальнейшие применения, за исключением конструкций схем переносов, будут ограничены случаем $\mathbb{S} = \mathbb{B}$ и $* \in \{\vee, \wedge, \oplus\}$. Схемы, вычисляющие систему (1) над базисом $\{*\}$, называются префиксными.

Префиксным схемам посвящена обширная литература, см., например, [15, 24],[28, §3.1]. Мы ограничимся лишь краткими сведениями. Очевидно, оператор $\mathtt{PREF}_n^*$ реализуется схемой из $n-1$ операций $*$, в которой префиксные суммы вычисляются последовательно. Сложность C и глубина D схемы из операций $*$, вычисляющей префиксные суммы n переменных, связаны соотношением $C + D \ge 2n - 2$. Поэтому сложность параллельной схемы не меньше $2n - \Theta(\log n)$. Эта оценка достигается, например, схемами, построенными Ю. П. Офманом [5] (но в литературе их обычно называют схемами Брента—Кунга). В схеме на $n = 2^k$ входах старшая сумма p_n вычисляется полным двоичным деревом T глубины k. Любая недостающая сумма p_i вычисляется как $p_{\bigtriangledown i} * p'$, где $\bigtriangledown i$ обозначает число, получаемое из i обнулением младшего единичного разряда, а p' — подходящая подсумма, вычисленная в дереве T. Легко проверить, что схема имеет глубину $2k - 2$. Схема для 8 входов изображена на рис. 1. Схема с произвольным числом $n < 2^k$ входов получается урезанием схемы на 2^k входах.

В случае булевых операций $*$ все перечисленные выше оценки справедливы для схем над рассматриваемым базисом из двуместных булевых функций, в частности, $\mathsf{C}(\mathtt{PREF}_n^*) = n - 1$ и $\mathsf{C}_{\log}(\mathtt{PREF}_n^*) = 2n - \Theta(\log n)$.

Также встречается задача совместного вычисления префиксных и суффиксных сумм n переменных. Последние определяются как

$$s_i = x_i * x_{i+1} * \ldots * x_n, \quad 1 \le i \le n.$$

Сложность соответствующего оператора $\mathtt{PS}_n^* : \mathbb{S}^n \to \mathbb{S}^{2n-1}$ очевидно равна $2n-3$. Минимальные параллельные префиксно-суффиксные схемы имеют

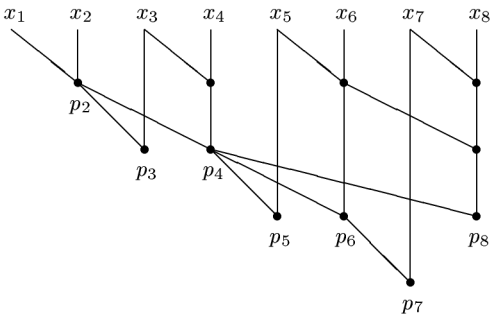

Рис. 1: Префиксная схема Офмана (Брента—Кунга)

сложность $3n - \Theta(\log n)$. Такую схему можно, например, построить совмещением префиксной и суффиксной схем Офмана — дерево, вычисляющее сумму $p_n = s_1$ всех переменных, является общей частью этих двух схем.

Инкрементор. Оператор $\mathtt{INC}_n : [\![2^n]\!] \to [\![2^n]\!]$ увеличивает n-разрядное число на единицу по модулю 2^n.

Верхняя оценка $\mathsf{C}(\mathtt{INC}_n) \le 2n - 2$ при $n \ge 2$ получается элементарно. Если обозначить вход через $X = [x_{n-1}, \ldots, x_0] \in [\![2^n]\!]$, разряды суммы $X + 1 \bmod 2^n = [z_{n-1}, \ldots, z_0]$ определяются формулами[1]

$$z_k = x_k \oplus x_{k-1} \cdot \ldots \cdot x_0. \tag{2}$$

Произведения в этих формулах вычисляются оператором $\mathtt{PREF}^{\wedge}_{n-1}$, после чего выполняется поразрядное сложение векторов длины n по модулю 2. Использование параллельной префиксной схемы приводит к оценке $\mathsf{C}_{\log}(\mathtt{INC}_n) \le 3n - \Theta(\log n)$.

Дополняющую нижнюю оценку $\mathsf{C}(\mathtt{INC}_n) \ge 2n - 2$, вероятно, проще доказать, переходя к рассмотрению оператора $\mathtt{INC}'_n$, вычисляющего сумму $X + 1$ целиком. Легко проверить, что $\mathsf{C}(\mathtt{INC}'_n) = \mathsf{C}(\mathtt{INC}_n) + 1$ при $n \ge 2$. Далее, $\mathsf{C}(\mathtt{INC}'_n) = 2n - 1$, поскольку при подстановке $x_{n-1} = 0$ в схему для $\mathtt{INC}'_n$ сложность уменьшается минимум на 2, а полученная схема вычисляет оператор $\mathtt{INC}'_{n-1}$.

[1] Формула справедлива и при $k = 0$, если принять соглашение, что конъюнкция с пустым множеством операндов равна 1.

Схемы декремента, вычисляющие $X-1 \bmod 2^n = [z_{n-1},\ldots,z_0]$, устроены двойственным образом: разряды разности определяются формулами

$$z_k = x_k \oplus \overline{x_{k-1}} \cdot \ldots \cdot \overline{x_0}. \tag{3}$$

Двусторонний счетчик. Оператор $\mathtt{UDC}_n : [\![2^n]\!] \times \mathbb{B} \to [\![2^n]\!]$ в зависимости от управляющего входа $\sigma \in \mathbb{B}$ выполняет либо инкремент (при $\sigma = 1$), либо декремент (при $\sigma = 0$) n-разрядного числа по модулю 2^n.

Оценки $\mathsf{C}(\mathtt{UDC}_n) \le 3n-3$ при $n \ge 2$ и $\mathsf{C}_{\log}(\mathtt{UDC}_n) \le 4n - \Theta(\log n)$ получаются совмещением схем инкрементора и декрементора из предыдущего пункта. Разряды результата вычисляются по формулам, объединяющим (2) и (3):

$$z_k = x_k \oplus x_{k-1}^{\sigma} \cdot \ldots \cdot x_0^{\sigma}.$$

Булевы степени x_k^{σ} вычисляются просто как $\sigma \sim x_k$, $k = 0,\ldots,n-2$. Далее используется оператор $\mathtt{PREF}^{\wedge}_{n-1}$, затем выполняется поразрядное сложение векторов длины n.

Счетчик Грея. Оператор $\mathtt{GRC}_n : \mathbb{B}^n \to \mathbb{B}^n$ по булевой строке длины n вычисляет следующую за ней циклически в стандартной кодировке Грея. Иначе говоря, это инкрементор в коде Грея. Разряды строк в этом пункте нумеруются справа налево.

Напомним, что последовательность n-разрядных строк Грея G_n можно задать рекурсивно: $G_1 = (0,\,1)$ и далее $G_{k+1} = (0 \,\|\, G_k,\, 1 \,\|\, G_k^R)$, где G_k^R — последовательность G_k, записанная в обратном порядке; операция конкатенации применяется построчно. Основное свойство последовательностей Грея: следующая строка отличается от предыдущей ровно в одной позиции. Подробнее см., например, в [10, Гл. 13].

Верхние оценки $\mathsf{C}(\mathtt{GRC}_n) \le 4n-7$ и $\mathsf{C}_{\log}(\mathtt{GRC}_n) \le 6n - \Theta(\log n)$ получаются переводом в двоичный номер строки и обратно. Пусть $\mathrm{bin}_n(X)$ означает двоичную запись номера строки X в последовательности G_n. Тогда $\mathtt{GRC}_n(X) = \mathrm{bin}_n^{-1}(\mathtt{INC}_n(\mathrm{bin}_n(X)))$. Теперь, обозначая $[y_{n-1},\ldots,y_0] = \mathrm{bin}_n(x_{n-1},\ldots,x_0)$, легко установить, что $y_{n-1} = x_{n-1}$ и при любом $i \le n-2$

$$y_i = x_i \oplus x_{i+1} \oplus \ldots \oplus x_{n-1}, \qquad x_i = y_{i+1} \oplus y_i.$$

Поэтому преобразование bin_n выполняется оператором $\mathtt{PREF}^{\oplus}_n$, а bin_n^{-1} сводится к поразрядному сложению строк длины $n-1$. Чуть удобнее вместо y_i вычислять дополняющие биты $\overline{y_i}$. При последовательной реализации оператора $\mathtt{INC}_n(X)$ два младших разряда вычислять не нужно — они равны x_0 и $\overline{y_0}$ — остальная часть схемы инкремента имеет сложность $2n-4$. Еще одна операция экономится на последнем шаге преобразования в код Грея: младший разряд результата просто равен $\overline{y_1}$.

