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

Интеллектуальные Системы.

Теория и приложения

ТОМ **27** ВЫПУСК 2 * **2023**

МОСКВА

ISSN 2411–4448

УДК 519.95; 007:159.955
ББК 32.81

Издается с 1996 г.*

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

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

Учредитель журнала: ООО «Интеллектуальные системы».

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

Спонсором издания является:

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

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

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

*) Прежнее название журнала: «Интеллектуальные системы».

ОГЛАВЛЕНИЕ

Часть 1.
Общие проблемы теории интеллектуальных систем

Оценка и мониторинг алкогольного благополучия регионов на основе статистической информации

А. П. Рыжов[1], Б. Э. Горный[2], А. В. Зудин[3]

Для снижения рисков, связанных со злоупотреблением алкоголем, чрезвычайно важно оценивать уровни алкогольного потребления. При этом недостаточно знать показатели, оценивающие только объем розничных продаж, который не учитывает незарегистрированное потребление алкоголя. Разработка интегрального показателя индекса, базирующегося на доступных статистических данных, позволит нивелировать риски экспертного оценивания и повысить эффективность расходования бюджетных средств в области здравоохранения.

В докладе приводится описание доступной экспертной и фактической информации, структура индекса, предлагается подход к его оценке средствами теории нечетких множеств. Рассмотрены особенности реализации расчета индекса в среде Matlab, приводятся примеры. Формулируются и обсуждаются связанные с этим прямые и обратные задачи, решение которых позволит оптимизировать эффективность системы здравоохранения по данному параметру на региональном и федеральном уровнях. Индекс алкогольного благополучия является одним из важных параметров системы здравоохранения с одной стороны и типичным индексом социально-экономических процессов - с другой, поэтому описанные в докладе подходы могут быть применены и для построения широкого набора таких индексов.

Ключевые слова: профилактическая медицина, алкогольное благополучие, оценка и мониторинг процессов.

[1] *Рыжов Александр Павлович* — профессор каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: ryjov@mail.ru.

Ryzhov Alexander Pavlovich — professor, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

[2] *Горный Борис Эмануилович* — кандидат медицинских наук, ведущий научный сотрудник национального медицинского исследовательского центра терапии и профилактической медицины Минздрава РФ, e-mail: BGornyy@gnicpm.ru.

Gorny Boris Emanuilovich — Candidate of Medical Sciences, Leading Researcher at the National Medical Research Center for Therapy and Preventive Medicine of the Ministry of Health of the Russian Federation

[3] *Зудин Андрей Владимирович* — студент каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: Andrey_cu1@mail.ru.

Zudin Andrey Vladimirovich — student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

1. Введение

В современной России злоупотребление алкогольной продукцией приводит к заболеваемости, преждевременной смерти людей, росту преступности, насилия, сиротства и других социально-экономических и медико-демографических проблем. С потреблением алкоголя связано около 30% всех смертей в России, что в 5-9 раз выше аналогичного показателя в некоторых странах Европейского региона, Японии и США. Разработка и мониторинг эффективности региональных программ, направленных на снижение масштабов злоупотребления алкогольной продукцией и профилактику алкоголизма среди населения, невозможны без оценки ситуации в регионе.

Традиционно оценка алкогольной ситуации основывается на статистических показателях розничных продаж различных видов алкоголя. Но необходимо понимать, что этот показатель не отражает истинной ситуации в регионе, так как часто имеет место незарегистрированное потребление алкоголя. Поэтому судить об алкогольной ситуации опираясь только на данные розничных продаж, неверно [1, 2]. Авторами был предложен интегральный показатель, который объединял в себе как факторный признак – уровень потребления, так и результирующие признаки, наиболее тесно связанные с ним, такие как смертность, заболеваемость и ряд других демографических и социально-экономических индикаторов, потенциально связанных с употреблением алкоголя.

Предложенный подход продемонстрировал прогнозную значимость данного индекса, названного интегральным индексом алкогольной ситуации (ИИАС), показав региональные различия в смертности от ряда хронических неинфекционных заболеваний в зависимости от величины ИИАС [3]. Но несмотря на это, были выявлены определенные ограничения индекса, связанные с недоучетом значимости каждого из показателей в общей оценке.

В данной работе был реализован другой подход для расчета индекса, который учитывал не только величину самих показателей, но и их значимость для оценки ситуации. Он основан на технологии оценки и мониторинга сложных процессов [4], позволяющей на основе фрагментарной информации получать интегральную оценку состояния процесса и моделировать возможные варианты его развития. Целью данной работы является описание построения нового индекса и обсуждение вариантов его использования.

Текст работы организован следующим образом. В разделе 2 обсуждается содержательная постановка задачи и приводится обзор известных результатов. В разделе 3 представлены характеристика доступных данных и экспертных знаний. Раздел 4 посвящен собственно построению

индекса, использование которого обсуждается в разделе 5. В заключении приведены выводы и возможные направления дальнейших исследований.

Исследование выполнено при поддержке Междисциплинарной научно-образовательной школы Московского университета «Мозг, когнитивные системы, искусственный интеллект».

2. Содержательная постановка задачи и анализ литературы

Алкоголь является одним из значимых факторов риска заболеваемости, нетрудоспособности и смертности в мире. По оценкам ВОЗ, чрезмерное потребление алкоголя обусловило в 2012 году около 3.3 миллиона смертей, или 5.9% от их общего количества [5]. В России с избыточным потреблением алкоголя связано 66% тяжких правонарушений, 50% убийств, 40% разбойных нападений, 24% дорожно-транспортных правонарушений. Общие затраты государства (прямые медицинские, прямые немедицинские и косвенные), связанные со злоупотреблением алкоголем, в 2011 г. составили 843.51 млрд. руб. В год затраты на одного человека, злоупотребляющего алкоголем, составляют 150 тыс. 845 руб., а затраты, связанные с лечением, – 23 тыс. 813 руб. [6].

В рамках Глобальной стратегии сокращения вредного употребления алкоголя важное место принадлежит действиям, в реализации которых важна роль здравоохранения, в том числе речь идет о проведении мониторинга и эпиднадзора за потреблением алкоголя [7]. Если на страновом уровне эта работа частично ведется, и Россия предоставляет информацию в Global Health Observatory data repository, то на региональном и особенно на муниципальном уровне отсутствует система мониторинга за потреблением алкоголя. А у лиц, принимающих управленческие решения, отсутствует информация о проблемах, связанных с этим фактором риска. Существующие подходы оценки алкогольной ситуации, реализованные в ряде эпидемиологических исследований, ориентируются либо на уровень розничных продаж и результаты социологических исследований [8], либо на комплексный анализ косвенных индикаторов алкогольных проблем: уровень продажи алкоголя на душу населения, уровень связанной с алкоголем смертности, а также данные статистической отчетности наркологической службы [9]. В некоторых исследованиях этот перечень расширяется за счет данных статистики правоохранительных органов [10] или еще более широкого перечня показателей, дополнительно включающих ряд экономических и социальных индикаторов [11]. Найденные в исследовании Немцова и Разводовского [12] ста-

тистические связи между потреблением алкоголя и некоторыми медико-демографическими и социальными показателями, и даже полученные на их основе регрессионные уравнения, не могут использоваться для региональных оценок, так как основаны на российских данных. Кроме того, оценка потребления, основанная на розничных продажах алкоголя, в пересчете на этанол, которая чаще всего используется в регрессионных уравнениях, не отражает структуры потребления по типам алкогольных напитков. В статистику продаж в некоторых регионах существенный вклад вносит миграционные и туристические потоки.

Поэтому наиболее целесообразным представляется использование интегрального показателя, который бы объединял в себе как факторный признак – уровень потребления, так и результирующие признаки, наиболее тесно связанные с ним, такие как смертность, заболеваемость и ряд других демографических и социально-экономических индикаторов, потенциально связанных с употреблением алкоголя. Все вышеизложенное определяет актуальность построения индекса алкогольного благополучия - важного параметра для описания социальной обстановки в стране, регионе, городе. Руководствуясь значением индекса органы государственного управления регионов, муниципалитетов, городов смогут эффективнее распределять бюджетные средства на улучшение алкогольной ситуации в том или ином месте.

3. Особенности доступных данных и экспертизы

Работа была организована следующим образом. Из базы данных ЕМИСС [13] были получены статистические показатели, которые имели прямое и косвенное отношение к алкогольному благополучию:

- показатели розничной продажи алкогольных напитков: продажи пива, вина, крепкого алкоголя;
- социально-экономические показатели: преступления, совершенные в состоянии опьянение; уровень безработицы; число детей, оставшихся без попечения родителей; количество лиц, погибших в ДТП; доля населения с доходами ниже прожиточного минимума; число разводов;
- показатели заболеваемости: алкоголизмом, алкогольным психозом, циррозом печени, панкреатитом, туберкулезом;
- показатели смертности: общая смертность; смертность от внешних причин; смертность от цереброваскулярных заболеваний; смертность от болезней печени и поджелудочной железы.

Эта информация собирается на регулярной основе статистическими органами страны на региональном и федеральном уровне.

Кроме перечисленных статистических данных, использовались экспертные знания о влиянии перечисленных выше признаков на алкогольное благополучие. Был сформулирован набор правил, которые позволят определенным образом структурировать имеющиеся показатели. В докладе показано, как, располагая большим количеством косвенных данных и знаниями специалистов в предметной области, можно построить индекс алкогольного благополучия.

Отметим, что рассматриваемый индекс - пример большого числа подобных индексов, которые можно применять в экономических, социологических исследованиях и в государственном управлении. В дальнейшем приведенную методику можно использовать в разработке аналогичных индексов, для этого лишь необходимы статистические данные и эксперты, которые смогут их структурировать.

4. Построение индекса алкогольного благополучия

4.1. Структура индекса

Структура индекса представлена на рис. 1.

Рис.1 Структура индекса алкогольного благополучия.

Значения каждого признака из листьев дерева и каждого узла были разбиты на 4 нечетких множества, соответствующие по смыслу значениям: «Низкий», «Относительно низкий», «Относительно высокий», «Высокий». Набор значений определялся исходя из удобства формулировки

правил. Были опробованы несколько алгоритмов построения функций принадлежности, наилучший результат показал известный алгоритм c-means.

Помимо признаков, была сформулирована база правил, обеспечивающая связь между узлами дерева. Правила имеют форму «Если ..., то ...». База содержит 74 таких правила. Например, для узла 1 «Розничные продажи алкогольных напитков» в базе 19 правил:

Правило 1: Если «Продажи крепкого алкоголя» = низкий и «Продажи вина» = низкий и «Продажи пива» = низкий, то «Розничные продажи алкогольных напитков» = низкий;

Правило 2: Если «Продажи крепкого алкоголя» = относительно низкий и «Продажи вина» = низкий и «Продажи пива» = относительно низкий, то «Розничные продажи алкогольных напитков» = низкий;

Правило 3: Если «Продажи крепкого алкоголя» = относительно низкий и «Продажи вина» = относительно низкий и «Продажи пива» = относительно высокий, то «Розничные продажи алкогольных напитков» = относительно низкий;

...

Правило 19: Если «Продажи крепкого алкоголя» = высокий и «Продажи вина» = высокий и «Продажи пива» = высокий, то «Розничные продажи алкогольных напитков» = высокий.

4.2. Построение индекса

В качестве среды для построения индекса был выбран Matlab [14], содержащий пакет расширения Fuzzy Logic Toolbox - инструмент для проектирования систем нечеткой логики.

Согласно структуре индекса, нам необходимо для каждого листового показателя построить 4 функции принадлежности, соответствующие множествам «Низкий», «Относительно низкий», «Относительно высокий», «Высокий». Для этого используем известный метод нечеткой кластеризации C-means [15]. Суть метода состоит в минимизации суммарного квадратичного отклонения точек кластера от центров этих кластеров. При этом для каждого элемента из рассматриваемого множества рассчитывается степень его принадлежности каждому из кластеров. В Matlab кластеризация данным методом выполняется с помощью команды fcm.

Отсортируем значения каждого показателя, разобьем на 4 кластера. Пример такого разбиения для показателя 1.3 представлен на Рис. 2.

Таким же образом определяются функции принадлежности для всех входных переменных.

Далее, для каждого узла дерева создается система нечеткого вывода на основе правил с соответствующими входными и выходными пере-

```
load 'data.dat';
q=sort(data);
[center, U, obj_fcm] = fcm( q,4);
y1=U(1,:);
y2=U(2,:);
y3=U(3,:);
y4=U(4,:);
plot(q,y1);
hold on;
plot(q,y2);
hold on;
plot(q,y3);
hold on;
plot(q,y4);
hold on;
```

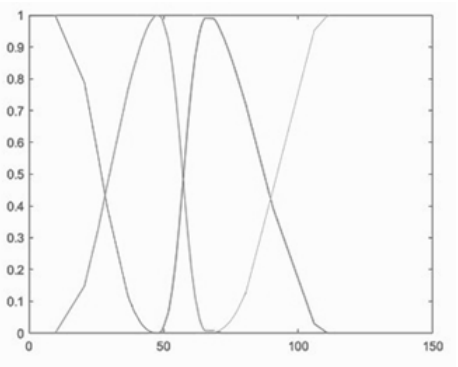


Рис. 2. Функции принадлежности для показателя 1.3.

менными. Рассмотрим в качестве примера узел 1 (Рис. 1). Для узла 1 «Розничные продажи алкогольных напитков» создаем три входных переменных «Продажи крепкого алкоголя», «Продажи вина», «Продажи пива» (Рис. 3).

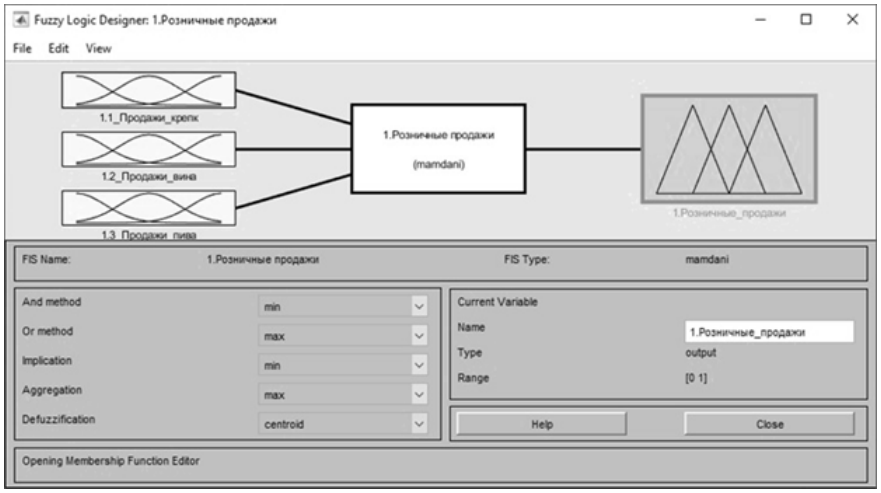


Рис. 3. Задание системы нечеткого вывода для узла 1.

Для каждой переменной создаем 4 нечетких множества, указав их функции принадлежности, найденные на предыдущем шаге (Рис. 2), и указываем область значения показателя (range) - минимальное и максимальное значение из таблицы со статистическими данными (Рис. 4). Далее переносим правила из базы в систему (Рис. 5).

Для промежуточных узлов (1, 2, 3, 3.1, 3.2) необходимо дополнительно определить универсальное множество и задать в нем функции принадлежности (для них нет статистических данных, как для листовых узлов). После реализации нескольких вариантов, был выбран следую-

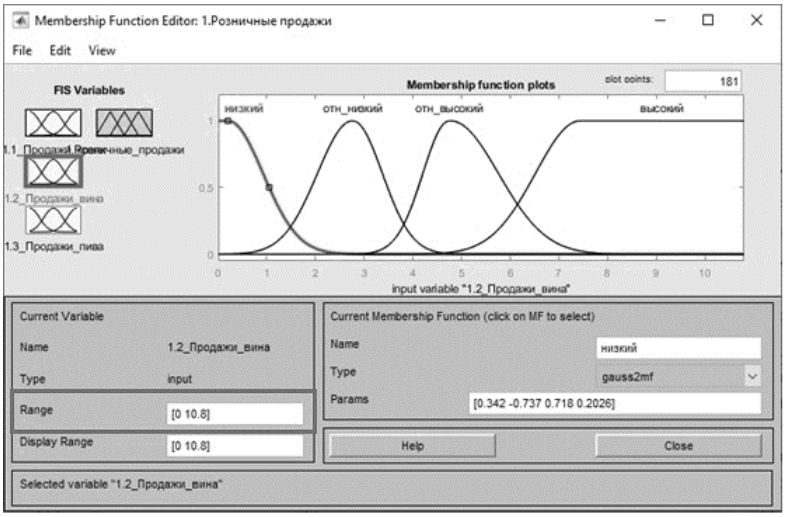


Рис. 4. Задание функций принадлежности входных переменных.

Рис. 5. Задание правил для узла 1.

щий. Запускается нечеткий вывод на всех значениях исходных данных, получающееся множество значений и образует универсальное множество для соответствующего узла. Далее в нем строятся функции принадлежности с помощью c-means, что позволяет применить описанную выше процедуру и для таких промежуточных узлов.

Результаты расчета по описанному алгоритму представлены на Рис. 6 (регионы упорядочены по индексу алкогольного благополучия). Положение регионов соответствует сложившимся в настоящее время представлениям экспертов.

Регион	Индекс алкогольного благополучия	Тульская область	0.653062186
		Забайкальский край	0.652878121
Республика Дагестан	0.886603419	Чувашская Республика - Чувашия	0.652518598
Москва	0.884745563	Пензенская область	0.652373333
Кабардино-Балкарская Республика	0.876961029	Брянская область	0.652350832
Республика Ингушетия	0.871673313	Смоленская область	0.652268842
Карачаево-Черкесская Республика	0.866385411	Республика Башкортостан	0.652208858
Чеченская Республика	0.859321036	Орловская область	0.652123882
Санкт-Петербург	0.859132897	Челябинская область	0.65208147
Краснодарский край	0.847668296	Кемеровская область	0.651914177
Белгородская область	0.843520876	Приморский край	0.651691516
Тюменская область	0.840311732	Красноярский край	0.651329892
Республика Северная Осетия - Алания	0.836739864	Ивановская область	0.651069198
Воронежская область	0.835080509	Владимирская область	0.650940962
Ханты-Мансийский автономный округ	0.824781344	Новгородская область	0.650887597
Томская область	0.818833488	Нижегородская область	0.6507979
Ставропольский край	0.808027209	Республика Марий Эл	0.650362666
Республика Татарстан	0.787586285	Пермский край	0.65021635
Московская область	0.782800301	Свердловская область	0.650180792
Тамбовская область	0.76308318	Оренбургская область	0.650156306
Волгоградская область	0.758493346	Ярославская область	0.650067261
Калининградская область	0.749125023	Калужская область	0.650011249
Кировская область	0.704999992	Хабаровский край	0.65000311
Удмуртская Республика	0.704997637	Республика Бурятия	0.650001651
Вологодская область	0.70499562	Республика Тыва	0.649932534
Архангельская область	0.70497662	Республика Калмыкия	0.647764849
Костромская область	0.699726063	Курганская область	0.646595094
Мурманская область	0.69932868	Иркутская область	0.646124936
Астраханская область	0.677584445	Республика Саха (Якутия)	0.620321759
Рязанская область	0.676536362	Псковская область	0.602394973
Ямало-Ненецкий автономный округ	0.673161861	Республика Хакасия	0.553864429
Саратовская область	0.659682209	Республика Адыгея (Адыгея)	0.34865473
Ростовская область	0.658566096	Магаданская область	0.339745034
Республика Мордовия	0.658524446	Ленинградская область	0.336522251
Новосибирская область	0.654830448	Амурская область	0.335027554
Самарская область	0.654633543	Республика Алтай	0.332954949
Тверская область	0.654514153	Сахалинская область	0.331147497
Курская область	0.654054138	Республика Карелия	0.315055653
Алтайский край	0.653965617	Ненецкий автономный округ	0.306980891
Ульяновская область	0.65391944	Республика Коми	0.273529626
Омская область	0.653723209	Камчатская край	0.268514946
Липецкая область	0.653651406	Еврейская автономная область	0.268099328
Тульская область	0.653062186	Чукотский автономный округ	0.115328549

Рис. 6. Расположение регионов в порядке убывания индекса.

5. Сценарий использования индекса

Построенный индекс может быть использован для моделирования ситуаций в области нашего интереса. Например, мы можем представить, что изменится один или нескольких косвенных признаков. Повлечет ли это изменение существенные изменения индекса или нет? Однозначного ответа нет, но, скорее всего, в разных регионах влияние изменяемых признаков будет разным - в зависимости от значений остальных параметров.

Рассмотрим два региона: Республику Адыгея с индексом 0.349 и Алтайский край с индексом 0.654. В этих регионах высокий показатель заболеваемости циррозом печени 140.2 и 196.1 соответственно. Смоделируем ситуацию в будущем, при которой этот показатель улучшился до 50 единиц в обоих регионах.

Республика Адыгея:

»Alcohol(. . . , 140.2, . . .)	Rating = 0.3490
»Alcohol(. . . , 50.0, . . .)	Rating = 0.6500

Алтайский край:

»Alcohol(. . . , 196.1, . . .)	Rating = 0.6540
»Alcohol(. . . , 50.0, . . .)	Rating = 0.6546

Как видно из приведенных расчётов, в Адыгее индекс алкогольного благополучия существенно улучшился, а в Алтайском крае наблюдается лишь незначительное его изменение.

Такое моделирование является решением прямой задачи (мы изменили один или несколько входных параметров и посмотрели отклик). Такое моделирование может помочь в оценке эффективности конкретных мероприятий с точки зрения улучшения алкогольного благополучия в регионе или городе. В такой ситуации, возможно также решение более сложных и ценных с практической точки зрения обратных задач.

Пусть задан бюджет на улучшение ситуации в регионе X. Пусть, далее, известны оценки стоимости изменения исходных данных (например, расходы на рекламу, «устрашающие наклейки», ограничения на продажу и пр.) c_j $(j = 1, ..., N)$, где N – количество косвенных (наблюдаемых) параметров. Обозначим изменение целевой вершины структуры индекса через $\triangle a_0$.

Задача 1: Найти множество параметров $i_1, ..., i_n$ $(n \leq N) : \triangle a_0 \to max$, $\sum\limits_{j=1}^{n} c_{i_j} \leq X$.

Это задача нахождения максимального эффекта в рамках заданного бюджета.

Пусть задано целевое изменение алкогольного благополучия q (например, q=10%). При заданных условиях возможно и решение сопряжённой задачи:

Задача 2: Найти множество параметров $i_1, ..., i_n$ $(n \leq N) : \sum\limits_{j=1}^{n} c_{i_j} \to min$, $\triangle a_0 \geq q$.

Это задача нахождения минимального бюджета, позволяющего достичь необходимого эффекта.

Решение таких задач позволит оптимизировать финансовые расходы при реализации целевых программ, направленных на снижение пагубного потребления алкоголя в стране, а также на региональном и муниципальном уровне.

6. Заключение

В работе была рассмотрена задача построения индекса алкогольного благополучия в субъектах Российской Федерации. Было представлено описание его построения в среде Matlab и приведены результаты расчета индекса для всех регионов страны. Результаты согласуются с представлением экспертов о расположении регионов, упорядоченных по значению

индекса, относительно друг друга. Также были предложены две задачи, решение которых определяет направление дальнейших исследований.

Список литературы

[1] Горный Б. Э., Калинина А. М., Бойцов С. А. Выявление лиц с риском пагубного употребления алкоголя в ходе диспансеризации: методологические аспекты. Профилактическая медицина. 2015;18(4):12-16. https://doi.org/10.17116/profmed201518412-16

[2] Горный Б. Э., Калинина А. М. Интегральная оценка алкогольной ситуации на региональном уровне. Профилактическая медицина. 2016;19(3):34-40. https://doi.org/10.17116/profmed201619334-40

[3] Горный Б. Э., Калинина А. М. Связь интегральной оценки региональной алкогольной ситуации и смертности населения от некоторых хронических неинфекционных заболеваний. Профилактическая медицина. 2019;22(4):65-68. https://doi.org/10.17116/profmed20192204165

[4] Рыжов А.П. Оценка и мониторинг процессов в социотехнических системах и связанные с ними задачи. Интеллектуальные Системы Теория и приложения. 2018;22(2):129-139. http://intsysjournal.org/pdfs/22-2/Rizov.pdf

[5] Global strategy to reduce the harmful use of alcohol, WHO

[6] Масленникова Г. Я., Лепехин В. А. Алкоголизм в Российской Федерации: время принятия решений. Профилактическая Медицина. 2012;15(2):46–49

[7] European action plan to reduce the harmful use of alcohol 2012-2020, WHO, 2012

[8] Краснова П. С. Потребление алкоголя: тенденции и социальные последствия. Проблемы Развития Территории. 2011;55(3)

[9] Разводовский Ю. Е. Алкогольная ситуация в Беларуси. Вопросы Наркологии. 2013;(3):5–13

[10] Сахаров А. В. Организационная модель мониторинга алкогольной ситуации в отдельном субъекте Российской Федерации. Сибирский Вестник Психиатрии и Наркологии. 2012;(1):80–82

[11] Аксютина С. В., Овсянкина Н. М. Актуальные вопросы алкогольной безопасности региона. Экономические и Социальные Перемены: Факты, Тенденции, Прогноз. 2015;1(37)

[12] Немцов А. В., Разводовский Ю. Е. Алкогольная ситуация в России, 1980-2005 гг. Социальная и Клиническая Психиатрия. 2008;18(2)

[13] ЕМИСС Государственная статистика. https://www.fedstat.ru/

[14] Matlab. https://www.mathworks.com/products/matlab.html

[15] Bezdek, James C. Pattern Recognition with Fuzzy Objective Function Algorithms. 1981

Measurement of alcohol well-being in regions based on statistical information

Ryjov A.P., Gorny B.E., Zudin A.V.

To reduce the risks associated with alcohol abuse, it is extremely important to assess alcohol consumption levels. At the same time, it is not enough to know indicators that assess only the volume of retail sales, which does not take into account unrecorded alcohol consumption. The development of an integral indicator of the index, based on available statistical data, will make it possible to level the risks of expert assessment and increase the efficiency of spending budget funds in the field of healthcare.

The report provides a description of the available expert and factual information, the structure of the index, an approach to its assessment by means of the theory of fuzzy sets is proposed. The features of the implementation of the index calculation in the Matlab environment are considered, examples are given. The related direct and inverse problems are formulated and discussed, the solution of which will optimize the efficiency of the healthcare system in this parameter at the regional and federal levels. The alcohol well-being index is one of the important parameters of the health care system, on the one hand, and a typical index of socio-economic processes, on the other, therefore, the approaches described in the report can be applied to construct a wide set of such indices.

Keywords: preventive medicine, alcohol well-being, process evaluation and monitoring.

References

[1] Gorny B.E., Kalinina A. M., Boytsov S. A. Identification of persons at risk of harmful use of alcohol during clinical examination:

methodological aspects. Preventive medicine. (In Russian) 2015;18(4):12-16. https://doi.org/10.17116/profmed201518412-16

[2] Gorny B.E., Kalinina A. M. Integral assessment of the alcoholic situation at the regional level. Preventive medicine. (In Russian) 2016;19(3):34-40. https://doi.org/10.17116/profmed201619334-40

[3] Gorny B.E., Kalinina A. M. The relationship between the integral assessment of the regional alcoholic situation and the mortality rate of the population from certain chronic non-infectious diseases. Preventive medicine. (In Russian) 2019;22(4):65-68. https://doi.org/10.17116/profmed20192204165

[4] Ryjov A.P. Assessment and monitoring of processes in socio-technical systems and related tasks. Intelligent Systems Theory and Applications. (In Russian) 2018;22(2):129-139. http://intsysjournal.org/pdfs/22-2/Rizov.pdf

[5] Global strategy to reduce the harmful use of alcohol, WHO

[6] Maslennikova G. Y., Lepekhin V. A. Alcoholism in the Russian Federation: Time for Decision Making. Preventive Medicine. (In Russian) 2012;15(2):46–49

[7] European action plan to reduce the harmful use of alcohol 2012-2020, WHO, 2012

[8] Krasnova P. S. Alcohol consumption: trends and social consequences. Problems of Development of the Territory. (In Russian) 2011;55(3)

[9] Razvodovsky Y. E. Alcohol situation in Belarus. Questions of Narcology. (In Russian) 2013;(3):5–13

[10] Sakharov A. V. Organizational model for monitoring the alcohol situation in a separate subject of the Russian Federation. Siberian Bulletin of Psychiatry and Narcology. (In Russian) 2012;(1):80–82

[11] Aksyutina S. V., Ovsyankina N. M. Topical issues of alcohol safety in the region. Economic and Social Change: Facts, Trends, Forecast. (In Russian) 2015;1(37)

[12] Nemtsov A. V., Razvodovsky Y. E. The alcohol situation in Russia, 1980-2005. Social and Clinical Psychiatry. (In Russian) 2008;18(2)

[13] Unified interdepartmental information and statistical system. State statistics. (In Russian) https://www.fedstat.ru/

[14] Matlab. https://www.mathworks.com/products/matlab.html

[15] Bezdek, James C. Pattern Recognition with Fuzzy Objective Function Algorithms. 1981

Метод чередования обучаемых параметров

А. А. Хусаенов[1]

В работе предлагается метод повышения качества обучения сверточных искусственных нейронных сетей (ИНС) за счет разделения параметров по их возможности расширения рецептивного поля. При обучении ResNet50 достигается увеличение точности за счет чередуемой остановки обучения в 4-х слоях, расширяющих рецептивное поле.

Показано, что повышение обобщающей способности модели при использовании предложенного метода достигается за счет устранения избыточного вклада отдельных существенных (окклюзивных) элементов изображения при формировании карт признаков. В пользу указанных предположений приводятся результаты экспериментов в задаче transfer learning и рассуждения относительно существования указанной проблемы.

Предлагаемые подходы могут оказаться полезными, в частности, при обучении ИНС на малых данных или дистилляции обучающего множества, где проблемы переобучения на отдельных окклюзивных признаках имеют высокую значимость.

Ключевые слова: сверточная искусственная нейронная сеть, рецептивное поле нейрона, проблемы переобучения моделей, окклюзия признаков в сверточных искусственных нейронных сетях

1. Введение

В задачах распознавания образов в сверточных искусственных нейронных сетях (ИНС) существует проблема переобучения на отдельных окклюзивных (существенных) признаках [1, 2, 3, 4] (п.3.3) исходного изображения.

Например, одна из популярных проблем распознавания «леопардовый диван»: за счет явно-выраженного образа текстуры игнорируются общие признаки объекта (то есть объект «диван» распознается как объект «леопард»)

Подобное переобучение приводит к понижению обобщающей способности модели при отсутствии или недостаточно полной аугментации данных. Указанная проблема переобучения связана с существенными поте-

[1] *Хусаенов Артем Азатович* — аспирант, м.н.с. кафедры математической теории интеллектуальных систем мех.-мат. ф-та МГУ; e-mail: a.khusaenov@mail.ru

Khusaenov Artem Azatovich — postgraduate student, junior research fellow, Moscow State University, faculty of Mechanics and Mathematics, Mathematical Theory of Intelligent Systems department

рями в точности распознавания, если при аугментации существенных признаков обучающего множества будут отсутствовать случаи, которые могут встречаться во время эксплуатации моделей. Причем указанные ошибки не будут выявлены во время проверки модели в случае, если в проверочном множестве соответствующая аугментация существенных признаков также отсутствует.

В задачах понижения вычислительной сложности ИНС и существенного сокращения объема обучающего множества (малые данные) [5, 6, 7, 8] рассматриваемая проблема переобучения усугубляется, а ее выявление во время валидации по-прежнему остается невозможным при недостаточной аугментации данных.

Существующие методы устранения указанной проблемы в большинстве своем сводятся к регуляризации параметров ИНС и более полной аугментации данных, что во многом позволяет сократить переобучение при достаточном числе эпох. Однако, методов решения проблем окклюзии карт признаков, возникающей из-за отдельных существенных элементов изображения, на сегодняшний день существует не так много (например, attention-механизмы [9]).

В данной работе предлагается метод оптимизации указанного переобучения за счет сокращения вклада отдельных окклюзивных элементов изображения при формировании карт признаков сверточной ИНС. Предполагается, что вклад указанных признаков может быть сокращен в слоях, производящих значительное увеличение рецептивного поля исходного изображения. Демонстрируется увеличение обобщающей способности карт признаков при использовании предлагаемого подхода в задачах transfer learning с несколькими типами архитектур ИНС.

2. Рецептивное поле

2.1. Естественно-биологические инварианты зрительного восприятия

С точки зрения естественно-биологического описания восприимчивых областей естественных нейронов, рецептивное поле зрительных рецепторов определяется как область в поле зрения, где зрительные нейроны реагируют на визуальные стимулы [10].

Базовое свойство зрительного восприятия [10] заключается в следующей особенности передачи сигнала рецептору: когда свет достигает визуального датчика, такого как сетчатка, информация, необходимая для определения свойств окружающего мира, содержится не в значениях интенсивности изображения в одной точке, а в соотношениях между значениями интенсивности в разных точках [10]. Вводя, теперь, более фор-

мальное описание, рецептивное поле зрительного нейрона может быть определено как область поля зрения (область визуальных датчиков) на визуальные стимулы которого он реагирует [11].

Основная мотивация вычислительной теории рецептивных полей [10, 12] заключается в учете свойств проекции 3-мерных объектов на 2-мерный датчик освещенности (сетчатку), где данные изображения могут подвергаться базовым преобразованиям следующего вида [10]:

локальные масштабные преобразования, вызванные объектами разных размеров и на разных расстояниях для наблюдателя	(1)
локальные аффинные преобразования, вызванные изменениями направления обзора относительно объекта	(2)
локальные преобразования Галилея, вызванные относительными движениями между объектом и наблюдателем	(3)
локальные мультипликативные преобразования интенсивности, вызванные изменениями освещенности	(4)

Тогда, поскольку зрительная система способна поддерживать стабильное восприятие окружающей среды в условиях указанной выше аугментации сигнала, одним из ключевых требований к математической формализации является устойчивость модели к таким преобразованиям.