Оператор переносов. Оператор $\mathtt{CAR}_n : (\mathbb{B}^n)^2 \to \mathbb{B}^n$ вычисляет систему функций переноса

$$c_1 = x_0, \qquad c_{i+1} = x_i \oplus y_i c_i, \quad i = 1, \ldots, n-1, \tag{4}$$

от булевых переменных $x_0, x_1, \ldots, x_{n-1}$ и $y_0, y_1, \ldots, y_{n-1}$.

Прямо из определения (4) очевидно следует $\mathsf{C}(\mathtt{CAR}_n) = 2n - 2$. Оценка $\mathsf{C}_{\log}(\mathtt{CAR}_n) \leq 5n - \Theta(\log n)$ доказывается сведением к вычислению системы префиксных сумм. Пусть бинарная операция $\star$ на множестве булевых векторов длины 2 определяется как $[a_1, b_1] \star [a_2, b_2] = [a_2 \oplus b_2 a_1, b_2 b_1]$. Несложно убедиться, что введенная операция ассоциативна, значит, $(\mathbb{B}^2, \star)$ — полугруппа. С обозначением $p_i = y_{i-1} \cdot \ldots \cdot y_1$ система (4) трансформируется в систему префиксных сумм

$$[c_1, p_1] = [x_0, 1], \qquad [c_{i+1}, p_{i+1}] = [c_i, p_i] \star [x_i, y_i], \quad i = 1, \ldots, n-1.$$

Заметим, что собственно произведения p_i при реализации переносов вычислять не нужно. Тогда для сложности параллельных схем переносов имеем оценку $\mathsf{C}_{\log}(\mathtt{CAR}_n) \leq \mathsf{C}_{\log}(\mathtt{PREF}^{\star}_n) - (n-1)$. Остается заметить, что сложность операции $\star$ равна 3.

Идея вычисления переносов параллельными префиксными схемами возникла не позднее 1960-х гг., например, в [5, 25].

Сумматор. Оператор $\mathtt{ADD}_n : [\![2^n]\!]^2 \to [\![2^{n+1}]\!]$ вычисляет сумму двух n-разрядных чисел A и B.

Верхние оценки $\mathsf{C}(\mathtt{ADD}_n) \leq 5n - 3$ и $\mathsf{C}_{\log}(\mathtt{ADD}_n) \leq 8n - \Theta(\log n)$ получаются добавлением к схемам переносов из предыдущего пункта двух слоев из $3n - 1$ элементов[2]. Введем обозначения $A = [a_{n-1}, \ldots, a_0]$, $B = [b_{n-1}, \ldots, b_0]$ и $A + B = [z_n, \ldots, z_0]$. Положим $x_i = a_i \wedge b_i$, $y_i = a_i \oplus b_i$. Пусть c_i определяются согласно (4). Тогда $z_0 = y_0$, $z_n = c_n$ и $z_i = y_i \oplus c_i$ при прочих i.

Н. П. Редькин [6] показал, что первая из двух оценок точна: в действительности, $\mathsf{C}(\mathtt{ADD}_n) = 5n - 3$.

Мы не рассматриваем отдельно операцию вычитания чисел, поскольку отрицательные числа обычно задаются дополнительным кодом, в котором вычитание и сложение выполняются единообразно при помощи сумматоров, см., например, [2, разд. 4.1], [28, §3.1].

Компаратор. Функция $\mathtt{CMP}_n : [\![2^n]\!]^2 \to \mathbb{B}$ выполняет сравнение двух n-разрядных чисел A, B, т. е. вычисляет значение предиката $A > B$.

Верхняя оценка $\mathsf{C}(\mathtt{CMP}_n) \leq 4n - 3$ получается непосредственно. Обозначим $A = [a_{n-1}, \ldots, a_0]$, $B = [b_{n-1}, \ldots, b_0]$, а также $x_i = a_i \wedge \overline{b_i}$,

[2] В работах [19, 28] оценки сложности параллельного сумматора приводятся в форме $\mathsf{C}_{\log}(\mathtt{ADD}_n) \leq 8n + O(1)$.

$y_i = a_i \sim b_i$. Тогда $\mathtt{CMP}_n(A,B) = c_n$, где c_n определяется согласно (4). Поэтому $\mathsf{C}(\mathtt{CMP}_n) \le 2n-1+\mathsf{C}(\mathtt{CAR}_n)$, учитывая, что y_0 вычислять не надо.

Аналогично устанавливается оценка $\mathsf{C}_{\log}(\mathtt{CMP}_n) \le 2n-1+\mathsf{C}_{\log}(c_n) \le 5n-\Theta(\log n)$. Вычислим c_n при помощи параллельного префиксного дерева глубины d за $n-1$ операций $\star$, при этом d элементов, вычисляющих вторые компоненты префиксных сумм (произведения входных переменных), можно сократить.

Нестрогое сравнение реализуется подобным образом, поскольку предикат $A \ge B$ является отрицанием предиката $B > A$.

Расширенный оператор $\mathtt{CMP}^*_n : [\![2^n]\!]^2 \to \mathbb{B}^2$ дополнительно вычисляет признак $A = B$, т. е. произведение $y_0 \cdot y_1 \cdot \ldots \cdot y_{n-1}$. Его сложность оценивается как $\mathsf{C}_{\log}(\mathtt{CMP}^*_n) \le 5n-3$: в описанную выше параллельную схему компаратора надо добавить вычисление y_0 и всех вторых компонент префиксных сумм — из этих компонент в итоге и получается искомое произведение.

Максимум двух чисел. Оператор $\mathtt{MAX}_n : [\![2^n]\!]^2 \to [\![2^n]\!]$ вычисляет максимум из двух n-разрядных чисел.

Верхние оценки $\mathsf{C}(\mathtt{MAX}_n) \le 6n-3$ и $\mathsf{C}_{\log}(\mathtt{MAX}_n) \le 7n-\Theta(\log n)$ получаются присоединением к схемам компаратора из предыдущего пункта слоя из $2n$ булевых операций. Если обозначить $c = \mathtt{CMP}_n(A,B)$, то $\max(A,B) = \overline{(A \sim B)} \cdot [c]^n \oplus B$, где операции в последней формуле — поразрядные. Напомним, что вектор $A \sim B$, за исключением его младшего разряда, уже вычислен схемой компаратора. Поэтому для определения каждого разряда результата достаточно дополнительно выполнить по две операции, при этом старший разряд максимума просто равен $a_{n-1} \vee b_{n-1}$.

Для того, чтобы вместе с максимумом вычислить минимум, достаточно добавить в схему еще n элементов, поскольку $\min(A,B) = \overline{(A \sim B)} \cdot [c]^n \oplus A$.

Дешифратор. Оператор $\mathtt{DEC}_n : [\![n]\!] \to \mathbb{B}^n$ реализует булеву строку длины n с единственной единицей на заданной позиции. Компоненты оператора — элементарные конъюнкции X^α набора переменных X, где вектор α пробегает множество двоичных представлений чисел от 0 до $n-1$.

Верхняя оценка $\mathsf{C}_{\log}(\mathtt{DEC}_n) \le n+\Theta(\sqrt{n})$ получается тривиально делением множества переменных пополам: если $X = [X_2, X_1]$, то $X^{\alpha_2 \,\|\, \alpha_1} = X_2^{\alpha_2} \cdot X_1^{\alpha_1}$. Тогда $\mathsf{C}_{\log}(\mathtt{DEC}_n) \le n+\mathsf{C}_{\log}(\mathtt{DEC}_{2^k})+\mathsf{C}_{\log}(\mathtt{DEC}_{\lceil n2^{-k} \rceil})$, где $k = |X_1|$. Остается выбрать $k \approx \log n/2$.

Нижнюю оценку в форме $\mathsf{C}(\mathtt{DEC}_n) \ge n+\Theta(\sqrt{n})$ легко установить, заметив, что множество элементов схемы, непосредственно предшествующих выходам, имеет мощность не менее $\sqrt{n}$.