2.2. Инварианты обработки зрительных образов в искусственных нейронных сетях

С точки зрения моделирования процессов зрительного восприятия в сверточных ИНС [13] от модели требуется устойчивость к указанным преобразованиям и достижение обобщающей способности при достаточно полном и аугментированном обучающем множестве. Однако, исходя из предпосылок к усилению отдельных сигналов в пользу минимизации функционала потерь, очевидно предполагать, что отдельные свойства, сохраняющиеся во всех объектах одного класса, будут вносить существенный вклад в карты признаков сверточной ИНС. Особенно, если эти свойства изображения являются устойчивыми (инвариантными) к преобразованиям (1)–(4) – то есть остаются неизменными во всем обучающем множестве. Для упрощения повествования подобные элементы

изображения будем называть **существенными элементами** или **существенными сигналами**.

Подобные усиления сигналов можно наблюдать [14] в автоассоциативных ИНС (автоэнкодерах), где для осуществления обратного отображения (декодирования) с наименьшими потерями ИНС избыточно усиливает сигналы отдельных признаков, имеющих сильную корреляцию с целевым признаком.

При формировании карт признаков во всех слоях сверточной ИНС может сохраняться избыточный вклад подобных существенных элементов. Об этом свидетельствуют исследования, посвященные окклюзии изображений в сверточных ИНС [1, 2, 3, 4].

Предполагается, что вклад подобных существенных элементов усиливается при формировании карт признаков в слоях ИНС, производящих расширение рецептивного поля исходного изображения (п.2.4).

В данной работе предлагается метод чередования обучаемых параметров ИНС, который позволяет увеличить обобщающую способность модели за счет ослабления вклада отдельных подобных сигналов путем заморозки отдельных слоев, увеличивающих рецептивное поле. Предполагается, что рассматриваемый эффект повышения обобщающей способности может быть достигнут при заморозке параметров указанных слоев ИНС на последних этапах обучения.

2.3. Вычисление рецептивного поля в искусственной сверточной нейронной сети

Рецептивное поле нейрона [6-8] – это область входного изображения в сверточной ИНС, от которой зависит реакция этого нейрона. На рис. 1 представлен пример [6] рецептивного поля для 2-х слоев свертки с ядром 3x3.

Пусть сверточная ИНС имеет L слоев. Выходные сигналы l-го слоя ($l = \overline{1,n}$) будем обозначать как f_l (то есть f_0 – это входное изображение, а f_n – сигналы последнего слоя). Каждый сверточный слой l имеет 3 параметра свертки:

- k_l - размер ядра свертки
- s_l – шаг ядра свертки
- p_l – размер паддинга

Рассмотрим случай одномерного входного сигнала. Размер рецептивного поля нейронов из слоя l будем обозначать как r_l. В работе [16] предлагаются подходы к вычислению исходного рецептивного поля для рассматриваемого случая.

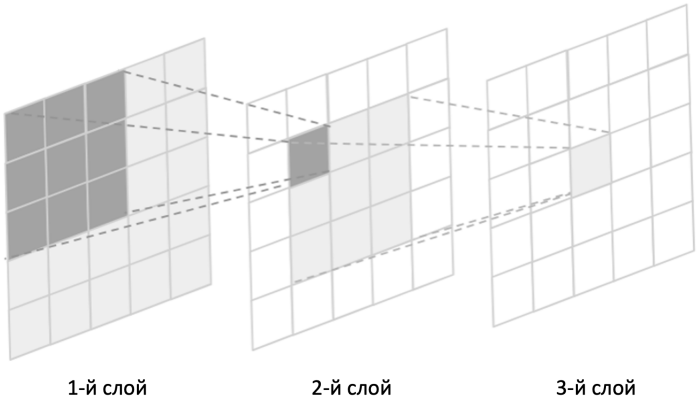


Рис 1. Пример рецептивного поля в сверточных слоях ИНС [16]
(1-й слой - входное изображение, 2-й слой - свертка с ядром 3x3, 3-й слой - свертка с ядром 3x3. Темно-серый - рецептивное поле одного нейрона из второго слоя, светло-серый - рецептивное поле для одного нейрона третьего слоя)

Рецептивное поле l-го сверточного слоя для 1-мерного входного вектора может быть вычислено [16] следующим образом

$$r_{l-1} = s_l \cdot r_l + (k_l - s_l) \tag{5}$$

Представленный подход предполагает рекурсивное вычисление [16] размера исходного рецептивного поля r_0 на основе параметров (s_i, k_i), где $i = \overline{0, L}$ (учет паддинга p_i будет представлен далее).

$$r_0 = \sum_{l=1}^{L} \left((k_l - 1) \prod_{i=1}^{l-1} s_i \right) + 1 \tag{6}$$

Причем, как будет рассмотрено позднее, размер рецептивного поля для слоев подвыборки в данном случае [16] аналогично представляется с помощью параметров s_i и k_i.

Рассмотрим, теперь, вывод [16] значений координат рецептивного поля. Пусть u_l и v_l – это крайний левый и крайний правый координаты рецептивного поля в слое l соответственно. Координаты рецептивного поля для $l - 1$ слоя могут быть выражены [16] следующим образом

$$u_{l-1} = -p_l + u_l \cdot s_l \tag{7}$$

$$v_{l-1} = -p_l + v_l \cdot s_l + k_l - 1 \tag{8}$$

Координаты исходного рецептивного поля могут быть выражены рекурсивно [16].

$$u_0 = u_L \prod_{i=1}^{L} s_i - \sum_{l=1}^{L} p_l \prod_{i=1}^{l-1} s_i \tag{9}$$

$$v_0 = v_L \prod_{i=1}^{L} s_i - \sum_{l=1}^{L} (1 + p_l - k_l) \prod_{i=1}^{l-1} s_i \tag{10}$$

Для удобства вычисления исходного рецептивного поля введем совокпный сдвиг S_l (страйд) и совокупный паддинг P_l [16]

$$S_l = \prod_{i=l+1}^{L} s_i \tag{11}$$

$$P_l = \sum_{m=l+1}^{L} p_m \prod_{i=l+1}^{m-1} s_i \tag{12}$$

Тогда, теперь, координаты исходного рецептивного поля могут быть выражены следующим образом

$$u_0 = -P_0 + u_L \cdot S_0 \tag{13}$$

$$v_0 = u_0 + r_0 - 1 \tag{14}$$

Размер рецептивного поля после применения подвыборки (пулинга) с ядром размера k_{L+1} в последнем слое может быть выражен следующим образом

$$r_0 = 1 + \sum_{l=1}^{L} \left((k_l - 1) \prod_{i=1}^{l-1} s_i \right) + (k_{L+1} - 1) \prod_{i=1}^{L} s_i \tag{15}$$

Нетрудно заметить, что **слой подвыборки значительно расширяет рецептивное поле**. Этим замечанием мы воспользуемся далее в п.2.3.

В таблице 1 представлен пример вычисления размера рецептивного поля для ИНС AlexNet [15]

Таблица 1. Пример вычисления рецептивного поля для ИНС AlexNet [15]

l	**Тип слоя**	r_l	k_l	s_l
8	max pooling	1	3	2
7	convolution	3	3	1
6	convolution	5	3	1
5	convolution	7	3	1
4	max pooling	9	3	2
3	convolution	19	5	1
2	max pooling	23	3	2
1	convolution	47	11	4
0	**input**	**195**	-	-

В таблице 2 [16] представлены размеры рецептивного поля для различных архитектур сверточных ИНС.

Таблица 2. Размер рецептивного поля для некоторых ИНС [16] (где $|l_0|$ - длина входного вектора)

Модель	r_0	$\|l_0\|$	S_0	P_0
alexnet v2	195	224	32	64
vgg 16	212	224	32	90
mobilenet v1	315	224	32	126
mobilenet v1 075	315	224	32	126
resnet v1 50	483	224	32	239
resnet v1 101	1027	224	32	511
resnet v1 152	1507	224	32	751
resnet v1 200	1763	224	32	879
inception v2	699	224	32	318
inception v3	1311	224	32	618
inception v4	2071	224	32	998
inception resnet v2	3039	224	32	1482

Необходимо заметить [16], что по мере развития моделей рецептивное поле исходного изображения увеличивается [16] (рис.2).

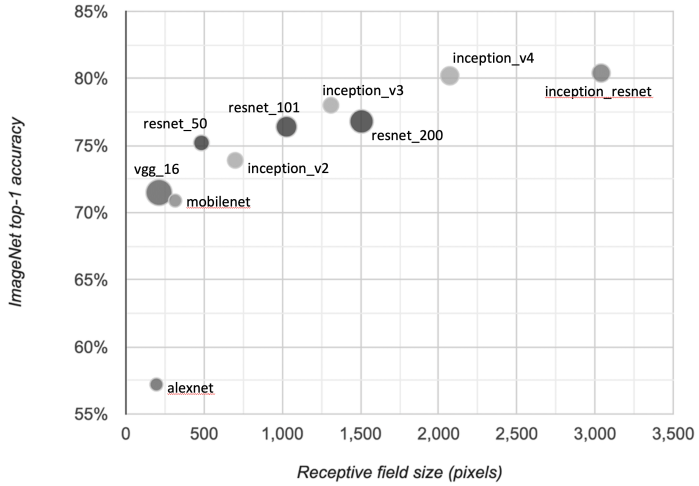


Рис 2. график качества классификации в зависимости от размера рецептивного поля [16]

2.4. Слои увеличения рецептивных полей

В работе [16] рецептивное поле определяется как область исходного изображения, сигналы которой участвуют в формировании карт признаков. Как было замечено ранее, рецептивное поле значительно увеличивается за счет слоев подвыборки.

При этом существуют слои, производящие значительное увеличение рецептивного поля не в смысле размера области исходного изображения, а относительно вклада его элементов в качество классификации.

Часть рецептивного поля, вносящая существенный вклад в качество классификации, будем называть *действительным (effective receptive filed)* [17]. Рецептивное поле, оцениваемое в смысле координат исходной области изображения, в дальнейшем будем называть *вычисляемым*.

В работе [17] демонстрируется увеличение действительного рецептивного поля как за счет отдельных слоев некоторой сверточной ИНС (рис.3), так и в процессе обучения указанной ИНС (рис.4). В качестве меры влияния отедельных входных сигналов $x^n_{(i,j)}$ некоторого слоя n на выходные сигналы $y^n_{(i,j)}$, оценивается частная производная $\partial y^n_{(i,j)}/\partial x^n_{(i,j)}$ [17].

Предполагается, что вклад существенных элементов изображения усиливается в слоях ИНС, производящих значительное расширение вычисляемого рецептивного поля (п.2.3). При формировании карт признаков отдельные сигналы, полученные из вычисляемого рецептивного поля, могут ослабляться в пользу сигналов, полученных на основе существенных элементов изображения, сохраняющихся во всех объектах обу-

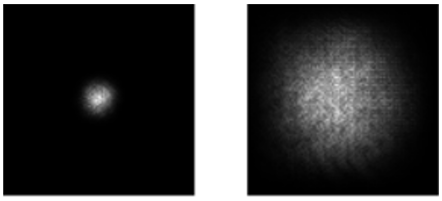


Рис 3. Рецептивное поле в зависимости от слоя ИНС [17]
(а) - рецептивное поле сверточного слоя; (б) - рецептивное поле слоя подвыборки (пуллинга)

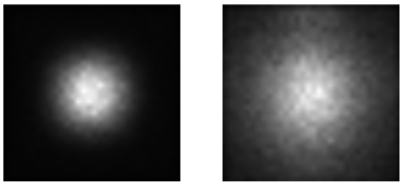


Рис 4. Рецептивное поле до и после обучения ИНС [17]
(а) - рецептивное поле до обучения; (б) - рецептивное поле после обучения (пуллинга)

чающего множества (п.2.2). Особенно, если эти элементы устойчивы (инварианты) к преобразованиям (1)-(4).

Таким образом, небольшие размеры **действительного рецептивного поля** по отношению к размерам входного изображения или **вычислимого рецептивного поля** можно рассматривать признаком недостаточного обучения ИНС. Понятно, что слои увеличения рецептивных полей играют особую роль в расширении действительного рецептивного поля.

3. Преодоление переобучения

3.1. Определение переобучения

В качестве базового представления проблемы переобучения модели рассмотрим *закон смещения-дисперсии* [18, 19] для задачи классификации.

Пусть $\{x_1, x_2, \ldots, x_n\}$ – некоторое обучающее множество, где каждому объекту x_i соответствует некоторое вещественное число y_i (принадлежность к классу, учитель). При этом существует целевая зависимость y, которая определена как на указанном множестве $\{x_1, x_2, \ldots, x_n\}$, так и за его пределами. Причем целевая зависимость может быть представ-

лена следующим образом.

$$y(x_i) = f(x_i) + \epsilon \tag{16}$$

где $f(x) : R^p \to R^k$ есть некоторая функция, а ϵ - есть случайная величина (шум). Будем считать, что ϵ имеет нулевое среднее $M\epsilon = 0$ и дисперсию σ.

Данную формализацию можно рассматривать следующим образом: для некоторых объектов обучающего множества $\{x_1, x_2, ..., x_n\}$ доступны ответы $\hat{f}(x)$, на основе которых необходимо аппроксимировать целевую зависимость y за пределами указанного множества $\{x_1, x_2, ..., x_n\}$.

В таком случае задача машинного обучения заключается в нахождении приближающей функции (модели) $a(x)$, которая с допустимой точностью аппроксимирует целевую зависимость y как на всем обучающем множестве $\{x_1, x_2, \ldots, x_n\}$, так и за его пределами. В качестве ошибки аппроксимации будем рассматривать среднеквдатратическое отклонение $(M(y) - a(x))^2$.

$$\begin{aligned}
M(y(x) - a(x))^2 &= M\big(y^2(x) - 2y(x)a(x) + a^2(x)\big) = \\
&= M\big(a^2(x)\big) - 2M\big(y(x)a(x)\big) + M\big(y^2(x)\big) = \\
&= M\big(a^2(x)\big) + M\big(y^2(x)\big) - 2M\big((f+\epsilon)a(x)\big) = \\
&= M\big(a^2(x)\big) + M\big(y^2(x)\big) - 2M\big(fa(x)\big) - 2M\big(\epsilon a(x)\big) = \\
&= M\big(a^2(x)\big) - \big(M(a(x))\big)^2 + \big(M(a(x))\big)^2 + \\
&\quad + M\big(y^2(x)\big) - \big(M(y(x))\big)^2 \\
&\quad + \big(M(y(x))\big)^2 - 2M\big(fa(x)\big) = \\
&= D\big(y(x)\big) + D\big(a(x)\big) + \big(M(y(x))\big)^2 + \\
&\quad + \big(M(a(x))\big)^2 - 2M\big(fa(x)\big) = \\
&= D\big(y(x)\big) + D\big(a(x)\big) + \big(M(f)\big)^2 - \\
&\quad - 2M\big(fa(x)\big) + \big(M(a(x))\big)^2 = \\
&= D\big(a(x)\big) + \big(M(f - a(x))\big)^2 + D\big(y(x)\big) = \\
&= variance\big(a(x)\big) + bias\big(f, a(x)\big) + \sigma^2
\end{aligned}$$

где

- $bias\big(\hat{f}, a(x)\big) = \big(M(\hat{f} - a(x))\big)^2$ - *смещение* модели, то есть ошибка относительно заданного множества точек $\{x_1, x_2, ..., x_n\}$
- $variance\big(a(x)\big) = D\big(a(x)\big) = M\big(a^2(x)\big) - \big(Ma(x)\big)^2$ - *дисперсия* модели, то есть разброс значений относительно среднего на заданном множестве

- $\sigma^2 = D(y)$ - дисперсия целевой зависимости, рассматриваемая как *неустранимая ошибка*

Закон смещения - дисперсии предполагает [19]: чем больше точек $(x_1, x_2, ..., x_n)$ захватывает модель $a(x)$, тем ниже смещение (bias), однако выше ее дисперсия (variance). Если предполагать, что при росте сложности модели увеличивается число точек, которые она способна захватить с допустимой точностью, то данный закон может быть проиллюстрирован следующим образом (рис.5) [19].

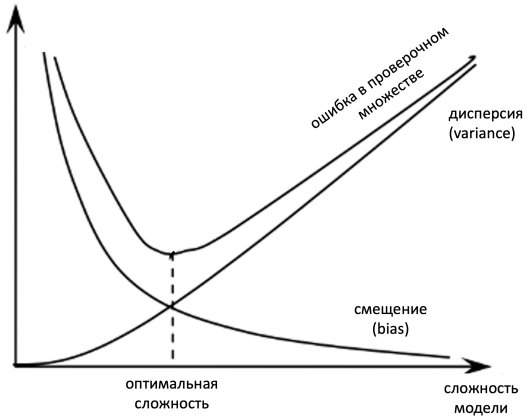


Рис 5. Иллюстрация закона смещения-дисперсии [19]

Тогда проблема переобучения может быть выражена следующим образом:

Определение: *переобучением ИНС будем называть рост ошибки на проверочном множестве при увеличении сложности модели.*

Для моделей ИНС данный закон может быть выражен в терминах сложности ИНС, где сложность модели определяется числом разделяющих гиперповерхностей (нейронов).

3.2. Избыточность модели или недостаточность данных

В данном параграфе приводятся рассуждения, подкрепляющие введенное ранее определение переобучения (п.3.1.).