Расширенный оператор $\mathtt{DEC}^*_n : [\![n]\!] \times \mathbb{B} \to \mathbb{B}^n$ имеет дополнительный информационный вход $y \in \mathbb{B}$ и вычисляет вектор с битом y в заданной позиции и остальными нулями. Такой оператор часто называется демультиплексором. Его сложность отличается от сложности дешифратора не более чем на 2: достаточно в схеме дешифратора вместо младшей переменной x и ее отрицания использовать соответственно xy и $\overline{x} \cdot y$.

Мультиплексоры. Функция $\mathtt{MUX}_n : [\![n]\!] \times \mathbb{B}^n \to \mathbb{B}$ реализует выбор одной из n информационных переменных по ее номеру (иначе называемому адресом). Эту функцию называют $(n, 1)$-мультиплексором, а также селектором, функцией выбора или функцией обращения к памяти.

Лучшая известная верхняя оценка $\mathsf{C}_{\log}(\mathtt{MUX}_n) \leq 2n + O(\sqrt{n})$ получена П. Клейном и М. Патерсоном в [18]. Адресный вход представим как $X = [X_2, X_1]$, где $|X_1| = q$. Строку информационных переменных разобьем на блоки длины 2^q: $Y = Y_0 \,\|\, Y_1 \,\|\, \ldots \,\|\, Y_{p-1}$, $p = \lceil n/2^q \rceil$ (последний блок может быть короче). Разложим функцию $\mathtt{MUX}_n$ по переменным X_2:

$$\mathtt{MUX}_n(X; Y) = \bigvee_{\alpha=0}^{p-1} X_2^\alpha \cdot \mathtt{MUX}_{|Y_\alpha|}(X_1; Y_\alpha). \tag{5}$$

Внутренние мультиплексорные функции вычисляются по формулам

$$\mathtt{MUX}_{|Y_\alpha|}(X_1; Y_\alpha) = \bigvee_{\beta=0}^{|Y_\alpha|-1} y_{\alpha,\beta} \cdot X_1^\beta, \tag{6}$$

где подразумевается введение нумерации с двумя индексами на множестве переменных $Y = \{y_{\alpha,\beta}\}$.

Все элементарные конъюнкции групп переменных X_1 и X_2 вычисляются со сложностью порядка $2^q + p$ при помощи дешифраторов $\mathtt{DEC}_{2^q}$ и $\mathtt{DEC}_p$. Еще $2n + p$ операций достаточно для завершения вычислений по формулам (5), (6). Остается выбрать $q \approx \log n/2$.

Указанная оценка асимптотически точна: В. Пауль [23] установил, что $\mathsf{C}(\mathtt{MUX}_n) \geq 2n - 2$.

Очень часто возникает более общая задача — реализация (n, k)-мультиплексора $\mathtt{MUX}^k_n : [\![n]\!] \times (\mathbb{B}^k)^n \to \mathbb{B}^k$, информационными входами которого являются k-разрядные числа или булевы векторы. Схема для оператора $\mathtt{MUX}^k_n$ получается параллельным объединением описанных выше схем $(n, 1)$-мультиплексоров для каждого из k разрядов. Эти схемы имеют общую часть $\mathtt{DEC}_{2^q}(X_1)$ и $\mathtt{DEC}_p(X_2)$. При $q = \lceil \min(\log n, \log(kn)/2) \rceil$ получается оценка $\mathsf{C}_{\log}(\mathtt{MUX}^k_n) \leq 2kn + O(\sqrt{kn})$.

Операторы сдвига. Оператор $\mathtt{CYC}_{k,n} : [\![k]\!] \times \mathbb{B}^n \to \mathbb{B}^n$ выполняет циклический сдвиг булева вектора длины n на $X \in [\![k]\!]$ позиций[3].

[3]Направление сдвига для оценок сложности несущественно.

Верхняя оценка $\mathsf{C}_{\log}(\mathtt{CYC}_{k,n}) \le 3\lceil\log k\rceil n$ тривиальна. Соответствующая схема имеет вид l-кратной композиции $\mathtt{MUX}_2^n \circ \ldots \circ \mathtt{MUX}_2^n$, где $l = \lceil\log k\rceil$. Запишем $X = [x_{l-1}, \ldots, x_1, x_0]$. Первая подсхема выполняет циклический сдвиг на x_0, вторая — на $2x_1$, третья — на $4x_2$, и т. д.

Аналогично реализуется оператор обычного сдвига $\mathtt{SFT}_{k,n} : [\![k]\!] \times \mathbb{B}^n \to \mathbb{B}^{n+k-1}$. Справедлива оценка $\mathsf{C}_{\log}(\mathtt{SFT}_{k,n}) \le 3\lceil\log k\rceil n - 2(k-1)$, учитывающая, что $k-1$ штук (2, 1)-мультиплексоров в описанной выше схеме упрощаются до одной операции.

Шифратор. Оператор $\mathtt{ENC}_n : \mathbb{B}^n \to [\![n]\!]$ — это линейный булев оператор с матрицей U_n размера $\lceil\log n\rceil \times n$, в столбцах которой последовательно записаны числа от 0 до $n-1$ в двоичном представлении[4]. У входного вектора веса 1 шифратор определяет номер единичной позиции. Поэтому на множестве векторов веса 1 шифратор $\mathtt{ENC}_n$ является обратным преобразованием к $\mathtt{DEC}_n$.

Известно, что $\mathsf{C}(\mathtt{ENC}_n) = \mathsf{C}_{\log}(\mathtt{ENC}_n) = 2(n - \lceil\log n\rceil - 1)$. Нижняя оценка доказана А. В. Чашкиным [12]. Верхнюю оценку можно получить элементарно, используя известное соотношение между сложностью линейных операторов с транспонированными матрицами (принцип транспонирования [4]): сложность оператора с матрицей U_n^T тривиально равна $n - \lceil\log n\rceil - 1$. Однако нетрудно и явно описать конструкцию соответствующих схем. Семейство схем Φ_n, вычисляющих $\mathtt{ENC}_n$, построим рекурсивно вместе со схемами Φ'_n, которые реализуют преобразования $\mathtt{ENC}'_n$ с матрицами U_n, дополненными строками из всех единиц. Пусть $n = 2^k + p \le 2^{k+1}$. Разобьем вектор X длины n на две части X_1, X_2 длины 2^k и p. Результат $\Phi_n(X)$ получается из $\Phi_{2^k}(X_1)$ и $\Phi'_p(X_2)$ при помощи дополнительных $\lceil\log p\rceil$ булевых операций сложения, а $\Phi'_n(X)$ получается из $\Phi'_{2^k}(X_1)$ и $\Phi'_p(X_2)$ при помощи $\lceil\log p\rceil + 1$ операций. Рекуррентные соотношения

$$\mathsf{C}(\Phi_n) = \mathsf{C}(\Phi_{2^k}) + \mathsf{C}(\Phi'_p) + \lceil\log p\rceil, \quad \mathsf{C}(\Phi'_n) = \mathsf{C}(\Phi'_{2^k}) + \mathsf{C}(\Phi'_p) + \lceil\log p\rceil + 1$$

разрешаются как $\mathsf{C}(\Phi_n) = 2(n - \lceil\log n\rceil - 1)$ и $\mathsf{C}(\Phi'_n) = 2n - \lceil\log n\rceil - 2$ с учетом начальных условий $\mathsf{C}(\Phi_1) = \mathsf{C}(\Phi'_1) = 0$. Глубина схем Φ_n и Φ'_n равна $\lceil\log n\rceil - 1$ и $\lceil\log n\rceil$ соответственно. Подробнее см. в [13].

В альтернативном варианте определения, шифратор $\mathtt{ENC}^*_n$ — это частичный булев оператор, определенный только на множестве векторов веса 1 и совпадающий с $\mathtt{ENC}_n$ на этом множестве. Следуя схеме доказательства нижней оценки из [13], несложно проверить, что ослабление определения не дает существенного выигрыша: $\mathsf{C}(\mathtt{ENC}^*_n) = 2n - \Theta(\log n)$.

[4]При $n = 2^k$ указанная матрица является с точностью до наличия нулевого столбца проверочной матрицей кода Хэмминга.

Унарная кодировка. Оператор $\mathtt{UN}_n : [\![n+1]\!] \to \mathbb{B}^n$ переводит число из стандартной двоичной записи в унарную — в ней число k записывается строкой, начинающейся с k единиц и продолжаемой нулями.