Избыточная сложность модели в случае ИНС может описываться как избыточное число разделяющих гиперповерхностей при высокой гранулированности групп объектов в пространстве признаков (рис.6).

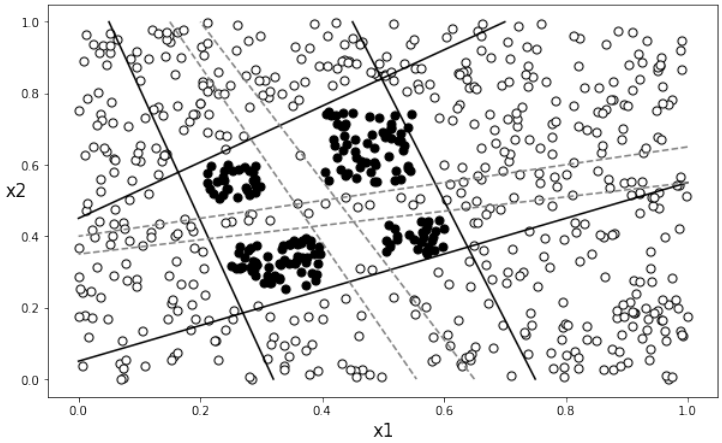


Рис 6. Пример избыточной сложности модели при гранулированных классах

При высокой гранулированности обучающего множества и при существовании отдельных признаков, по которым объекты разных классов становятся легко отделимыми, появляется набор существенных переменных для функции классификации (рис.7).

В таком случае переобучение ИНС возникает не только при длительном обучении, но и при излишнем дроблении признакового пространства из-за избыточной сложности модели.

3.3. Переобучение на окклюзивных признаках

В задаче распознавания образов отдельные элементы изображения, сохраняющиеся во всем множестве объектов некоторого класса, являются существенными признаками. В особенности, если эти признаки являются устойчивыми (инвариантными) к преобразованиям (1)-(4), так как их образ сохраняется неизменным во всех объектах обучающего множества.

О существовании проблем переобучения на существенных признаках (п.3.2) в задачах распознавания образов свидетельствуют эксперименты по оценке окклюзии изображений [1, 2, 3, 4] в сверточных ИНС. О подобных проблемах так же свидетельствует множество экспериментов с

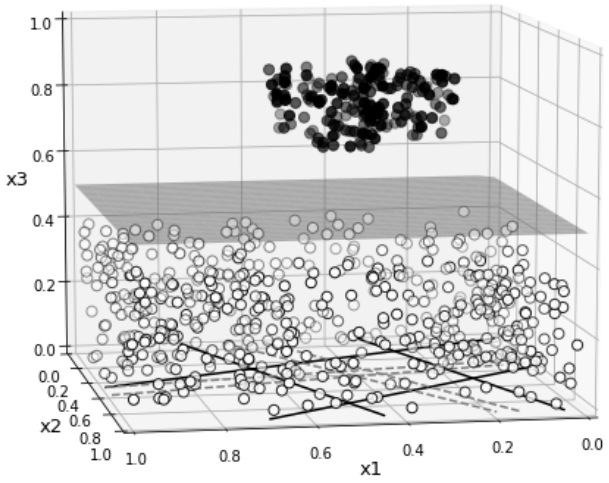


Рис 7. Пример существенной переменной (x_3) в пространстве признаков

зашумлением изображений (атаки на ИНС), приводящим к значительному понижению точности [20].

Базовые подходы по оценке окклюзии [1] позволяют определить влияние отдельных существенных признаков на точность классификации. При зашумлении отдельных элементов изображения оценивается изменение точности классификации в обученной ИНС. Производя указанную операцию итерационно по всем областям изображения, возможно оценить степень влияния отдельных элементов изображения на финальное предсказание [4].

Эксперименты по окклюзии изображений [1, 2, 3, 4] демонстрируют проблему ухудшения обобщающей способности модели за счет избыточного внимания на отдельных признаках изображения. То есть, проблема переобучения в данном случае рассматривается как проблема избыточного вклада отдельных окклюзивных элементов изображения в карты признаков сверточной ИНС.

4. Метод чередования обучаемых параметров

4.1. Описание метода

В данной работе предлагается метод чередования обучаемых параметров ИНС, позволяющий увеличить обобщающую способность модели за счет ослабления вклада отдельных существенных сигналов (п.2.2).

Предполагается, что рассматриваемое переобучение можно регуляризировать за счет попеременной остановки обучения слоев, увеличивающих вычисляемое рецептивное поле. В таком случае возможно усилить участие других менее существенных элементов изображения при формировании карт признаков, то есть увеличить действительное рецептивное поле, что приведет к повышению обобщающей способности модели. Данную технику чередования обучаемых параметров будем далее называть **обучением с фиксированным рецептивным полем (RFF – receptive field freeze)**.

4.2. Эксперименты

Для демонстрации метода рассматривается классическая задача transfer-learning. Данный подход выбран, поскольку позволяет оценить повышение обобщающей способности модели на уровне формирования карт признаков, а не классификатора в последнем слое.

ИНС обучается на некотором наборе изображений (указано далее), принадлежащих множеству базовых классов. Далее производится обучение последнего (выходного) классификатор-слоя ИНС на группе классов, не входящих в базовое обучающее множество. Обучение ИН на исходном наборе классов (до transfer learning) для упрощения изложения будем называть базовым обучением.

Эксперимент имеет следующий вид (рис.8):

Шаг 1. Базовое обучение модели

Базовое обучение модели в течении n эпох

Шаг 2. Receptive Field Freeze - обучение (RFF)

Дообучение модели в двух вариациях:

- **модель 1:** базовое обучение ИНС в эпохе $(n+1)$
- **модель 2:** RFF обучение ИНС в эпохе $(n+1)$

Шаг 3. Transfer learning

На основе каждой из 2-х моделей производится transfer-learning (с одинаковыми гиперпараметрами)

Для чистоты эксперимента на шаге 1 на протяжении первых n эпох обучается 1 общая модель (шаг 1). Затем на эпохе $(n+1)$ производится разветвление на 2 модели (шаг 2). Далее для каждой модели производится transfer learning с одинаковым гиперпараметрами, одинаковой

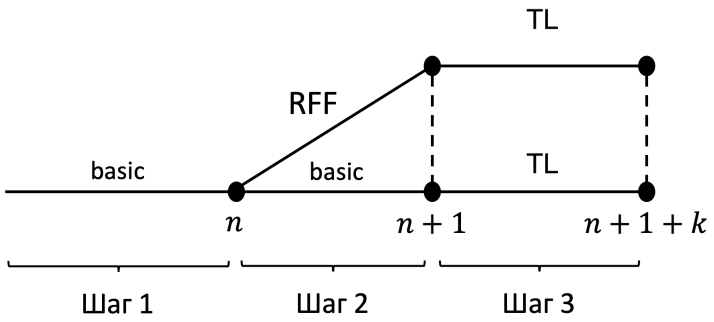


Рис 8. Схема эксперимента

стартовой инициализацией весов в последнем классификатор-слое и общим генератором случайных чисел.

На основе указанной схемы далее производятся 3 эксперимента: базовый эксперимент, эксперимент с преодолением переобучения ИНС, эксперимент с глубокой ИНС ResNet50.

4.2.1. Базовый эксперимент

В базовом эксперименте используется датасет CIFAR-100. Рассматривается архитектура ИНС, состоящая из 2-х типов блоков: блоки сверток без подвыборки и блоки с подвыборкой. Архитектура ИНС представлена в таблице 3.

В рамках обучения с фиксированными рецептивными полями (receptive field freeze, RFF) в указанной ИНС фиксируются параметры слоев, производящих свертки сразу после операции подвыборки (max pooling), то етсь слои 7 и 13. Шаги эксперимента указаны в таблице 4.

Для оценки устойчивости метода, датасет CIFAR-100 был разбит на 10 подмножеств классов. Каждое из подмножеств было разбито на 2 группы классов: классы для базового обучения и классы для transfer-learning. Итоговые группы подмножеств представлены в таблице 5.

Таблица 3. Архитектура базовой ИНС

N	**Слой**	**Размер карты**	**Кол-во ядер**	**Размер ядра**
1	Convolutional	(32,32)	32	(3,3)
2	BatchNorm	(32,32)	-	-
3	ReLU	(32,32)	-	-
4	Convolutional	(32,32)	32	(3,3)
5	ReLU	(32,32)	-	-
6	MaxPooling	(16,16)	32	(2,2)
7	Convolutional	(16,16)	64	(3,3)
8	BatchNorm	(16,16)	-	-
9	ReLU	(16,16)	-	-
10	Convolutional	(16,16)	64	(3,3)
11	ReLU	(16,16)	-	-
12	MaxPooling	(8,8)	64	(2,2)
13	Convolutional	(8,8)	64	(3,3)
14	BatchNorm	(8,8)	-	-
15	ReLU	(8,8)	-	-
16	Convolutional	(8,8)	64	(3,3)
17	ReLU	(8,8)	-	-
18	Flatten	4096	-	-
19	Danse	64	-	-
20	ReLU	64	-	-
21	Danse	8	-	-

Таблица 4. Схема экспериментов 1.1-1.10

Номер эпохи	**Модель 1**	**Модель 2**
0	инициализация параметров	
[1,4]	базовое обучение	
5	базовое обучение	RFF-обучение
6	transfer-learning	transfer-learning

Таблица 5. Разбиение классов в экспериментах 1.1-1.10

N	**Классы для базового обучения**	**Классы для transfer-learning**
1.1	яблоко, хомяк, ребенок, медведь, часы, пчела, велосипед, бутылка	жук, бобр
1.2	мост, автобус, верблюд, девушка, банка, замок, гусеница, крупный рогатый скот	чаша, мальчик
1.3	шимпанзе, облако, кровать, таракан, диван, гора, чашка, динозавр	стул, крокодил
1.4	дельфин, камбала, лес, лиса, бабочка, дом, кенгуру, клавиатура	слон, аквариумная рыбка
1.5	лампа, газонокосилка, леопард, лев, ящерица, омар, черепаха, кленовое дерево	краб, мотоцикл
1.6	мышь, гриб, дубовое дерево, апельсин, орхидея, выдра, пикап, сосновое дерево	пальмовое дерево, груша
1.7	равнина, тарелка, мак, опоссум, дикобраз, енот, луч, дорога	кролик, ракета
1.8	роза, море, тюлень, землеройка, небоскреб, улитка, змея, паук	скунс, акула
1.9	белка, поезд, сладкий перец, стол, танк, телевизор, тигр, трактор	телефон, трамвай
1.10	подсолнух, форель, шкаф, кит, ивовое дерево, волк, женщина, червь	мужчина, тюльпан

Таким образом, для оценки устойчивости метода, эксперимент повторялся 10 раз на разных классах изображений. На каждом из подмножеств классов производился эксперимент следующего вида. Для валидации отводилось 20% объектов из каждой группы классов для transfer learning. В качестве метрики оценивается accuracy валидации (val_accuracy).

Таблица 6. Результаты экспериментов 1.1-1.10

Номер эксп-та	**Модель 1**	**Модель 2**	**delta**
1.1	0,640	0,645	**+0,005**
1.2	0,590	0,595	**+0,005**
1.3	0,675	0,680	**+0,005**
1.4	0,765	0,760	$-0,005$
1.5	0,700	0,745	**+0,045**
1.6	0,890	0,905	**+0,015**
1.7	0,795	0,800	**+0,005**
1.8	0,880	0,875	$-0,005$
1.9	0,640	0,655	**+0,015**
1.10	0,790	0,825	**+0,035**
avarage delta			**+0,012**

На графиках ниже (рис.9) представлены показатели точности обучения и валиации.

Нетрудно заметить, что в экспериментах 1.4 и 1.8 (где точность валидации модели 2 оказалась ниже) модель 1 достигает наибольшей обобщающей способности относительно прочих экспериментов (так как показатели точности валидации сопоставимы с точностью обучения). В многих остальных случаях (1.1, 1.3, 1.6, 1.7, 1.9) ИНС оказалась близка к фазе переобучения на первых 4 эпохах.

Исходя из этого наблюдения, естественным образом возникает предположение, что RFF обучение позволяет избежать ранней стадии переобучения, которая выражается высокой окклюзией изображения. Эксперимент с этим предположением рассматривается в следующем разделе.

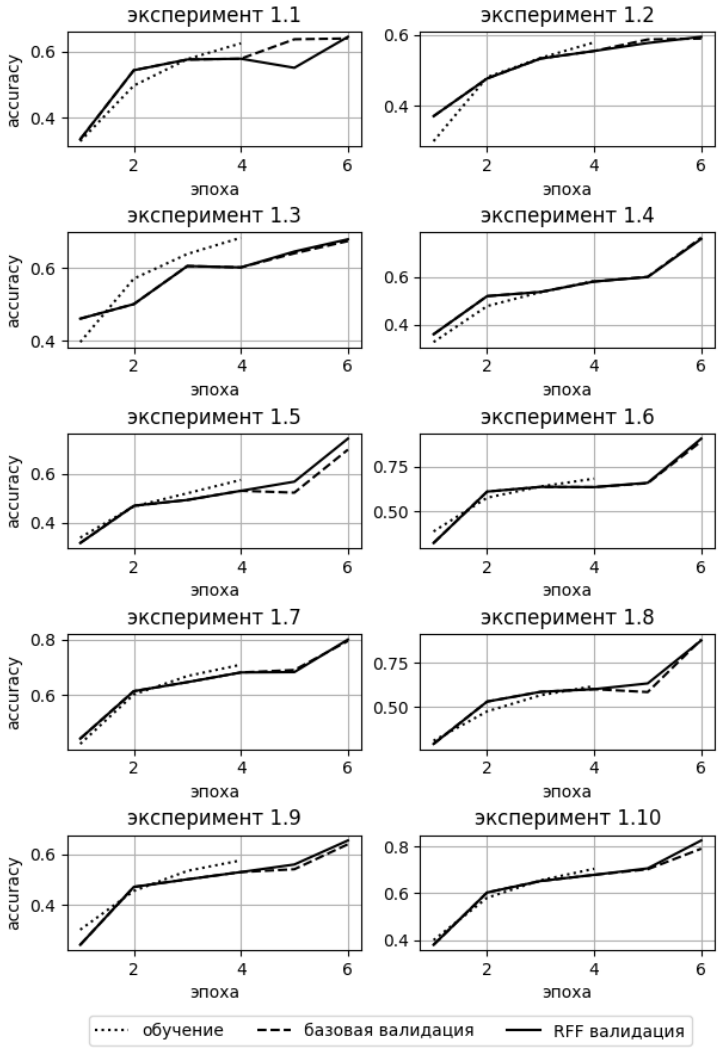


Рис 9. Результаты экспериментов 1.1-1.10

4.2.2. Эксперимент с преодолением переобучения

Для оценки предположения, рассмотренного в прошлом разделе, производится ряд экспериментов с базовым обучением в n эпох, где значение n определяется стадией переобучения ИНС. В каждом из 10 экспериментов базовое обучение производится до тех пор, пока ИНС не войдет в фазу переобучения на некоторой эпохе n. То есть, эксперимент имеет следующий вид:

Таблица 7. Схема экспериментов 2.1-2.10

Номер эпохи	**Модель 1**	**Модель 2**
0	инициализация параметров	
[1,$n-1$]	базовое обучение	
n	базовое обучение	RFF-обучение
$n+1$	transfer-learning	transfer-learning

Как и в экспериментах прошлого раздела, для валидации transfer learning отводилось 20% объектов из каждой группы классов. В качестве метрики оценивается accuracy при валидации (val_accuracy)

Таблица 8. Результаты экспериментов 2.1-2.10

Номер эксп-та	**Модель 1**	**Модель 2**	**delta**
2.1	0,640	0,635	$-0,005$
2.2	0,690	0,670	$-0,020$
2.3	0,830	0,850	+ **0,020**
2.4	0,900	0,905	+ **0,005**
2.5	0,825	0,830	+ **0,005**
2.6	0,870	0,900	+ **0,030**
2.7	0,755	0,795	+ **0,040**
2.8	0,855	0,905	+ **0,050**
2.9	0,830	0,850	+ **0,020**
2.10	0,740	0,725	$-0,015$
avarage delta			**+0,013**

На графиках ниже (рис.10) представлены показатели точности обучения и валиации. Анализ и выводы приведены в параграфе 4.3.1.

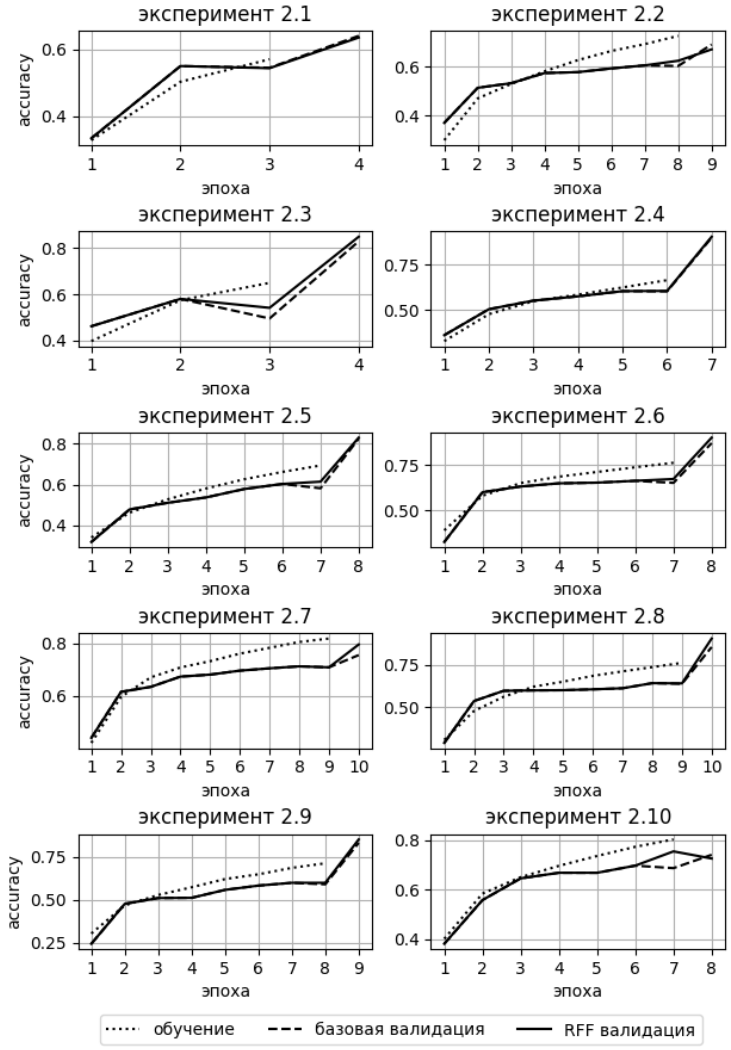


Рис 10. Результаты экспериментов 2.1-2.10

4.2.3. Эксперимент с глубокой нейронной сетью

В экспериментах №1.1-1.10 и №2.1-2.10 эмитировались слои глубоких ИНС, производящие сжатие карт признаков (то есть слои, расширяющие рецептивное поле). В данном разделе эксперимент повторяется для глубокой ИНС ResNet50.

Из датасета ImageNet были выбраны 10 случайных классов. Далее из указанного подмножества классов случайным образом были выбраны 2 класса для transfer learning. В ИНС ResNet50 в рамках RFF-обучения были зафиксированы параметры в слоях, понижающих размер карты признаков (таблица 9) : 8, 40, 82, 144. Шаги эксперимента указаны в таблице 9.

Таблица 9. Слои, понижающие размер карты признаков в ResNet50

Номер слоя в ResNet50	**Размер входной карты признаков**	**Размер выходной карты признаков**
8	(114,114)	(56,56)
40	(56,56)	(28,28)
82	(28,28)	(14,14)
144	(14,14)	(7,7)

Таблица 10. Схема эксперимента 3

Номер эпохи	**Модель 1**	**Модель 2**
0	инициализация параметров	
[1,$n-1$]	базовое обучение	
n	базовое обучение	RFF-обучение
$n+1$	transfer-learning	transfer-learning

Как и в экспериментах прошлых разделов, для валидации transfer learning отводилось 20% объектов из каждой группы классов. В качестве метрики оценивается accuracy при валидации (val_accuracy)

Таблица 11. Результаты эксперимента 3

Номер эксп-та	**Модель 1**	**Модель 2**	**delta**
3	0,6827	0,7346	**+0,052**

На графике ниже (рис.11) представлены показатели точности обучения и валидации. Анализ и выводы приведены в параграфе 4.3.1.

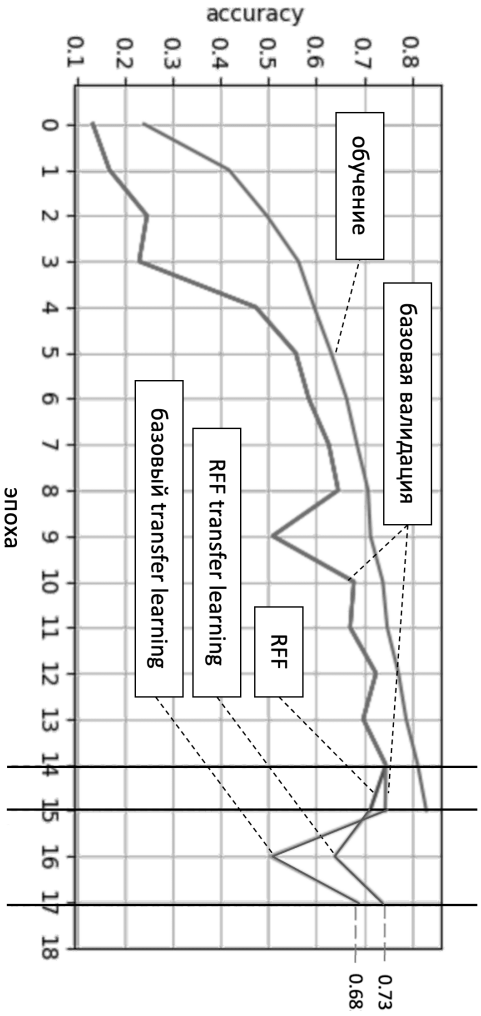


Рис 11. Результаты эксперимента 3

4.3. Результаты экспериментов

4.3.1. Увеличение точности

В таблице 12 представлены результаты всех экспериментов (п.4.2.). В рамках выбранных архитектур ИНС наблюдается увеличение обобщающей способности моделей.

Таблица 12. Результаты всех экспериментов

N	**тип модели**	**кол-во эпох**	**критерий остановки**	**M1**	**M2**	**delta**
1.1	базовая	6	число эпох	0,640	0,645	+ **0,005**
1.2	базовая	6	число эпох	0,590	0,595	+ **0,005**
1.3	базовая	6	число эпох	0,675	0,680	+ **0,005**
1.4	базовая	6	число эпох	0,765	0,760	$-0,005$
1.5	базовая	6	число эпох	0,700	0,745	+ **0,045**
1.6	базовая	6	число эпох	0,890	0,905	+ **0,015**
1.7	базовая	6	число эпох	0,795	0,800	+ **0,005**
1.8	базовая	6	число эпох	0,880	0,875	$-0,005$
1.9	базовая	6	число эпох	0,640	0,655	+ **0,015**
1.10	базовая	6	число эпох	0,790	0,825	+ **0,035**
2.1	базовая	4	переобучение	0,640	0,635	$-0,005$
2.2	базовая	9	переобучение	0,690	0,670	$-0,020$
2.3	базовая	4	переобучение	0,830	0,850	+ **0,020**
2.4	базовая	7	переобучение	0,900	0,905	+ **0,005**
2.5	базовая	8	переобучение	0,825	0,830	+ **0,005**
2.6	базовая	8	переобучение	0,870	0,900	+ **0,030**
2.7	базовая	10	переобучение	0,755	0,795	+ **0,040**
2.8	базовая	10	переобучение	0,855	0,905	+ **0,050**
2.9	базовая	9	переобучение	0,830	0,850	+ **0,020**
2.10	базовая	8	переобучение	0,740	0,725	$-0,015$
avarage delta						**+0,013**
3	ResNet50	17	число эпох	0.6827	0.7346	**+0,052**
avarage delta						**+0,052**

Увеличение обобщающей способности достигается на уровне слоев, формирующих карты признаков (поскольку эксперименты производятся в рамках задачи transfer learning).

Результаты, полученные в экспериментах с преодалением переобучения могут сигнализировать о повышении обобщающей способности модели за счет ослабления вклада отдельных окклюзивных признаков изображения.

В эксперименте с глубокой ИНС ResNet50 (п.4.2.3.) достигается более высокий прирост точности, чем в ИНС со значительно меньшей архитектурой (п.4.2.1, п.4.2.2.).

4.3.2. Оценка времени обучения

В экспериментах при переходе от модели 1 к модели 2 не зафиксировано повышение времени обучения. В таблице 13 демонстрируется результаты профилирования времени.

Предположительно, незначительные изменения скорости обучения (на всем числе эпох) вызваны техническими издержками на фиксирование и пропуск отдельных связей при распространении сигнала в ИНС.

Таблица 13. Изменение времени обучения

N	**тип модели**	**M1 (sec)**	**M2 (sec)**	**delta (sec)**
1.1 – 1.10	базовая	67.0	67.5	+0.7%
2.1 – 2.10	базовая	108.0	108.5	+0.4%
3	ResNet50	30499	30828	+0.1%

5. Вывод

Реализована техника, при которой обучение сверточной ИНС происходит с чередуемой остановкой обучения в слоях, производящих расширение рецептивного поля. На примере задачи transfer leaning продемонстрировано увеличение обобщающей способности модели во множестве экспериментов с базовой архитектурой ИНС.

Для глубокой архитектуры ResNet50 получено увеличение точности на 5% за счет остановки обучения всего для 4 слоев и в рамках всего 1 эпохи.

Список литературы

[1] Zeiler M.D., Fergus R., Fleet D., Pajdla T., Schiele B., Tuytelaars T., “Visualizing and Understanding Convolutional Networks.”, *Computer Vision – ECCV 2014.*, 2014, № 8689.

[2] Kortylewski A., He J., Liu Q., Yuille A., “Compositional Convolutional Neural Networks: A Deep Architecture with Innate Robustness to Partial Occlusion”, *CVPR 2020*, 2020.

[3] Kortylewski A., Liu Q., Wang A., Sun Y., Yuille A., "Compositional Convolutional Neural Networks: A Robust and Interpretable Model for Object Recognition under Occlusion", *International Journal of Computer Vision*, 2021, № 129, 736-760.

[4] Zhou B., Khosla A., Lapedriza A., Oliva A., Torralba A., "Learning Deep Features for Discriminative Localization", *CVPR*, 2016.

[5] Hinton, G., Vinyals, O., Dean, J., "Distilling the Knowledge in a Neural Network", *NIPS*, 2015.

[6] Wang T., Zhu J., Torralba A., Efros A., "Dataset Distillation", 2018..

[7] Sucholutsky, I., Schonlau M., "Soft-Label Dataset Distillation and Text Dataset Distillation", 2019.

[8] Medvedev D., D'yakonov A., "New Properties of the Data Distillation MethodWhen Working With Tabular Data", 2020.

[9] Meng-Hao Guo, Tian-Xing Xu, Jiang-Jiang Liu, Zheng-Ning Liu, Peng-Tao Jiang, Tai-Jiang Mu, Song-Hai Zhang, Ralph R. Martin, Ming-Ming Cheng, "Attention Mechanisms in Computer Vision: A Survey", *Computational Visual Media*, 2022, № 8, 331–368.

[10] Lindeberg T., "A computational theory of visual receptive fields", *Biological Cybernetics*, 2013, № 107, 589–635.

[11] Hubel DH, Wiesel TN, "Receptive fields of single neurones in the cat's striate cortex", *Physiol*, 1959, № 147, 226-238.

[12] Lindeberg T., "Normative theory of visual receptive fields", *Heliyon*, **1**:7 (2021).

[13] LeCun Y., Haffner P., Bottou L. Bengio Y., "Object Recognition with Gradient-Based Learning", *Lecture Notes in Computer Science*, 1999, № 1681.

[14] Хусаенов А.А.,, "Автоассоциативные нейронные сети в задаче классификации с усеченным множеством", *Интеллектуальные Системы. Теория и приложения*, **2**:26 (2022).

[15] Krizhevsky.A, Sutskever.I, Hinton G.E.,, "ImageNet Classification with Deep Convolutional Neural Networks", *Advances in Neural Information Processing Systems*, 2012, № 25.

[16] Araujo A., Norris W., Sim J.,, "Computing Receptive Fields of Convolutional Neural Networks", *Distill*, 2019.

[17] Luo W, Li Y., Urtasun R., Zemel R.,, "Understanding the Effective Receptive Field in Deep Convolutional Neural Networks", *Neural Information Processing Systems*, 2016, №29.

[18] Geman S., Bienenstock E., Doursat R., "Neural networks and the bias/variance dilemma", *Neural Computation*, 1992, №4.

[19] Lean Yu, Kin Keung Lai, Shouyang Wang, Wei Huang, "A Bias-Variance-Complexity Trade-Off Framework for Complex System Modeling", *Computational Science and Its Applications*, 2006.

[20] Alparslan Y., Alparslan K., Keim-Shenk J., Khade S., Greenstadt R.,, "Adversarial Attacks on Convolutional Neural Networks in Facial Recognition Domain", *IEEE Access*, 2021.

Learning parameters rotation method
Khusaenov A.A.

A method is proposed for improving the quality of training of convolutional artificial neural networks (ANN) by dividing the parameters according to their ability to expand the receptive field. For example, ResNet50 accuracy increase is achieved by only 4 layers freeze.

It is shown that the model generalizing ability increase is achieved by eliminating the excessive contribution of individual significant (occlusive) image elements in the feature maps. In favor of these assumptions the results of experiments in the transfer learning task and reasoning about the existence of this problem are presented.

The proposed approaches can be useful in training ANNs on small data or distillation of the training set, where the problems of overfitting on individual occlusive features are of high importance.

Keywords: convolutional artificial neural network, neuron receptive field, model overfitting problems, feature occlusion in convolutional artificial neural networks

References

[1] Zeiler M.D., Fergus R., Fleet D., Pajdla T., Schiele B., Tuytelaars T., "Visualizing and Understanding Convolutional Networks.", *Computer Vision – ECCV 2014.*, 2014, №8689.

[2] Kortylewski A., He J., Liu Q., Yuille A., "Compositional Convolutional Neural Networks: A Deep Architecture with Innate Robustness to Partial Occlusion", *CVPR 2020*, 2020.

[3] Kortylewski A., Liu Q., Wang A., Sun Y., Yuille A., "Compositional Convolutional Neural Networks: A Robust and Interpretable Model for Object Recognition under Occlusion", *International Journal of Computer Vision*, 2021, № 129, 736-760.

[4] Zhou B., Khosla A., Lapedriza A., Oliva A., Torralba A., "Learning Deep Features for Discriminative Localization", *CVPR*, 2016.

[5] Hinton, G., Vinyals, O., Dean, J., "Distilling the Knowledge in a Neural Network", *NIPS*, 2015.

[6] Wang T., Zhu J., Torralba A., Efros A., "Dataset Distillation", 2018..

[7] Sucholutsky, I., Schonlau M., "Soft-Label Dataset Distillation and Text Dataset Distillation", 2019.

[8] Medvedev D., D'yakonov A., "New Properties of the Data Distillation MethodWhen Working With Tabular Data", 2020.

[9] Meng-Hao Guo, Tian-Xing Xu, Jiang-Jiang Liu, Zheng-Ning Liu, Peng-Tao Jiang, Tai-Jiang Mu, Song-Hai Zhang, Ralph R. Martin, Ming-Ming Cheng, "Attention Mechanisms in Computer Vision: A Survey", *Computational Visual Media*, 2022, № 8, 331–368.

[10] Lindeberg T., "A computational theory of visual receptive fields", *Biological Cybernetics*, 2013, № 107, 589–635.

[11] Hubel DH, Wiesel TN, "Receptive fields of single neurones in the cat's striate cortex", *Physiol*, 1959, № 147, 226-238.

[12] Lindeberg T., "Normative theory of visual receptive fields", *Heliyon*, **1**:7 (2021).

[13] LeCun Y., Haffner P., Bottou L. Bengio Y., "Object Recognition with Gradient-Based Learning", *Lecture Notes in Computer Science*, 1999, № 1681.

[14] Khusaenov A.A., "Autoassociative neural networks in a classification problem with truncated dataset", *Intelligent systems. Theory and applications*, **2**:26 (2022) (In Russian).

[15] Krizhevsky.A, Sutskever.I, Hinton G.E.,, "ImageNet Classification with Deep Convolutional Neural Networks", *Advances in Neural Information Processing Systems*, 2012, № 25.

[16] Araujo A., Norris W., Sim J.,, "Computing Receptive Fields of Convolutional Neural Networks", *Distill*, 2019.

[17] Luo W, Li Y., Urtasun R., Zemel R.,, “Understanding the Effective Receptive Field in Deep Convolutional Neural Networks”, *Neural Information Processing Systems*, 2016, № 29.

[18] Geman S., Bienenstock E., Doursat R., “Neural networks and the bias/variance dilemma”, *Neural Computation*, 1992, № 4.

[19] Lean Yu, Kin Keung Lai, Shouyang Wang, Wei Huang, “A Bias-Variance-Complexity Trade-Off Framework for Complex System Modeling”, *Computational Science and Its Applications*, 2006.

[20] Alparslan Y., Alparslan K., Keim-Shenk J., Khade S., Greenstadt R.,, “Adversarial Attacks on Convolutional Neural Networks in Facial Recognition Domain”, *IEEE Access*, 2021.

Часть 2.
Специальные вопросы теории интеллектуальных систем

К вопросу о восстановлении изображения по стертому коду

Д. В. Алексеев[1]

Предлагается алгоритм для восстановления изображения по стертому коду, т.е. коду из которого удалена информация об индексах. Алгоритм обладает полиномиальной сложностью. Введено понятие треугольников общего положения. Для множества точек, образующих треугольники общего положения, доказана корректность работы алгоритма восстановления.

Ключевые слова: код изображения, стертый код, кодирование изображений, аффинная эквивалентность.