Верхнюю оценку $\mathsf{C}(\mathtt{UN}_n) \le 2n + O(\sqrt{n})$ просто получить при помощи схемы типа $\mathtt{PREF}^{\vee} \circ \mathtt{DEC}$. Оператор $\mathtt{DEC}_{n+1}(x)$ вычисляет вектор с одной единицей в позиции x. Отбрасывая младшую компоненту вектора, вычисляем суффиксные суммы оставшихся компонент.

Указанная оценка усиливается до $\mathsf{C}_{\log}(\mathtt{UN}_n) \le 2n + O(\sqrt{n})$. Сначала индуктивно построим схемы для $\mathtt{UN}_{2^k-1}$ сложности $2(2^k - k - 1)$ и глубины $k-1$. Пусть булева строка $S = [s_1, \ldots, s_{2^k-1}]$ получается как $S = \mathtt{UN}_{2^k-1}(X)$, где X — вектор булевых переменных длины k. Тогда

$$\mathtt{UN}_{2^{k+1}-1}(y, X) = \begin{cases} S \parallel [0]^{2^k}, & y = 0 \\ [1]^{2^k} \parallel S, & y = 1 \end{cases} = \\ = [s_1 \vee y,\ \ldots,\ s_{2^k-1} \vee y,\ y,\ s_1 \cdot y,\ \ldots,\ s_{2^k-1} \cdot y].$$

Таким образом, схема для $\mathtt{UN}_{2^{k+1}-1}$ строится из схемы для $\mathtt{UN}_{2^k-1}$ добавлением слоя из $2(2^k-1)$ элементов конъюнкции и дизъюнкции, откуда с учетом $\mathsf{C}(\mathtt{UN}_1) = 0$ получаются требуемые оценки.

Схему для $\mathtt{UN}_n$ построим блочным методом. Пусть $X = [X_2, X_1]$, где $|X_1| = k \approx \log n/2$. Обозначим $p = \lceil (n+1)/2^k \rceil$. Результат $\mathtt{UN}_n(X)$ можно записать в блочном виде как $B_0 \parallel B_1 \parallel \ldots \parallel B_{p-1}$, где все строки B_i, за исключением последней неполной[5], имеют длину 2^k. Вычислим $\mathtt{DEC}_p(X_2) = [a_0, a_1, \ldots, a_{p-1}]$ — это вектор с индикатором позиции первого не сплошь единичного блока. Такой блок имеет вид $S = \mathtt{UN}_{2^k-1}(X_1) \parallel 0$ (укороченный в случае крайнего блока), слева от него блоки сплошь из единиц, справа — сплошь из нулей. При помощи оператора $\mathtt{PREF}^{\vee}_p$ вычислим префиксные суммы $b_i = a_0 \vee a_1 \vee \ldots \vee a_i$. Тогда

$$B_i = \begin{cases} [1]^{2^k}, & a_i = b_i = 0 \\ S, & a_i = b_i = 1 \\ [0]^{2^k}, & a_i = 0, b_i = 1 \end{cases} = S \cdot [a_i]^{2^k} \vee [\overline{b_i}]^{2^k}, \qquad (7)$$

где в правой части операции с булевыми строками выполняются поразрядно; при $i = p-1$ блок урезан. Окончательно, схема состоит из подсхем $\mathtt{DEC}_p$, $\mathtt{PREF}_p$, $\mathtt{UN}_{2^k-1}$ сложности $O(\sqrt{n})$ и слоя из $2n$ операций, предусмотренных в (7).

[5] Ее длина — от 0 до $2^k - 1$.

Обратное преобразование $\mathtt{UN}_n^{-1} : \mathbb{B}^n \to [\![n+1]\!]$ из унарного кода в двоичный выполняется несложно: $\mathsf{C}_{\log}(\mathtt{UN}_n^{-1}) = n - 1$. Разряды числа $[y_k, \ldots, y_0] = \mathtt{UN}_n^{-1}(s_1, \ldots, s_n)$ определяются формулами

$$y_j = \bigoplus_{i \geq 1} s_{i \cdot 2^j}.$$

Все необходимые суммы можно вычислить деревом глубины $\lceil \log n \rceil$ из $n - 1$ элементов $\oplus$.

Оператор обрезки. Оператор $\mathtt{TRN}_n : [\![n+1]\!] \times \mathbb{B}^n \to \mathbb{B}^n$ в булевой строке длины n сохраняет первые k бит, а остаток строки заполняет нулями.

Верхняя оценка $\mathsf{C}_{\log}(\mathtt{TRN}_n) \leq 3n + O(\sqrt{n})$ получается автоматически ввиду $\mathtt{TRN}_n(k; X) = \mathtt{UN}_n(k) \wedge X$ (конъюнкция выполняется поразрядно).

Индикатор первой единицы. Оператор $\mathtt{FOI}_n : \mathbb{B}^n \to \mathbb{B}^n \times \mathbb{B}$ оставляет в булевой строке длины n самую первую единицу, заменяя остальные нулями, и дополнительно вычисляет признак наличия единицы в строке.

Легко видеть, что оператор преобразует строку $X = [x_0, x_1, \ldots, x_{n-1}]$ в $Y = [y_0, y_1, \ldots, y_{n-1}; z]$, где $y_k = \overline{(x_0 \vee \ldots \vee x_{k-1})} \cdot x_k$ и $z = x_0 \vee \ldots \vee x_{n-1}$. Для его вычисления достаточно к схеме, вычисляющей $\mathtt{PREF}_n^{\vee}(X)$, добавить слой из $n - 1$ булевых операций типа $\overline{a} \wedge b$. Как следствие, $\mathsf{C}(\mathtt{FOI}_n) \leq 2n - 2$ и $\mathsf{C}_{\log}(\mathtt{FOI}_n) \leq 3n - \Theta(\log n)$.

Первая оценка точна: несложно проверить, что схема, реализующая $\mathtt{FOI}_n$, после подстановки $x_{n-1} = 0$ вычисляет $\mathtt{FOI}_{n-1}$ и при этом в ней сокращаются минимум два элемента.

Приоритетный шифратор. Оператор $\mathtt{PENC}_n : \mathbb{B}^n \to [\![n]\!] \times \mathbb{B}$ определяет номер позиции первой единицы в булевой строке длины n и дополнительно вычисляет признак наличия единицы в строке[6].

Верхние оценки $\mathsf{C}(\mathtt{PENC}_n) \leq 2n - 2$ и $\mathsf{C}_{\log}(\mathtt{PENC}_n) \leq 3n - \Theta(\log n)$ достигаются схемами вида $\mathtt{UN}^{-1} \circ \mathtt{PREF}$. Действительно, номер первой единичной позиции строки X можно найти как $\mathtt{UN}_n^{-1}(\overline{\mathtt{PREF}_n^{\vee}(X)})$, где операция отрицания применяется поразрядно. Попутно внутренний оператор $\mathtt{PREF}_n^{\vee}$ вычисляет признак наличия единицы. При $n \geq 3$ справедливо даже $\mathsf{C}(\mathtt{PENC}_n) \leq 2n - 3$: номер позиции корректно вычисляется как $\mathtt{UN}_{n-1}^{-1}(\overline{\mathtt{PREF}_{n-1}^{\vee}(X')})$, где X' — строка X без последнего символа.

Нижнюю оценку можно указать в виде $\mathsf{C}(\mathtt{PENC}_n) \geq \mathsf{C}(\mathtt{ENC}_n^*) = 2n - \Theta(\log n)$, поскольку оператор $\mathtt{PENC}_n$ является доопределением оператора $\mathtt{ENC}_n^*$.

Суммирование битов. Оператор $\mathtt{SUM}_n : \mathbb{B}^n \to [\![n+1]\!]$ вычисляет арифметическую сумму n булевых переменных.

[6] Это частичный булев оператор: номер позиции не определен при нулевом входе.

Сложность суммирования битов достаточно хорошо изучена. Эффективные схемы суммирования битов строятся из подсхем-компрессоров. Компрессор преобразует несколько битовых входов в меньшее число выходов с сохранением суммы (с учетом разряда). Пример схемы суммирования, составленной из (3, 2)-компрессоров $\Sigma_{3,2}$, вычисляющих сумму трех бит, показан на рис. 2.