1. Введение

Одной из задач распознавания образов является задача распознавания зрительных образов. Одним из вариантов распознавания является сопоставление с известным образцом. Очевидно, изображение может быть сдвинуто, повернуто, иметь другой масштаб. При этом человечески глаз распознает эти изображения (см. рис. 1(a)). Таким образом имеет смысл отождествлять изображения, полученные путем применения преобразования из некоторого класса геометрических преобразований. В частности,одним из таких классов является множество аффинных преобразований. Этот класс является достаточно богатым и включает в себя сдвиги, повороты, растяжения. Заметим, что на практике, когда, например, изображение проектируется на сетчатку глаза или фоточувствительный элемент камеры, с плоским изображением происходят, вообще говоря, проективные преобразования. Впрочем, если расстояние до объекта велико по сравнению с его характерными размерами, то можно считать аффинные преобразования достаточно хорошим приближением проективных.

Каждое изображение может быть представлено как множество точек (см. рис. 1(b)). Заметим, что координатное представление этих точек, очевидно, зависит от выбора системы координат. Кроме того, оно не является инвариантным относительно геометрических преобразований.

[1] *Алексеев Дмитрий Владимирович* — кандидат физ.-мат. наук, с.н.с. лаборатории Проблем теоретической кибернетики мех.-мат. ф-та МГУ, e-mail: alekseev@intsys.msu.ru

Alekseev Dmitriy Vladimirovich — Candidate of Physical and Matematical Sciences, senior staff scientist, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Problems of Theorecical Cybernetics Lab.

Поэтому имеет смысл рассматривать другие представления, обладающие свойством инвариантности.

В.Н. Козлов в [7] ввел понятие *кода изображения*, являющегося *правильным* для какого-то семейства геометрических преобразований. Кодом является набор значений, инвариантный относительно этого семейства. Код является *правильным*, в случае, когда верно и обратное: из совпадение кодов вытекает эквивалентность фигур относительно данного семейства преобразований. Например, код, состоящий из попарных расстояний между точками, является правильным относительно движений плоскости. Там же в [7] был предложен код, состоящий из попарных отношений площадей треугольников, и доказана теорема о том, что такой код является правильным для множества невырожденных аффинных преобразований плоскости. В ходе доказательства теоремы был представлен алгоритм восстановления исходного изображения по его коду (с точностью до аффинной эквивалентности). Позднее П.Г. Агниашвили в [1] обобщил этот метод на случай n–мерного пространства. В [2] были выведены характеристические свойства кодов, т.е. условия, при которых некоторый набор чисел является кодом существующего изображения. Построение кода, правильного относительно невырожденных проективных преобразований плоскости, было сделано в [3].

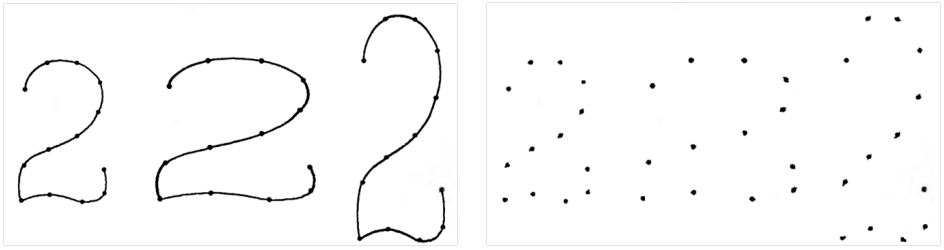
Стертым кодом называется код, который получен из кода изображения стиранием информации об индексах. Впервые задача была сформулирована акад. В.Б. Кудрявцевым более 40 лет назад следующим образом: Если оставить только попарные расстояния между точками изображения («палки»), и выстроить их в некотором порядке, например, возрастающем, и сделать огибающую у этой «гребенки», то можно ли осуществлять распознавание по виду огибающей? Позднее задача была переформулирована для кода, сохраняющего аффинную эквивалентность. На данный момент автору не известны работы, содержащие продвижения по решению этих задач.

Стертый код может быть использован, например, как показано в [4], для построения алгоритма цифровой подписи.

В данной работе рассматривается задача восстановления изображения по стертому коду с точностью до аффинной эквивалентности. Предложено определение *треугольников общего положения*. Предложен алгоритм восстановления и доказана его корректность при условии, что точки исходного изображения образуют треугольники общего положения. Алгоритм восстановления обладает полиномиальной сложностью.

Автор хотел бы выразить благодарность проф. В.Н. Козлову за внимание к работе. Также автор хотел бы выразить этой статьей дань памяти акад. В.Б. Кудрявцева.

Дальнейшая часть работы организована следующим образом. В разделе 2 даются необходимые определения и формулировка основного результата. В разделе 3 описан алгоритм восстановления изображения по стертому коду и доказана его корректность. В разделе 4 дана алгоритмическая сложность и описаны результаты численных экспериментов. В разделе 5 содержатся выводы и дальнейшие планы.



(a) Аффинные преобразования (b) Приближение точками

Рис. 1.

2. Основные понятия и формулировка результата

Определение 1. Изображением *будем называть непустое конечное множество точек на плоскости* $A = \{a_1, ..., a_n\}$.

Определение 2. Кодом *изображения* $A = \{a_1, ..., a_n\}$ *(при* $n \geq 3$*) будем называть множество* $\mathcal{R}(A) = \left\{\rho_{ijk,pqr} = \frac{S(\triangle a_i a_j a_k)}{S(\triangle a_p a_q a_r)}\right\}$[1]. *При этом элементы, полученные перестановками индексов* i, j, k *и* p, q, r *будем считать одним и тем же элементом кода, таким образом мощность множества* $|\mathcal{R}(A)| = (C_n^3)^2$.

Далее, в данной работе будут рассматриваться только такие изображения, в которых никакие три точки не расположены на одной прямой.

Свойства кодов изображения

1) $\rho_{ijk,ijk} = 1$;

2) $\rho_{pqr,ijk} = (\rho_{ijk,pqr})^{-1}$;

3) $\rho_{ijk,pqr} \times \rho_{pqr,uvw} = \rho_{ijk,uvw}$;

[1]Если точки a_p, a_q, a_r расположены на одной прямой, то ставим формальный символ $\rho_{ijk,pqr} = \infty$

4) $\pm\rho_{ijk,*} \pm \rho_{ikl,*} = \pm\rho_{ijl,*} \pm \rho_{jkl,*}$, где на месте «$*$» стоит произвольная (одна и та же) тройка, а знаки выбираются в соответствии с ориентацией треугольника, площадь которого стоит в знаменателе. Например, если вершины a_i, a_j, a_k идут против часовой стрелки, то $\rho_{ijk,*}$ идет со знаком «+», а иначе — со знаком «−».

В работе [2] было показано, что эти свойства кода являются в некотором смысле характеристическими. А именно, если зафиксировать ориентацию каждого треугольника и взять набор произвольных чисел $\rho_{ijk,pqr}$, удовлетворяющий свойствам 1-4, то по этому набору строится (причем однозначно) множество точек, кодом которого являются эти числа.

Определение 3. Стертым кодом *изображения* $A = \{a_1, ..., a_n\}$ *будем называть мультимножество* $\mathcal{Z}(A) = \{\rho_1, ..., \rho_M\}$, $M = (C_n^3)^2$, *полученное «стиранием» индексов во множестве* $\mathcal{R}(A)$.

Определение 4. *Будем говорить, что множество точек* A *образует множество треугольников общего положения* $\mathfrak{T}(A)$, *если выполнены свойства:*

1) *Никакие три точки не лежат на одной прямой, другими словами, все треугольники, образованные тройками точек из* A *— невырожденные;*

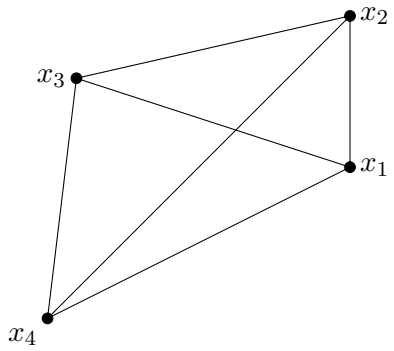
2) *Все нетривиальные (т.е. не* $\rho_{ijk,ijk} = 1$*) элементы кода* $\mathcal{R}(A) = \{\rho_{ijk,lmn}\}$ *попарно различны; Замечание: из этого вытекает, что площади всех треугольников, образованных точками из* A*, также различны;*

3) *Равенство* $S(\triangle_1) + S(\triangle_2) = S(\triangle_3) + S(\triangle_4)$ *выполняется только в случае, когда объединения треугольников* $\triangle_1 \cup \triangle_2$ *и* $\triangle_3 \cup \triangle_4$ *образуют один и тот же четырехугольник (см. рис. 2(a));*

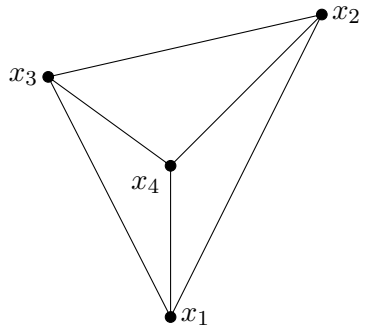
4) *Равенство* $S(\triangle_1) + S(\triangle_2) + S(\triangle_3) = S(\triangle_4)$ *выполняется только в случае, когда объединения треугольников* $\triangle_1 \cup \triangle_2 \cup \triangle_3$ *образует* $\triangle_4$ *(см. рис. 2(b));*

5) *Равенство* $\rho_{i_1j_1k_1,l_1m_1n_1} \cdot \rho_{i_2j_2k_2,l_2m_2n_2} = \rho_{i_2j_2k_2,l_2m_2n_2}$ *выполнено тогда и только тогда, когда* $\{l_1, m_1, n_1\} = \{i_2, j_2, k_2\}$ *или* $\{i_1, j_1, k_1\} = \{l_2, m_2, n_2\}$, *т.е. когда один и тот же треугольник присутствует в числителе и знаменателе.*

Другими словами, множество треугольников общего положения таково, что между площадями треугольников отсутствуют нетривиальные соотношения, кроме тех, которые вытекают из равносоставленности фигур.



(a) $\triangle x_1x_2x_3 \cup \triangle x_1x_3x_4 = \triangle x_1x_2x_4 \cup \triangle x_2x_3x_4$



(b) $\triangle x_1x_2x_4 \cup \triangle x_2x_3x_4 \cup \triangle x_1x_3x_4 = \triangle x_1x_2x_3$

Рис. 2. Треугольники общего положения

Определение 5. *Если множество A' не аффинно эквивалентно множеству A, но обладает идентичным стертым кодом $\mathcal{Z}(A) \equiv \mathcal{Z}(A')$, то A' называем призраком множества A.*

Замечание 1. *Отношение «быть призраком» симметрично, т.е. A является призраком A'.*

Гипотеза У множества треугольников общего положения призраков не существует.

Теорема 1. *Пусть A образует множество треугольников общего положения. Существует алгоритм, который по стертому коду $\mathcal{Z}(A)$ либо строит изображение аффинно-эквивалентно исходному A, либо два изображения, одно из которых аффинно-эквиваалентно A, а другое является его призраком A'.*

3. Описание алгоритма восстановления и доказательство его корректности.

Для обоснования алгоритма докажем несколько вспомогательных утверждений.

Лемма 1. *Рассмотрим произвольные треугольники $\triangle', \triangle'' \in \mathfrak{T}(A)$. Треугольники $\triangle'$ и $\triangle''$ имеют общую сторону тогда и только тогда, когда существуют $\triangle_i, \triangle_j \in \mathfrak{T}(A)$ и $e_l \in \{\pm 1\}, l = 1, 2, 3, 4$, такие, что*

$$e_1S(\triangle') + e_2S(\triangle'') + e_3S(\triangle_i) + e_4S(\triangle_j) = 0$$

$A = \{a_1, \ldots, a_n\}$	Множество точек (изображение)
$A' = A \setminus \triangle_{\min}$	Множество точек без вершин минимального треугольника
$\mathcal{R}(A)$	Код изображения A
$\mathcal{Z}(A)$	Стертый код изображения A
$\mathfrak{S}(A)$	Множество пар $\rho^{(1)}, \rho^{(2)} \in \mathcal{Z}(A)$, для которых выполнено $\rho^{(1)} \cdot \rho^{(2)} = \rho^*$
$\mathfrak{T}(A)$	Множество всех треугольников с вершинами из A
$\mathfrak{T}'(A)$	Множество всех треугольников $\mathfrak{T}(A)$ без минимального
$\mathfrak{P}(A)$	Множество троек, соответствующих точкам A
$\mathcal{T}^*(A)$	Подмножество $\mathfrak{T}(A)$ — те треугольники, которые входят в тройки $\mathfrak{P}(A)$
β	Биекция между треугольниками $\mathfrak{T}'(A)$ и множеством пар $\mathfrak{S}(A)$
τ^*	Биекция между точками A' и соответствующими тройками

Таблица 1. Список обозначений

Доказательство. Если в равенстве $\pm S(\triangle') \pm S(\triangle'') \pm S(\triangle_i) \pm S(\triangle_j) = 0$ знаков «плюс» больше, чем знаков «минус», то домножим его на -1. Поскольку все знаки не могут быть одинаковыми, то в итоге получим либо равенство вида $S_1 + S_2 = S_3 + S_4$, либо вида $S_1 + S_2 + S_3 = S_4$, где $S_k \in \{S(\triangle'), S(\triangle''), S(\triangle_i), S(\triangle_j)\}, k = 1, 2, 3, 4$. В первом случае по свойству 3 треугольников общего положения получим, что любые два треугольника из этого множества имеют общую сторону. А во втором случае аналогичное утверждение вытекает из свойства 4.

В обратную сторону, пусть треугольники $\triangle_{i_1 i_2 i_3}$ и $\triangle_{i_1 i_2 i_4}$ имеют общую сторону $a_{i_1} a_{i_2}$. Точки a_{i_3} и a_{i_4} могут лежать по одну сторону от прямой $(a_{i_1} a_{i_2})$ или по разные стороны от нее. Выпуклая оболочка всех четырех точек может представлять собой треугольник или четырехугольник. Получаем 4 случая:

- Случай 1 (a_{i_3} и a_{i_4} по одну сторону, оболочка — треугольник): Не ограничивая общности, можно считать, что выпуклая оболочка это $\triangle_{i_1 i_2 i_3}$, в этом случае $S_{i_1 i_2 i_3} = S_{i_1 i_2 i_4} + S_{i_2 i_3 i_4} + S_{i_3 i_1 i_4}$ (см. рис. 3(a)).
- Случай 2 (a_{i_3} и a_{i_4} по одну сторону, оболочка — четырехугольник): Не ограничивая общности, можно считать, что порядок вершин при обходе выпуклой оболочки — $a_{i_1}, a_{i_2}, a_{i_3}, a_{i_4}$, следовательно $S_{i_1 i_2 i_3} + S_{i_1 i_3 i_4} = S_{i_2 i_3 i_4} + S_{i_3 i_4 i_1}$ (см. рис. 3(b)).

- Случай 3 (a_{i_3} и a_{i_4} по разные стороны, треугольник): Не ограничивая общности, можно считать, что выпуклая оболочка — $\triangle_{i_1 i_3 i_4}$, в этом случае $S_{i_1 i_3 i_4} = S_{i_1 i_2 i_3} + S_{i_1 i_2 i_4} + S_{i_2 i_3 i_4}$ (см. рис. 3(c)).

- Случай 4 (a_{i_3} и a_{i_4} по разные стороны, четырехугольник): Не ограничивая общности, можно считать, что порядок вершин при обходе выпуклой оболочки — $a_{i_1}, a_{i_3}, a_{i_2}, a_{i_4}$, следовательно $S_{i_1 i_2 i_3} + S_{i_1 i_2 i_4} = S_{i_2 i_3 i_4} + S_{i_1 i_3 i_4}$ (см. рис. 3(d)).

□

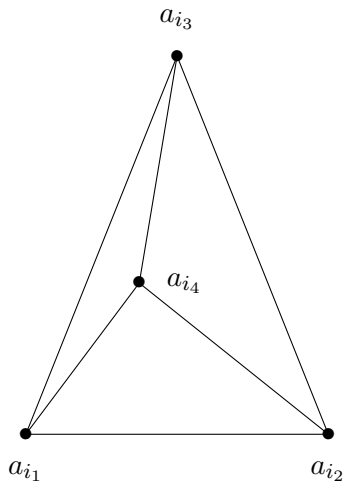
Определение 6. *Зафиксируем треугольник $\triangle^* \in \mathfrak{T}(A)$, не содержащий точек из A внутри и будем называть его центральным. Рассмотрим множество треугольников из $\mathfrak{T}(A)$, имеющих общую сторону с центральным. Можно разбить их на тройки следующим образом. Пусть $\triangle* = \triangle a_1 a_2 a_3$, для произвольной точки $a_i, i = 4, 5, ..., n$ рассмотрим треугольники $\triangle_{1i} = \triangle a_2 a_3 a_i$, $\triangle_{2i} = \triangle a_1 a_3 a_i$, $\triangle_{3i} = \triangle a_1 a_2 a_i$. Будем называть такую тройку $T_i = \{\triangle_{1i}, \triangle_{2i}, \triangle_{3i}\}$* соответствующей *вершине a_i. Будем обозначать $\mathfrak{P}(A)$ множество всех таких троек и обозначать соответствие $\tau^* : A^* = A \setminus \{a_1, a_2, a_3\} \to \mathfrak{P}(A)$. Множество всех треугольников, входящих в соответствующие тройки будем обозначать $\mathcal{T}^*(A)$.*

Замечание 2. *Отображение τ^* является биективным по построению. Действительно, взяв разные вершины a_i, a_j, $i \neq j$, получим разные тройки. С другой стороны, множество $\mathfrak{P}(A)$ задано как образ этого отображения.*

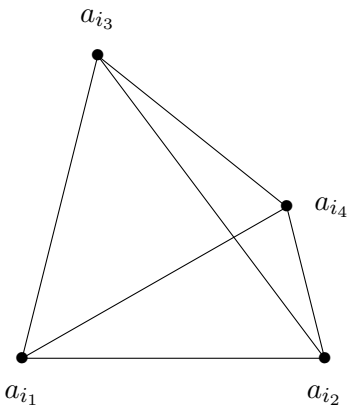
Рассмотрим тройку с вершинами a_1, a_2, a_3, a_i, $i = 4, ..., n$. Найдем выпуклую оболочку этих точек. Очевидно, она может состоять из 3 или 4 точек.

- Допустим, что она состоит из 3 точек. Очевидно, a_i не может лежать внутри по выбору центрального треугольника. Значит внутри лежит вершина центрального треугольника, не ограничивая общности, допустим, вершина a_3 (см. рис. 4(a)). В этом случае будем называть эту тройку тройкой 1-го типа, а треугольник $\triangle_{12i}$ — главным (для этой тройки).

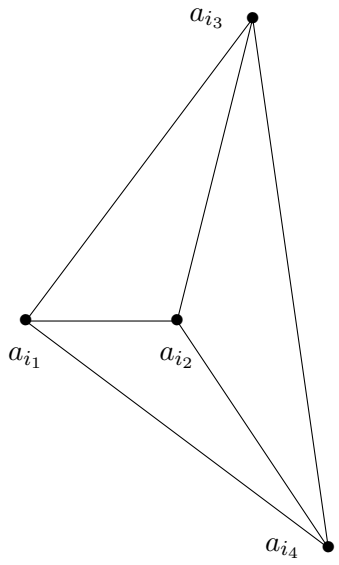
- Допустим, что она состоит из 4 точек. Очевидно, одна из сторон центрального треугольника будет диагональю этого четырехугольника. Не ограничивая общности, допустим, что это сторона $a_1 a_2$ (см. рис. 4(b)). В этом случае будем называть эту тройку тройкой 2-го типа, а треугольник $\triangle_{12i}$ — главным (для этой тройки).



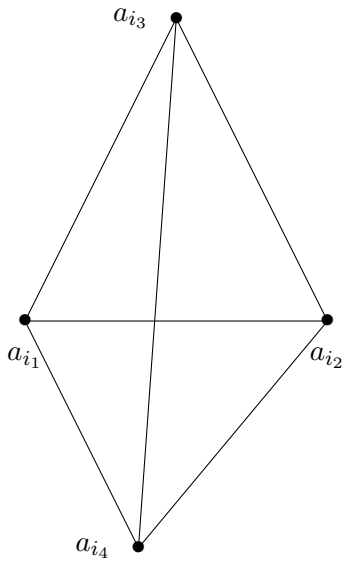
(a) Случай 1



(b) Случай 2



(c) Случай 3



(d) Случай 4

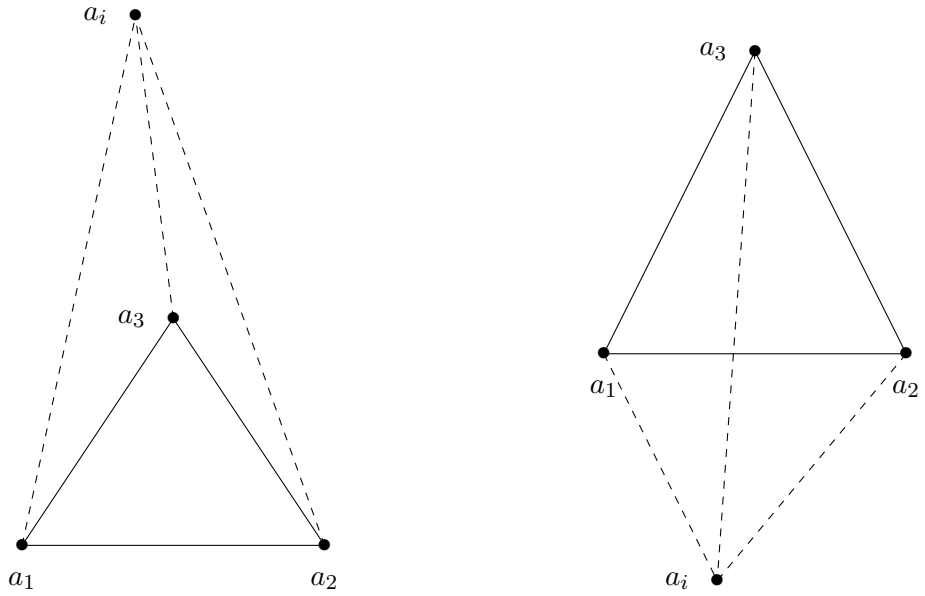
Рис. 3. К доказательству леммы 1 .

Заметим, что для троек первого типа выполнено равенство $S(\triangle_i^{(m)}) = S(\triangle^*) + S(\triangle') + S(\triangle'')$, а для второго типа — равенство $S(\triangle_i^{(m)}) = -S(\triangle^*) + S(\triangle') + S(\triangle'')$ где $\triangle^*$ — центральный треугольник, $\triangle_i^{(m)}$ — главный треугольник тройки, а $\triangle', \triangle''$ — два оставшихся

Это свойство троек, является, в некотором роде, характеристическим, как показано в следующей лемме.

Лемма 2. *Если треугольники* $\triangle^{(m)}, \triangle', \triangle'' \in \mathfrak{T}(A) \setminus \{\triangle^*\}$ *удовлетворяют соотношению* $S(\triangle^{(m)}) = \pm S(\triangle^*) + S(\triangle') + S(\triangle'')$*, то найдется* $a_i \in A^*$*, для которой* $\tau^*(a_i) = \{\triangle_i^{(m)}, \triangle', \triangle''\} \in \mathfrak{P}(A)$*, т.е. эта тройка треугольников является соответствующей.*

Доказательство. Очевидно вытекает из свойств 3 и 4 множества треугольников общего положения.



(a) Первый тип (b) Второй тип

Рис. 4. Примеры соответствующих троек.

Введем отношение смежности на множестве $\mathcal{T}^*(A)$. Будем называть смежными треугольники, не входящие в одну тройку, у которых имеется общая сторона. Заметим, что если треугольники не входят в одну тройку, то общая сторона может быть только одной из из сторон центрального треугольника $\triangle^*$. Таким образом, это отношение разбивает множество $\mathcal{T}^*(A)$ на три класса эквивалентности.

Замечание 3. *Свойство «иметь общую сторону» легко проверить, используя лемму 1 .*

Лемма 3. *Построим прямые* a_1a_2*,* a_2a_3*,* a_1a_3*. Они разобьют плоскость на 6 областей (исключая центральный треугольник). Для каждой соответствующей тройки* T_i *по типу и классу эквивалентности главного треугольника однозначно определяется, в какой из областей расположена точка* a_i *(см. рис. 5).*

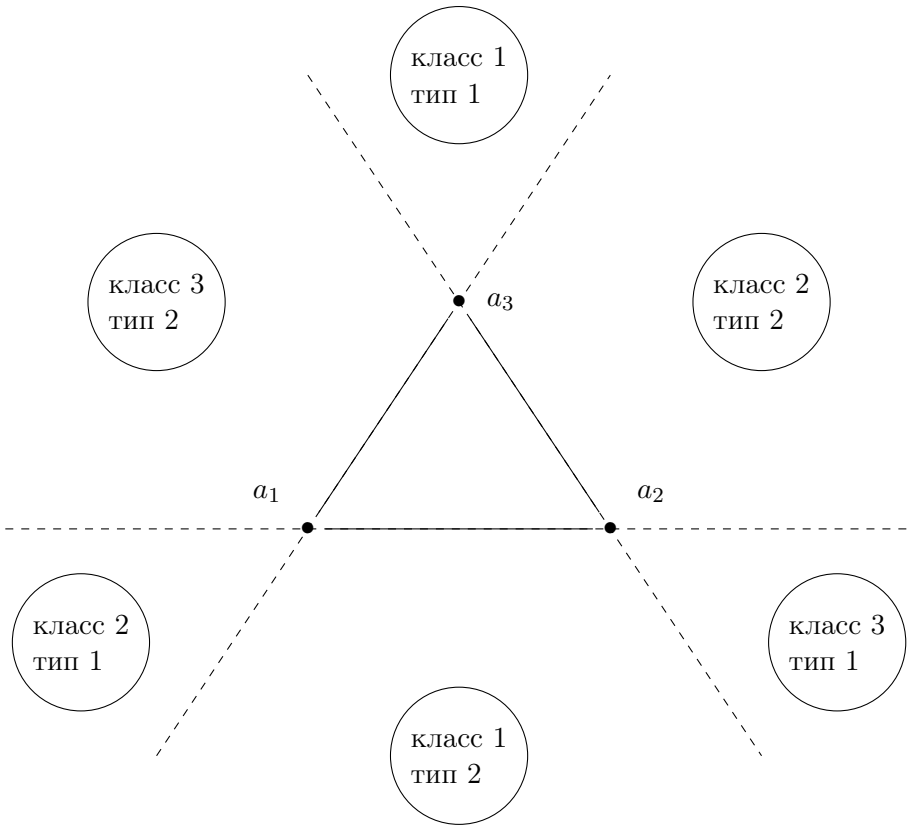


Рис. 5. Положение вершины a_i в зависимости от типа и класса.

Доказательство. Например, рассмотрим тройку первого типа, не ограничивая общности можно считать, что главный треугольник прилегает к стороне a_1a_2. Из равенства

$$S(\triangle_{12i}) = S(\triangle_{123}) + S(\triangle_{13i}) + S(\triangle_{23i})$$

вытекает, что этот треугольник полностью содержит $\triangle_{123}$, следовательно, его третья вершина a_i содержится в угле, образованном лучами $[a_1; a_3[$ и $[a_2; a_3[$ (см. рис. 5). Остальные случаи рассматриваются аналогично. □

Замечание 4. *Предполагается, что нумерация классов эквивалентности соответствует нумерации сторон: a_1a_2 — первый класс, a_2a_3 — второй, a_3a_1 — третий.*

Замечание 5. *Здесь и далее в статье предполагается, что $\mathfrak{T}(A)$ — множество треугольников общего положения.*

Будем обозначать $\mathfrak{T}(A)$ множество всех треугольников, образованных точками множества A. Площади треугольников различны, обозначим $\triangle_{\min}, \triangle_{\max} \in \mathfrak{T}(A)$ треугольники наибольшей и наименьшей площади, соответственно. Индексы точек, соответствующих этим треугольникам будем обозначать $(i_{\min}, j_{\min}, k_{\min})$ и $(i_{\max}, j_{\max}, k_{\max})$. Обозначим $\rho^* = \frac{S(\triangle_{\max})}{S(\triangle_{\min})}$. Аффинным преобразование отобразим $\triangle_{\min}$ в треугольник единичной площади и будем в-дальнейшем полагать $S_{\min} = 1, S_{\max} = \rho^*$. Докажем (в этих предположениях) следующие вспомогательные леммы:

Лемма 4. *Пусть $\mathfrak{T}(A)$ — множество треугольников общего положения, $\mathcal{Z}(A)$ — стертый код, соответствующий их вершинам. Существует биекция β между множеством треугольников*

$$\mathfrak{T}'(A) = \mathfrak{T}(A) \setminus \{\triangle_{\min}, \triangle_{\max}\}$$

и множеством неупорядоченных пар

$$\mathfrak{S}(A) = \left\{(\rho^{(1)}, \rho^{(2)}) \,:\, \rho^{(1)}, \rho^{(2)} \in \mathcal{Z}(A) \setminus \{1, \rho^*\};\, \rho^{(1)} \cdot \rho^{(2)} = \rho^*\right\}$$

Доказательство. Построим отображение β следующим образом: Пусть $\triangle = \triangle_{ijk} \in \mathfrak{T}'(A)$ — произвольный треугольник, очевидно,

$$\frac{S(\triangle_{\max})}{S(\triangle)} \cdot \frac{S(\triangle)}{S(\triangle_{\min})} = \frac{S(\triangle_{\max})}{S(\triangle_{\min})} = \rho^*$$

и можно взять в качестве $\beta(\triangle)$ пару:

$$(\rho^{(1)}, \rho^{(2)}) = \left(\frac{S(\triangle_{\max})}{S(\triangle)} = \rho_{i_{\max}j_{\max}k_{\max},ijk}, \frac{S(\triangle)}{S(\triangle_{\min})} = \rho_{ijk,i_{\min}j_{\min}k_{\min}}\right).$$

Покажем, что отображение β является инъекцией. Предположим обратное: $\beta(\triangle) = \beta(\triangle') = (\rho^{(1)}, \rho^{(2)})$ для некоторых $\triangle \neq \triangle'$. Но это противоречит свойству 2 треугольников общего положения — для разных треугольников получаются одинаковые коды.

Покажем, что отображение β является сюръекцией. Предположим, что найдется пара $(\rho^{(1)}, \rho^{(2)})$, не принадлежащая образу $\beta(\mathfrak{T}'(A))$, такая, что $\rho^{(1)} \cdot \rho^{(2)} = \rho^*$. По свойству 5 это означает, что для некоторых треугольников $\triangle_1, \triangle_2, \triangle_3$ выполнено равенство $\frac{S(\triangle_1)}{S(\triangle_2)} \cdot \frac{S(\triangle_2)}{S(\triangle_3)} = \rho^*$, т.е. $\frac{S(\triangle_1)}{S(\triangle_3)} = \frac{S(\triangle_{\max})}{S(\triangle_{\min})}$. Из этого следует, что $\triangle_1 = \triangle_{\max}$ и $\triangle_3 = \triangle_{\min}$, поскольку для любой другой пары треугольников отношение площадей меньше ρ^*, а значит $\beta(\triangle_2) = (\rho^{(1)}, \rho^{(2)})$. □

Лемма 5. *Пусть $\mathfrak{T}(A)$ — множество треугольников общего положения. Выберем произвольные $\triangle_{ijk}, \triangle_{pqr} \in \mathfrak{T}'(A)$, $\triangle_{ijk} \neq \triangle_{pqr}$. Обозначим $\beta(\triangle_{ijk}) = (\rho^{(1)}, \rho^{(2)})$, $\beta(\triangle_{pqr}) = (\rho^{(3)}, \rho^{(4)})$, пусть для определенности $S_{ijk} = \rho^{(1)}$. Тогда из чисел $\rho' = \frac{\rho^{(3)}}{\rho^{(1)}}$ и $\rho'' = \frac{\rho^{(4)}}{\rho^{(1)}}$ одно и только одно принадлежит стертому коду $\mathcal{Z}(A)$.*

Доказательство. Из условия $\beta(\triangle_{pqr}) = (\rho^{(3)}, \rho^{(4)})$ вытекает, что либо $S(\triangle_{pqr}) = \rho^{(3)}$, либо $S(\triangle_{pqr}) = \rho^{(4)}$. Если $S(\triangle_{pqr}) = \rho^{(3)}$, то в коде должен присутствовать $\rho' = \frac{\rho^{(3)}}{\rho^{(1)}}$, а если $S(\triangle_{pqr}) = \rho^{(4)}$, то $\rho'' = \frac{\rho^{(4)}}{\rho^{(1)}}$. Одновременно два таких элемента присутствовать не могут. Действительно, предположим для определенности, что в исходном изображении $S_{pqr} = \rho^{(3)}$. Тогда

$$\rho'' = \frac{\rho^{(4)}}{\rho^{(1)}} = \frac{\rho^*}{\rho^{(1)}\rho^{(3)}} = \frac{\rho^{(2)}}{\rho^{(3)}},$$

следовательно, $\rho'' \cdot \rho^{(3)} = \rho^{(2)}$, что противоречит свойству 5. □

3.1. Алгоритм восстановления изображения по стертому коду

1) Выберем в качестве центрального треугольник минимальной площади. Поскольку восстановление происходит с точностью до аэквивалентности, то можно назначить минимальному треугольнику любую площадь, например $S(\triangle_{\min}) = S_{\min} = 1$ для определенности. Тогда площадь наибольшего треугольника определена однозначно $S(\triangle_{\max}) = S_{\max} = \rho^*$. Заметим, что ρ^* известно, т.к. это максимальный элемент стертого кода.

2) Зафиксируем одну из пар $(\rho^{(1)}, \rho^{(2)}) \in \mathfrak{S}(A)$. По лемме 4 ей соответствует треугольник $\triangle_{ijk} = \beta^{-1}(\rho^{(1)}, \rho^{(2)})$. Возможно два варианта

$S_{ijk} = \rho^{(1)}$ или $S_{ijk} = \rho^{(2)}$. В дальнейшем построение идет параллельно для обоих вариантов. В одном из них должно получиться исходное изображение. Для определенности будем считать, что $\frac{S_{ijk}}{S_{min}} = S_{ijk} = \rho^{(1)}$, второй вариант рассматривается аналогично.