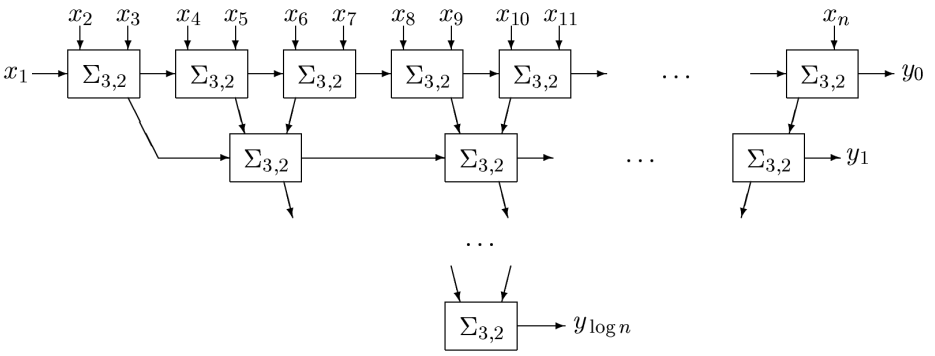

Рис. 2: Стандартная схема суммирования n бит

В работе [16] из специальных (5, 3)-компрессоров сконструирована схема сложности $4.5n - \Theta(\log n)$. Ценой увеличения сложности до $4.5n + o(n)$ схему можно сделать параллельной [7]. Для этого, например, можно разбить входы на группы по $\log n$ штук, вычислить суммы в группах методом [16], затем сложить групповые суммы при помощи любой параллельной схемы компрессоров. Первый этап имеет сложность $4.5n - o(n)$, а второй — $o(n)$. Таким образом, $\mathsf{C}(\mathtt{SUM}_n) \le 4.5n - \Theta(\log n)$ и $\mathsf{C}_{\log}(\mathtt{SUM}_n) \le 4.5n + o(n)$. О построении параллельных схем из компрессоров подробно рассказывается в [22], см. также [28, §3.2].

Нижнюю оценку можно указать в виде $\mathsf{C}(\mathtt{SUM}_n) \ge 2.5n + \Theta(\log n)$. Она получается сопоставлением доказанной Л. Стокмейером [26] оценки $2.5n - O(1)$, справедливой для каждой компоненты оператора $\mathtt{SUM}_n$, кроме разве что младшей и нескольких старших, и простого соотношения [20]

для сложности системы функций:

$$\mathsf{C}(f_1, \ldots, f_k) \geq \min_i \mathsf{C}(f_i) + k - 1. \tag{8}$$

Подсчет числа единиц в блоке. Оператор $\mathtt{BW}_n : \mathbb{B}^n \to [\![n+1]\!]$ для булевой строки длины n, которая содержит только один блок из единиц, вычисляет длину этого блока.

Верхние оценки $\mathsf{C}(\mathtt{BW}_n) \leq 4n - \Theta(\log n)$ и $\mathsf{C}_{\log}(\mathtt{BW}_n) \leq 4n + o(n)$ доказываются методом компрессоров. Базовая схема строится как на рис. 2 из подсхем-компрессоров $\Sigma_{3,2} : (a, b, c) \to [u, v]$, суммирующих тройки бит: $a + b + c = 2u + v$. При условии, что единицы на входе расположены одним сплошным блоком, такие подсхемы можно реализовать со сложностью 4: $u = b(a \vee c)$, $v = a \oplus b \oplus c$. Легко видеть, что если на вход схемы рис. 2 подается строка с одним единичным блоком, то единицы на входе каждого внутреннего компрессора будут смежны, иначе говоря, входной вектор (1, 0, 1) не встретится. Общая сложность описанной схемы равна $4n - \Theta(\log n)$. Параллельная схема строится как указано в предыдущем пункте.

Пороговые симметрические функции. Оператор $\mathtt{THR}_n^k : \mathbb{B}^n \to \mathbb{B}$ сравнивает число единиц в булевой строке X длины n с порогом k и вычисляет значение предиката $\nu(X) \geq k$.

Верхние оценки $\mathsf{C}(\mathtt{THR}_n^k) \leq 4.5n + O(\log n)$ и $\mathsf{C}_{\log}(\mathtt{THR}_n^k) \leq 4.5n + o(n)$ вытекают непосредственно из соответствующих оценок сложности оператора суммирования битов; к схеме для $\mathtt{SUM}_n$ достаточно присоединить схему сравнения суммы с порогом. Л. Стокмейер [26] доказал нижнюю оценку $\mathsf{C}(\mathtt{THR}_n^k) \geq 2n + \min\{k - 2, n - k - 1\} - 3$.

Аналогично строятся схемы для других симметрических функций, например, периодических.

Для постоянного порога k специальные методы синтеза позволяют получить лучшие оценки сложности. В частности, $\mathsf{C}(\mathtt{THR}_n^2) = \mathsf{C}_{\log}(\mathtt{THR}_n^2) = 2n + \Theta(\sqrt{n})$, $\mathsf{C}_{\log}(\mathtt{THR}_n^3) \leq 3n + O(\log n)$, а в общем виде для $2^{p-1} < k \leq 2^p$ имеет место $\mathsf{C}_{\log}(\mathtt{THR}_n^k) \leq (4.5 - 2^{2-p})n + \theta_k(n)$, где $\theta_k(n) = O(\sqrt{n})$ или $\theta_k(n) = O(\log n)$, см. [8].

Сортировка. Оператор $\mathtt{SORT}_n : \mathbb{B}^n \to \mathbb{B}^n$ упорядочивает строку битов в порядке возрастания (нули слева, единицы справа).

Верхние оценки $\mathsf{C}(\mathtt{SORT}_n) \leq 6.5n + O(\sqrt{n})$ и $\mathsf{C}_{\log}(\mathtt{SORT}_n) \leq 6.5n + o(n)$ достигаются на схемах типа $\mathtt{UN} \circ \mathtt{SUM}$. Сначала при помощи оператора $\mathtt{SUM}_n$ вычисляется число единиц в строке, затем посредством $\mathtt{UN}_n$ — унарное представление этого числа.

Нижняя оценка $\mathsf{C}_{\log}(\mathtt{SORT}_n) \geq 3n - 6$ получается сопоставлением доказанной Б. М. Клоссом [1] нижней оценки $2n - 3$, справедливой для

любой компоненты оператора, кроме младшей и старшей (минимума и максимума), и оценки (8).

Некоторые приложения

В этом разделе мы рассмотрим несколько примеров применения рассмотренных выше конструкций к построению параллельных схем более узкого назначения. Первый пример иллюстрирует применение параллельных префиксно-суффиксных схем. Второй пример опирается на эффективную реализацию унарно-бинарных преобразований. Третий пример иллюстрирует принцип массового производства и приводит к четвертому примеру.

2-селектор. Оператор $\mathtt{TOI}_n : \mathbb{B}^n \to \mathbb{B}^n \times \mathbb{B}$ выделяет в булевой строке длины n две единицы, заменяя остальные нулями, и дополнительно вычисляет признак наличия единицы в строке. Этот оператор является расширением оператора $\mathtt{FOI}_n$ и применяется для выделения в наборе из n каналов данных двух активных — отмеченных единицами, см., например, [14].

Оценка $\mathsf{C}_{\log}(\mathtt{TOI}_n) \le 5n - \Theta(\log n)$ достигается на схеме, выделяющей две крайних единицы с разных сторон. При помощи оператора $\mathtt{PS}_n^{\vee}$ по исходной строке $[x_1, \ldots, x_n]$ вычисляются строка префиксных сумм $[p_1, \ldots, p_n] = [0]^k[1]^{n-k}$ и строка суффиксных сумм $[s_1, \ldots, s_n] = [1]^l[0]^{n-l}$, где $k+1$ и l — позиции первой и последней единицы (если единицы есть). При этом $p_n = s_1$ служит признаком наличия единиц. Результирующая строка $[z_1, \ldots, z_n]$ вычисляется при помощи поразрядных операций $z_i = x_i \cdot (\overline{p_{i-1}} \vee \overline{s_{i+1}})$.

Счетчик с сохранением веса. Оператор $\mathtt{NCK}_n : [\![2^n]\!] \to [\![2^n]\!]$ вычисляет следующее за данным в порядке возрастания число с таким же двоичным весом. Если большего числа нет, то выход совпадает со входом. Этот оператор, в частности, рассматривался в [21] в связи с приложениями в области обработки изображений — там он назван $\binom{n}{k}$-счетчиком. Сложность параллельной реализации оператора $\mathtt{NCK}_n$ в [21] указана как $O(n)$, из описания схемы извлекается оценка около $16n$.

Верхнюю оценку сложности можно уточнить до $\mathsf{C}_{\log}(\mathtt{NCK}_n) \le 13n + o(n)$. Приписав к произвольному отличному от нуля n-разрядному числу дополнительный ноль слева, это число можно однозначно представить в виде $S \,\|\, 0\,[1]^j[0]^i$, где $i \ge 0$, $j \ge 1$ и S — непустая подстрока в случае, когда число не максимально среди чисел одинакового веса. Тогда оператор $\mathtt{NCK}_n$ преобразует немаксимальное число как $S \,\|\, 0\,[1]^j[0]^i \to S \,\|\, 1\,[0]^{i+1}[1]^{j-1}$. Последовательность вычислений, приводящих к требуемому результату, изображена на рис. 3. Через $*$ обозначены несущественные части строк, `bit[]` означает соответствующую поразрядную операцию. Аргументы и

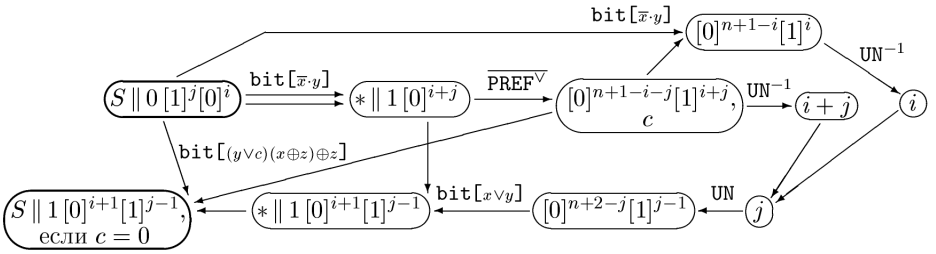

Рис. 3: Диаграмма алгоритма вычислений в схеме счетчика с сохранением веса

результаты некоторых операций сдвинуты на 1. Дополнительные биты показаны для общности записи, в вычислениях они не используются. Схема состоит из подсхем, реализующих операторы $\mathtt{PREF}^\vee_n$, $\mathtt{UN}_n$, $\mathtt{UN}^{-1}_n$ (дважды), семь поразрядных операций с векторами длины n и вычитание $\log n$-разрядных чисел. Признак c максимальности числа записывается как $(i+j=n)$; он получается на втором шаге как второй слева (т. е. старший значащий) бит числа $[0]^{n+1-i-j}[1]^{i+j}$. На заключительном шаге алгоритма при $c=1$ выбирается исходное число, при $c=0$ результат составляется из фрагмента S исходного числа и вычисленного фрагмента $1\,[0]^{i+1}[1]^{j-1}$. Схема корректно работает и с нулевым входом.

В практической схемотехнике условие $\overline{c}$ может использоваться отдельно как признак валидности результата (для записи его в регистр), тогда финальную операцию можно упростить до $\mathtt{bit}[xy\vee z\overline{y}]$. При этом сложность схемы понизится до $12n+o(n)$.

Оператор множественного выбора. Рассмотрим еще одно обобщение мультиплексора: оператор $\mathtt{SEL}^k_n : [\![n]\!]^k \times \mathbb{B}^n \to \mathbb{B}^k$ реализует выбор k из n информационных булевых переменных по их адресам. Метод Д. Улига (см. [27, 28]) позволяет строить более компактные схемы для множественного выбора в сравнении с простым объединением k независимых схем мультиплексоров. Для демонстрации идеи ограничимся случаем $k=2$.

Справедлива оценка $\mathsf{C}_{\log}(\mathtt{SEL}^2_n) \le 3n+O(n^{2/3})$. Обозначим $m=\lceil \log n \rceil$. Разобьем строку $Y \in \mathbb{B}^n$ на 2^r строк Y_i, $0 \le i < 2^r$, где строка $Y_i =$

$[y_i,\, y_{i+2^r},\, y_{i+2\cdot 2^r}, \ldots]$ включает переменные с индексами, равными i по модулю 2^r. По построению, $|Y_i| = q := \lceil n/2^r \rceil$ (более короткие строки дополним до длины q произвольно). Составим строки

$$\widetilde{Y}_0 = Y_0,\ \ \widetilde{Y}_1 = Y_0 \oplus Y_1,\ \ \ldots,\ \ \widetilde{Y}_i = Y_{i-1} \oplus Y_i,\ \ \ldots,$$
$$\widetilde{Y}_{2^r-1} = Y_{2^r-2} \oplus Y_{2^r-1},\ \ \widetilde{Y}_{2^r} = Y_{2^r-1}, \quad (9)$$

где операции сложения — поразрядные. Заметим, что при любом i

$$\widetilde{Y}_0 \oplus \ldots \oplus \widetilde{Y}_i = Y_i = \widetilde{Y}_{i+1} \oplus \ldots \oplus \widetilde{Y}_{2^r}.$$

Обозначим через $a = [a_1,\, a_0]$ и $b = [b_1,\, b_0]$ адресные входы схемы — в них выделены группы младших r разрядов: $|a_0| = |b_0| = r$.

Таким образом, если $a_0 \le b_0$, то можно определить

$$\mathtt{SEL}^2_n(a, b; Y) = [\mathtt{MUX}_n(a; Y),\, \mathtt{MUX}_n(b; Y)] = [\mathtt{MUX}_q(a_1; Y_{a_0}),\, \mathtt{MUX}_q(b_1; Y_{b_0})]$$

при помощи операторов $\mathtt{MUX}_q(z_i; \widetilde{Y}_i)$, полагая $z_i = a_1$ при $0 \le i \le a_0$ и $z_i = b_1$ при $b_0 < i \le 2^r$. В случае $a_0 > b_0$ переменные a и b меняются местами. В согласии с указанным правилом введем функции-индикаторы

$$\eta^a_i = (i \le a_0 \le b_0) \vee (i > a_0 > b_0), \quad \eta^b_i = (i > b_0 \ge a_0) \vee (i \le b_0 < a_0), \quad (10)$$

выбирающие, использовать подсхему $\mathtt{MUX}_q(z_i; \widetilde{Y}_i)$ для вычисления $\mathtt{MUX}_n(a; Y)$ или для вычисления $\mathtt{MUX}_n(b; Y)$. Окончательно результат определяется по формулам

$$\mathtt{MUX}_n(a; Y) = \bigoplus_{i=0}^{2^r} \eta^a_i\, \mathtt{MUX}_q(z_i; \widetilde{Y}_i), \quad \mathtt{MUX}_n(b; Y) = \bigoplus_{i=0}^{2^r} \eta^b_i\, \mathtt{MUX}_q(z_i; \widetilde{Y}_i),$$
$$z_i = [\eta^a_i]^{m-r} \cdot a_1 \vee [\eta^b_i]^{m-r} \cdot b_1, \quad (11)$$

где операции в последней формуле — поразрядные. Сложность схемы складывается из вычислений, предусмотренных (9), (10), (11), и оценивается как

$$\mathsf{C}_{\log}(\mathtt{SEL}^2_n) \le (2^r - 1)q + (2^r + 1)\mathsf{C}_{\log}(\mathtt{MUX}_q) + O(2^r m) \le$$
$$\le 3 \cdot 2^r q + O(2^r(m + \sqrt{q}) + q).$$

Требуемая оценка получается при выборе $r \approx \log n/3$. Метод практичен уже при $r = 1$.

На самом деле, линейная верхняя оценка сложности $\mathsf{C}_{\log}(\mathtt{SEL}^k_n) = O(n)$ справедлива для всех $k \le n/\log^3 n$, как показано в [17] весьма нетривиальным методом.

Перестановка пары битов. Оператор $\mathtt{EXC}_n : [\![n]\!]^2 \times \mathbb{B}^n \to \mathbb{B}^n$ выполняет перестановку двух бит с заданными адресами в булевой строке длины n. Эта довольно популярная операция (особенно в программировании) обсуждается, например, в [3, §7.1.3].

Опираясь на результат предыдущего пункта, несложно вывести оценку $\mathsf{C}_{\log}(\mathtt{EXC}_n) \leq 7n + O(n^{2/3})$. Решение также использует известный прием обмена содержимым между двумя регистрами, см. [10, Гл. 2]. Вычислим искомую пару битов $[u, v] = \mathtt{SEL}^2_n(a, b; Y)$. Затем при помощи двух демультиплексоров построим строки $\mathtt{DEC}^*_n(a; u \oplus v)$ и $\mathtt{DEC}^*_n(b; u \oplus v)$ с битами $u \oplus v$ в каждой из заданных позиций a, b. Поразрядно сложим эти строки друг с другом и с исходной строкой Y (по модулю 2).

Автор благодарен рецензенту за полезные замечания, в частности, за наблюдения, позволившие уточнить сложность преобразования из унарной кодировки в бинарную, приоритетного шифратора и 2-селектора.

Список литературы

[1] Клосс Б.М., Малышев В.А., “Оценки сложности некоторых классов функций”, *Вестник Моск. университета. Серия 1. Матем. Мех.*, 1965, № 4, 44–51.

[2] Кнут Д.Э., *Искусство программирования. Т.* 2. *Получисленные алгоритмы*, Вильямс, М., 2004, 832 с.

[3] Кнут Д.Э., *Искусство программирования. Т.* 4, *А. Комбинаторные алгоритмы, часть* 1, Диалектика, М., 2020, 960 с.

[4] Митягин Б.С., Садовский Б.Н., “О линейных булевских операторах”, *Доклады АН СССР*, **165**:4 (1965), 773–776.

[5] Офман Ю.П., “Алгоритмическая сложность дискретных функций”, *Доклады АН СССР*, **145**:1 (1962), 48–51.

[6] Редькин Н. П., “О минимальной реализации двоичного сумматора”, *Проблемы кибернетики. Вып.* 38, Наука, М., 1981, 181–216.

[7] Сергеев И.С., “Верхние оценки глубины симметрических булевых функций”, *Вестник Моск. университета. Серия 15. Выч. матем. и киберн.*, 2013, № 4, 39–44.

[8] Сергеев И.С., “О сложности монотонных схем для пороговых симметрических булевых функций”, *Дискретная математика*, **32**:1 (2020), 81–109. DOI: 10.4213/dm1547.

[9] Угрюмов Е.П., *Цифровая схемотехника*, БХВ-Петербург, СПб., 2010, 816 с.

[10] Уоррен Г.С., мл., *Алгоритмические трюки для программистов*, Вильямс, М., 2003, 288 с.

[11] Харрис Д.М., Харрис С.Л., *Цифровая схемотехника и архитектура компьютера*, ДМК Пресс, М., 2017, 792 с.

[12] Чашкин А.В., “О сложности булевых матриц, графов и соответствующих им булевых функций”, *Дискретная математика*, **6**:2 (1994), 43–73.

[13] Чашкин А.В., *Дискретная математика*, Академия, М., 2012, 352 с.

[14] Ahn J.-S., Jeong D.-K., Yang M., "Fast three-dimensional programmable two-selector", *Electronics Letters*, **40**:18 (2004), 1098–1100. DOI: 10.1049/el:20045935

[15] Blelloch G.E., "Prefix sums and their applications", *Synthesis of parallel algorithms*, Morgan Kaufmann, San Francisco, 1993, 35–60

[16] Demenkov E., Kojevnikov A., Kulikov A., Yaroslavtsev G., "New upper bounds on the Boolean circuit complexity of symmetric functions", *Inform. Proc. Letters*, **110**:7 (2010), 264–267. DOI: 10.1016/j.ipl.2010.01.007

[17] Holmgren J., Rothblum R., "Linear-size Boolean circuits for multiselection", *Proc. CCC (Ann Arbor, USA,* 2024*). LIPIcs*, **300**, 2024, 11:1–11:20. DOI: 10.4230/LIPIcs.CCC.2024.11

[18] Klein P., Paterson M.S., "Asymptotically optimal circuit for a storage access function", *IEEE Trans. on Computers*, **C-29**:8 (1980), 737–738. DOI: 10.1109/TC.1980.1675657

[19] Ladner R.E., Fischer M.J., "Parallel prefix computation", *J. ACM*, **27**:4 (1980), 831–838. DOI: 10.1145/322217.322232

[20] Lamagna E.A., Savage J.E., "On the logical complexity of symmetric switching functions in monotone and complete bases", *Tech. report*, Brown Univ., Rhode Island, 1973

[21] Nakano K., Yamagishi Y., "Hardware n choose k counters with applications to the partial exhaustive search", *IEICE Trans. on Information & Systems*, **E88-D**:7 (2005), 1350–1359. DOI: 10.1093/ietisy/e88-d.7.1350

[22] Paterson M.S., Pippenger N., Zwick U., "Optimal carry save networks", *LMS Lecture Notes Series. Boolean function complexity. Vol.* 169, Cambridge Univ. Press, Cambridge, 1992, 174–201. DOI: 10.1017/CBO9780511526633.014

[23] Paul W.J., "A $2.5n$-lower bound on the combinational complexity of Boolean functions", *SIAM J. Comput.*, **6**:3 (1977), 427–443. [Перевод: Пауль В.Й., "Нижняя оценка $2.5n$ для комбинационной сложности булевых функций", *Кибернетический сборник. Вып.* 16, Мир, М., 1979, 23–44.]. DOI: 10.1145/800116.803750

[24] Sergeev I.S., "On the complexity of parallel prefix circuits", *Electr. Colloq. on Comput. Complexity, TR13–041*, 2013

[25] Sklansky J., "Conditional-sum addition logic", *IRE Trans. Electr. Comput*, **EC-9** (1960), 226–231. DOI: 10.1109/TEC.1960.5219822

[26] Stockmeyer L.J., "On the combinational complexity of certain symmetric Boolean functions", *Math. Systems Theory*, **10** (1977), 323–336 [Перевод: Стокмейер Л.Дж., "О комбинационной сложности некоторых симметрических булевых функций", *Кибернетический сборник. Вып.* 16, Мир, М., 1979, 45–61.]. DOI: 10.1007/BF01683282

[27] Uhlig D., "Networks computing Boolean functions for multiple input values", *LMS Lecture Notes Series. Boolean function complexity. Vol.* 169, Cambridge Univ. Press, Cambridge, 1992, 165–173. DOI: 10.1017/CBO9780511526633.013

[28] Wegener I., *The complexity of Boolean functions*, Wiley–Teubner, Stuttgart, 1987, 470 с.

Статья поступила 12 января 2026 г.

Complexity of basic boolean operators for digital circuit design

I. S. Sergeev

This article provides a survey of circuit complexity bounds for basic boolean transforms exploited in digital circuit design and efficient methods for synthesizing such circuits. The exposition covers structurally simple functions and operators, such as counters, adders, encoders, and multiplexors, and excludes more complex algebraic operations with numbers, polynomials, and matrices. Several applications to implementing more specific operations are also discussed.

Keywords: boolean circuits, complexity, depth, parallel circuits, prefix circuits, incrementor, up-down counter, Gray counter, adder, comparator, decoder, multiplexor, encoder, compressor, threshold symmetric functions, priority encoder, unary coding, sorting.

References

[1] Kloss B.M., Malyshev V.A., "Otsenki slozhnosti nekotorykh klassov funktsii [Complexity bounds for some classes of functions]", *Vestnik Mosk. Univ. Ser. 1. Mat. Mekh.*, 1965, № 4, 44–51 (In Russian)

[2] Knuth D.E., *The art of computer programming. Vol.* 2. *Seminumerical algorithms*, Addison-Wesley, Reading, 1997, 762 pp.

[3] Knuth D.E., *The art of computer programming. Vol.* 4*A*. *Combinatorial algorithms, part* 1, Addison-Wesley, Upper Saddle River, 2011, 883 pp.

[4] Mitiagin B.S., Sadovskii B.N., "O lineinykh bulevskikh operatorakh [On linear Boolean operators]", *Dokl. AN SSSR*, **165**:4 (1965), 773–776 (In Russian)

[5] Ofman Y.P., "On the algorithmic complexity of discrete functions", *Soviet Phys. Dokl.*, **7** (1963), 589–591

[6] Red'kin N.P., "O minimal'noi realizatsii dvoichnogo summatora [On the minimal implementation of a binary adder]", *Problemy Kibernetiki. Vol.* 38, Nauka, Moscow, 1981, 181–216 (In Russian)

[7] Sergeev I.S., "Upper bounds on the depth of symmetric Boolean functions", *Moscow Univ. Comput. Math. and Cybern.*, **37**:4 (2013), 195–201. DOI: 10.3103/S0278641913040080

[8] Sergeev I.S., "On the complexity of monotone circuits for threshold symmetric Boolean functions", *Discrete Math. and Appl.*, **31**:5 (2021), 345–366. DOI: 10.1515/dma-2021-0031

[9] Ugryumov E.P., *Tsifrovaya skhemotekhnika* [*Digital circuit design*], BHV-Peterburg, Saint Petersburg, 2010 (In Russian), 816 pp.

[10] Warren H.S., Jr., *Hacker's delight*, Addison-Wesley, Upper Saddle River, 2002, 494 pp.

[11] Harris D.M., Harris S.L., *Digital design and computer architecture*, Morgan Kaufmann, San Francisco, 2012, 712 pp.

[12] Chashkin A.V., "On the complexity of Boolean matrices, graphs and their corresponding Boolean functions", *Discrete Math. and Appl.*, **4**:3 (1994), 229–257. DOI: 10.1515/dma.1994.4.3.229

[13] Chashkin A.V., *Diskretnaya matematika* [*Discrete mathematics*], Akademiya, Moscow, 2012 (In Russian), 352 pp.

[14] Ahn J.-S., Jeong D.-K., Yang M., "Fast three-dimensional programmable two-selector", *Electronics Letters*, **40**:18 (2004), 1098–1100. DOI: 10.1049/el:20045935

[15] Blelloch G.E., "Prefix sums and their applications", *Synthesis of parallel algorithms*, Morgan Kaufmann, San Francisco, 1993, 35–60

[16] Demenkov E., Kojevnikov A., Kulikov A., Yaroslavtsev G., "New upper bounds on the Boolean circuit complexity of symmetric functions", *Inform. Proc. Letters*, **110**:7 (2010), 264–267. DOI: 10.1016/j.ipl.2010.01.007

[17] Holmgren J., Rothblum R., "Linear-size Boolean circuits for multiselection", *Proc. CCC (Ann Arbor, USA,* 2024*). LIPIcs*, **300**, 2024, 11:1–11:20. DOI: 10.4230/LIPIcs.CCC.2024.11

[18] Klein P., Paterson M.S., "Asymptotically optimal circuit for a storage access function", *IEEE Trans. on Computers*, **C-29**:8 (1980), 737–738. DOI: 10.1109/TC.1980.1675657

[19] Ladner R.E., Fischer M.J., "Parallel prefix computation", *J. ACM*, **27**:4 (1980), 831–838. DOI: 10.1145/322217.322232

[20] Lamagna E.A., Savage J.E., "On the logical complexity of symmetric switching functions in monotone and complete bases", *Tech. report*, Brown Univ., Rhode Island, 1973

[21] Nakano K., Yamagishi Y., "Hardware n choose k counters with applications to the partial exhaustive search", *IEICE Trans. on Information & Systems*, **E88-D**:7 (2005), 1350–1359. DOI: 10.1093/ietisy/e88-d.7.1350

[22] Paterson M.S., Pippenger N., Zwick U., "Optimal carry save networks", *LMS Lecture Notes Series. Boolean function complexity. Vol.* 169, Cambridge Univ. Press, Cambridge, 1992, 174–201. DOI: 10.1017/CBO9780511526633.014

[23] Paul W.J., "A $2.5n$-lower bound on the combinational complexity of Boolean functions", *SIAM J. Comput.*, **6**:3 (1977), 427–443. DOI: 10.1145/800116.803750

[24] Sergeev I.S., "On the complexity of parallel prefix circuits", *Electr. Colloq. on Comput. Complexity, TR13–041*, 2013

[25] Sklansky J., “Conditional-sum addition logic”, *IRE Trans. Electr. Comput*, **EC-9** (1960), 226–231. DOI: 10.1109/TEC.1960.5219822

[26] Stockmeyer L.J., “On the combinational complexity of certain symmetric Boolean functions”, *Math. Systems Theory*, **10** (1977), 323–336. DOI: 10.1007/BF01683282

[27] Uhlig D., “Networks computing Boolean functions for multiple input values”, *LMS Lecture Notes Series. Boolean function complexity. Vol.* 169, Cambridge Univ. Press, Cambridge, 1992, 165–173. DOI: 10.1017/CBO9780511526633.013

[28] Wegener I., *The complexity of Boolean functions*, Wiley–Teubner, Stuttgart, 1987, 470 pp.

Received on January 12, 2026

Таблица 1: Оценки сложности базовых операторов

опер. F	нижняя оценка $\mathsf{C}(F)$	верхняя оценка $\mathsf{C}(F)$	верхняя оценка $\mathsf{C}_{\log}(F)$
$\mathtt{PREF}_n$	$n-1$		$2n-\Theta(\log n)$ [5]
$\mathtt{PS}_n$	$2n-3$		$3n-\Theta(\log n)$
$\mathtt{INC}_n$	$2n-2$		$3n-\Theta(\log n)$
$\mathtt{UDC}_n$	—	$3n-3$	$4n-\Theta(\log n)$
$\mathtt{GRC}_n$	—	$4n-7$	$6n-\Theta(\log n)$
$\mathtt{CAR}_n$	$2n-2$		$5n-\Theta(\log n)$
$\mathtt{ADD}_n$	$5n-3$ [6]		$8n-\Theta(\log n)$
$\mathtt{CMP}_n$	—	$4n-3$	$5n-\Theta(\log n)$
$\mathtt{MAX}_n$	—	$6n-3$	$7n-\Theta(\log n)$
$\mathtt{DEC}_n$	$n+\Theta(\sqrt{n})$		
$\mathtt{MUX}_n$	$2n-2$ [23]	$2n+O(\sqrt{n})$ [18]	
$\mathtt{MUX}_n^k$	—	$2kn+O(\sqrt{kn})$	
$\mathtt{CYC}_{k,n}$	—	$3\lceil\log k\rceil n$	
$\mathtt{SFT}_{k,n}$	—	$3\lceil\log k\rceil n-\Theta(k)$	
$\mathtt{ENC}_n$	$2(n-\lceil\log n\rceil-1)$ [12]		
$\mathtt{UN}_n$	—	$2n+O(\sqrt{n})$	
$\mathtt{UN}_n^{-1}$	$n-1$		
$\mathtt{TRN}_n$	—	$3n+O(\sqrt{n})$	
$\mathtt{FOI}_n$	$2n-2$		$3n-\Theta(\log n)$
$\mathtt{PENC}_n$	$2n-\Theta(\log n)$	$2n-3$	$3n-\Theta(\log n)$
$\mathtt{SUM}_n$	$2.5n+\Theta(\log n)$ [20, 26]	$4.5n-\Theta(\log n)$ [16]	$4.5n+o(n)$ [7]
$\mathtt{THR}_n^k$	$2n+\min\{k,\, n-k\}-5$ [26]	$4.5n+O(\log n)$	$4.5n+o(n)$
$\mathtt{BW}_n$	—	$4n-\Theta(\log n)$	$4n+o(n)$
$\mathtt{SORT}_n$	$3n-6$ [1, 20]	$6.5n+O(\sqrt{n})$	$6.5n+o(n)$

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://intsysmagazine.ru/submit .
2. В преамбуле статьи должны содержаться название статьи на русском и английском языках, аннотация на русском и английском языках (не более 150 слов), список ключевых слов на русском и английском языках (не более 12 слов), информация об авторах: Ф.И.О. полностью, место работы, должность, ученая степень и/или звание (если имеется), для аспирантов ФИО научного руководителя, e-mail, ORCID.
3. Список литературы оформляется в едином формате, установленном системой Российского индекса научного цитирования. Списки на русском и английском языках приводятся в конце файла с текстом статьи, и должны быть оформлены с помощью пакета `amsbib`.
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. Авторам бесплатно предоставляется номер журнала, в котором вышла статья. Журнал распространяется по подписке, экземпляры журнала рассылаются подписчикам наложенным платежом. Условия подписки публикуются в каталоге НТИ «Роспечать», индекс журнала 64559.
5. Электронная версия журнала находится в открытом доступе. Все материалы распространяются на условиях лицензии Creative Commons Attribution 4.0 International (CC BY 4.0). Электронные версии всех выпущенных номеров размещаются на сайте

http://intsysmagazine.ru

и доступны для свободного чтения и скачивания. Авторы сохраняют авторские права на свои статьи.

Подписано в печать: 10.03.2026
Дата выхода: 25.03.2026
Тираж: 200 экз.
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).