3) Перебираем все пары $(\rho^{(3)}, \rho^{(4)}) \in \mathfrak{S}(A)$. По лемме 4 в исходном изображении найдется треугольник $\triangle_{pqr} = \beta^{-1}(\rho^{(3)}, \rho^{(4)})$. По лемме 5 либо $\rho' = \frac{\rho^{(3)}}{\rho^{(1)}} \in \mathcal{Z}(A)$, тогда $S(\triangle_{pqr}) = \rho^{(3)}$, либо $\rho'' = \frac{\rho^{(4)}}{\rho^{(1)}} \in \mathcal{Z}(A)$ и тогда $S(\triangle_{pqr}) = \rho^{(4)}$, Повторив эту процедуру для всех пар из $\mathfrak{S}(A)$, получаем список площадей всех треугольников из $\mathfrak{T}'(A)$. А $S(\triangle_{\min}), S(\triangle_{\max})$ уже известны из первого шага алгоритма.

4) Выберем в качестве центрального $\triangle_{\min}$ (очевидно, не содержащий внутри себя точек из A). По лемме 2 найдем все тройки $\mathfrak{P}(A)$, соответствующие точкам $A' = A \setminus \{a_1, a_2, a_3\}$ (т.е. кроме вершин центрального треугольника). В каждой тройке выделим главный треугольник $\triangle_i^{(m)}$, $i = 4, ..., n$.

5) Разбиваем треугольники $\mathcal{T}^*(A)$ на классы эквивалентности. Треугольники эквивалентны тогда и только тогда, когда а) Они не содержатся в одной тройке; б) Имеют общую сторону (что можно проверить по лемме 1).

6) Для каждой тройки из $T_i = \mathfrak{P}(A)$ определим по лемме 5 в какой области находится соответствующая точка. Таким образом, мы получим, в какой полуплоскости относительно прямых b_1b_2, b_2b_3 и b_3b_1 будет расположена восстановленная точка. Если выбрать восстановленные точки как $b_1(0,0)$, $b_2(1,0)$, $(0,1)$, то это позволяет определить знаки координат (см. рис. 6).

7) Зафиксируем на плоскости треугольник $\triangle b_1b_2b_3$ и будем считать, что это образ центрального треугольника $\triangle_{\min}$ при некотором аффинном преобразовании. Для каждой тройки из $\mathfrak{P}(A)$ определим к каким сторонам центрального треугольника прилегают главный и побочные треугольники. Если известна площадь $S_i = S(\triangle_{12i})$, то расстояние точки b_i до прямой b_1b_2 равно $d_i = 2S_i/|b_1b_2|$. Геометрическим местом таких точек являются две прямые, параллельные b_1b_2 и проходящие на расстоянии d_i от нее. Но, поскольку известно (из предыдущего пункта), в какой полуплоскости находится b_i, то остается только одна прямая. Проделав аналогичные построения для стороны b_2b_3, получим вторую прямую, не параллельную первой. Очевидно, что точка их пересечения и есть b_i.

8) Находим стертый код восстановленного изображения и сравниваем с $\mathcal{Z}(A)$. Если коды не совпадают (такое возможно в случае, если пошли по неправильной ветке в пункте 2), то отбрасываем результат.

Корректность алгоритма вытекает из доказанных ранее лемм.

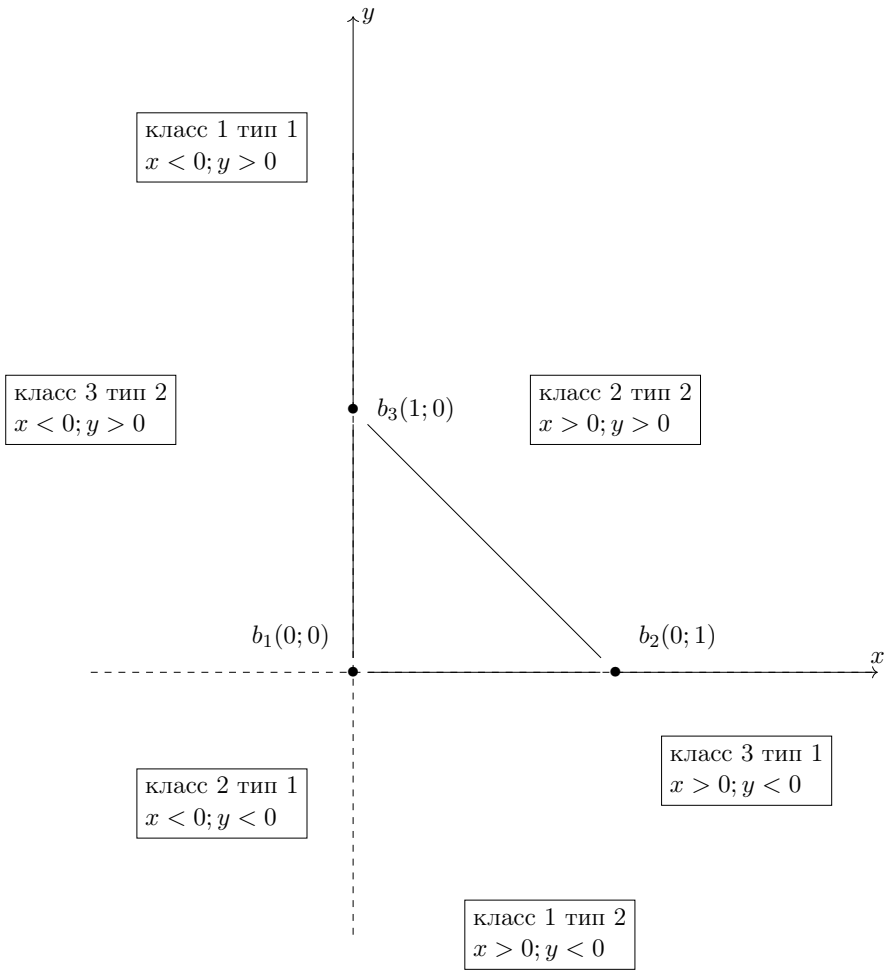


Рис. 6. Определение знака координат по классу и типу тройки (шаг 6).

Шаг	**Сложность**	**Комментарий**
1	$O(n^6)$	Перебор элементов стертого кода $\lvert\mathcal{Z}(A)\rvert = C^2_{C^3_n}$
2	$O(1)$	Фиксируем пару из $\mathfrak{S}(A)$
3	$O(n^3)$	Перебор пар из $\lvert\mathfrak{S}(A)\rvert = C^3_n$
4	$O(n^9)$	Перебор троек треугольников $C^3_{C^3_n}$
5	$O(n^7)$	Разбиение троек из $\lvert\mathfrak{P}(A)\rvert = n-3$ на классы эквивалентности[2]
6	$O(n)$	Перебор точек и определение положения относительно $\triangle^*$
7	$O(n)$	Перебор точек и восстановление координат
8	$O(n^6)$	Нахождение стертого кода восстановленного изображения и сравнение с исходным кодом

Таблица 2. Оценка алгоритмической сложности

4. Алгоритмическая сложность и эксперименты

Алгоритмическая сложность оценивается в случае, когда все точки множества A обладают целочисленными координатами. В этом случае площади всех треугольников целые или полуцелые, следовательно, все элементы стертого кода являются рациональными. Модель вычислителя предполагает сложность 1 для любого арифметического действия с рациональными числами произвольной точности. Это достаточно адекватно моделирует случай, когда и числитель, и знаменатель дроби помещаются в регистр вычислительного устройства (CPU).

Наиболее трудоемкой является процедура перебора троек треугольников на этапе 4 (см. таблицу 2). Количество операций сравнения равно $2 \cdot C^3_{C^3_n} = O(n^9)$, где n — количество точек изображения. Сложность, хотя и является полиномиальной по n, не позволяет проводить алгоритм для большого количества точек (например, $n = 100$).

Алгоритм был реализован в виде скрипта для Matlab (версия R2020b). Краткое описание скрипта:

- Случайным образом порождает координаты n точек;
- Строит стертый код для этих точек;
- Применяет алгоритм из раздела 3.1;
- Проводит замер времени;

n	t (сек.)
8	0.6565
9	2.1726
10	5.7466
11	15.595
12	35.471
13	70.919
14	153.00
15	280.39
16	542.05
17	954.20
18	1489.2
19	2553.6

Таблица 3. Время работы алгоритма

- Проверяет корректность восстановления с точностью до аффинной эквивалентности.

Скрипт выложен в общий доступ и доступен по адресу `https://github.com/dvalex/erased_code_recovery`.

Тестирование проводилось на компьютере с процессором Intel® Core™ i7-8700 (тактовая частота 3.20GHz), объем оперативной памяти 32GB, ОС Windows 10 (версия 22H2), Проводилось восстановление для $n = 8, 9, ..., 19$ с замером времени выполнения (1 запуск для каждого n).

Результаты приводятся в таблице 3 и изображены на рис. 7.

Замечание 6. *В алгоритме шаг 4 (перебор троек треугольников) обладает наибольшей вычислительной сложностью. Этот шаг может быть выполнен параллельно, что, впрочем, не снижает асимптотическую сложность.*

5. Заключение: выводы и дальнейшие планы.

Итогом данной работы является создание алгоритма для восстановления изображения по стертому коду за полиномиальное время и доказательство корректности его работы при условии треугольников общего положения. Сделана оценка алгоритмической сложности показано ее соответствие экспериментальным данным.

Возможные направления дальнейшего развития:

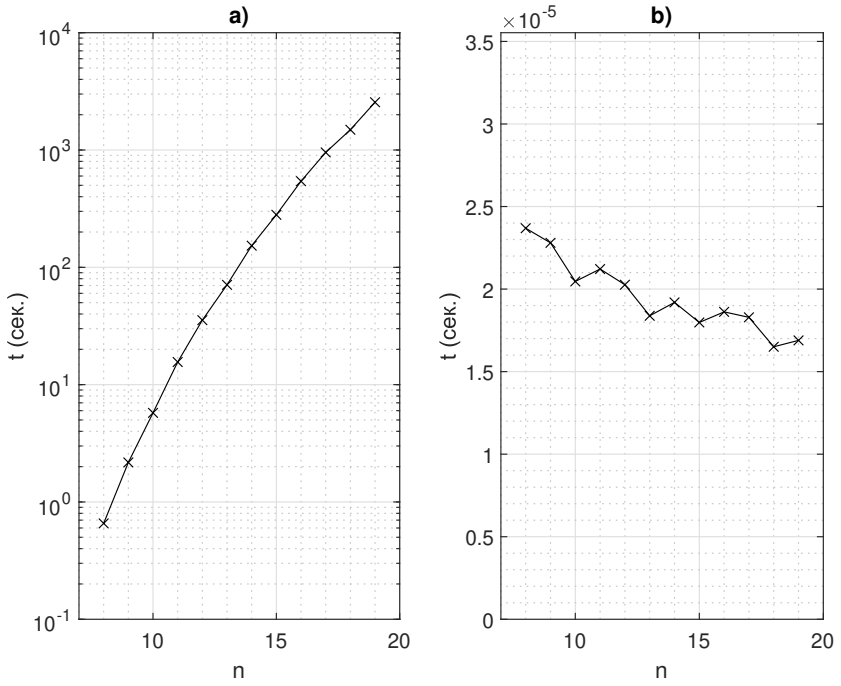


Рис. 7. а) Время выполнения в зависимости от числа точек; б) Время выполнения деленное на $C^3_{C^3_n}$.

1) Ослабить условия на множество треугольников общего положения — сейчас они выглядят не очень естественно и сложными для проверки. Поскольку для проверки требуется перебор четверок треугольников, то алгоритмическая сложность составляет $O(n^{12})$, т.е. превосходит сложность решения самой задачи!

2) В случае целочисленных координат, лежащих в определенном диапазоне, оценить вероятность генерации множества треугольников общего положения.

3) Сделать построение однозначным. Например, доказать, что у изображения в общем положении нет призраков или вывести достаточные условия, при которых изображение не может иметь призрака.

4) Снизить алгоритмическую сложность, которая в текущей версии составляет $O(n^9)$.

5) Реализовать алгоритм с использованием рациональных чисел произвольной точности (например, на Python). Сейчас используется встроенный в Matlab тип «double», сравнение таких чисел производится приближенно, с точностью до 10^{-8}.

6) Применить аналогичный подход к решению изначальной задачи (про попарные расстояния между точками)

Список литературы

[1] Агниашвили П.Г., "Однозначность восстановления изображения по его коду в n-мерном случае", *Интеллектуальные системы.*, **15**:1–4 (2011), 293–332.

[2] Алексеев Д.В., "Необходимые и достаточные условия существования изображения с заданным кодом", *Интеллектуальные системы. Теория и приложения (ранее: Интеллектуальные системы по 2014, № 2, ISSN 2075-9460)*, **24**:2 (2020), 55–66.

[3] Алексеев Д.В., "Кодирование изображений, инвариантное относительно проективных преобразований.", *Интеллектуальные системы. Теория и приложения.*, **13**:1–4 (2009), 35–40.

[4] Козлов В.Н., *Способ аутентификации электронного изображения.*, Патент на изобретение № 2779379., Дата государственной регистрации в Государственном реестре изобретений Российской Федерации 06 сентября 2022 года.

[5] Козлов В.Н., "Доказательность и эвристика при распознавании визуальных образов", *Интеллектуальные системы.*, **14**:1–4 (2010), 35–52.

[6] Козлов В.Н., *Элементы математической теории зрительного восприятия*, Изд–во ЦПИ при мех.–мат. ф–те МГУ, Москва, 2001, 128 с.

[7] Козлов В.Н., "О кодировании дискретных фигур", *Дискретная математика*, **8**:6 (1996), 57–61.

[8] Kozlov V.N., "Image Coding and Recognition and Some Problems of Stereovision", *Pattern Recognition and Image Analysis*, **7**:4 (1997), 448–466.

To the question of restoring an image from an erased code
Alekseev D.V.

The article proposes an algorithm for image recovery by erased code, i.e. the code without information about indices. The algorithm has a polynomial complexity. The definition of general position triangles is given. The algoritm correctness is proved for wide class of points set which form general positions triangles.

Keywords: image code, image encoding, erased code, affine equivalence.

References

[1] Agniashvili P.G., "Unique image reconstruction from its code in the n-dimensional case", *Интеллектуальные системы.*, **15**:1–4 (2011), 293–332 (in Russian).

[2] Alekseev D.V., "Necessary and sufficient conditions for the existence of an image with a given code", *Intelligent Systems. Theory and Applications (formerly: Intelligent Systems by 2014, №2, ISSN 2075-9460)*, **24**:2 (2020), 55–66 (in Russian).

[3] Alekseev D.V., "Image encoding invariant with respect to projective transformations.", *Intelligent Systems.*, **13**:1–4 (2009), 35–40 (in Russian).

[4] Kozlov V.N., *A way of authenticating a digital image.*, Patent for inventione № 2779379., Date of the official registration in the State Register of Inventions of the Russian Federation on September 06, 2022 (in Russian).

[5] Kozlov V.N., "Evidence and heuristics in visual pattern recognition", *Intelligent systems*, **14**:1–4 (2010), 35–52 (in Russian).

[6] Kozlov V.N., *Elements of the mathematical theory of visual perception*, Central Publishing House at the Faculty of Mechanics and Mathematics, Moscow State University, Moscow, 2001 (in Russian), 128 c.

[7] Kozlov V.N., "About discrete images encoding", *Discrete Mathematics*, **8**:6 (1996), 57–61 (in Russian).

[8] Kozlov V.N., "Image Coding and Recognition and Some Problems of Stereovision", *Pattern Recognition and Image Analysis*, **7**:4 (1997), 448–466.

О сложности перехода к правильному линейному виду

П. С. Дергач[1] Д. А. Сальцова[1]

Данная работа посвящена изучению правильного линейного вида для регулярных языков с полиномиальной функцией роста и улучшению соответствующих оценок на сложность перехода от линейного вида к правильному линейному виду.

Удалось понизить ранее известную из работы [1] оценку n^2 до оценки $\frac{n^2}{2}+n$. Так же получена верхняя оценка $\frac{n^2}{4}$ для языков вида $\beta_1^*\beta_2^*\dots\beta_s^*$, в которых слова β_i соизмеримы.

Ключевые слова: регулярный язык, функция роста, правильный линейный вид, сложность.

1. Введение

Известно, что регулярные языки с полиномиальной функцией роста задаются регулярными выражениями линейного вида [1]. Выражениями линейного вида называются конечные объединения выражений вида

$$P = \alpha_1 \cdot (\beta_1)^* \cdot \alpha_2 \dots \alpha_s \cdot (\beta_s)^* \cdot \alpha_{s+1},$$

где $s \in \mathbb{N}_0,\ \alpha_1, \dots, \alpha_{s+1} \in A^*,\ \beta_1, \dots, \beta_s \in A^* \setminus \{\lambda\}$.

Выражениями правильного линейного вида называются выражения этого же вида, для которых $\alpha_2, \dots, \alpha_s$ не пусты и первые буквы слов $\beta_i,\ \alpha_{i+1}$ различны при всех $1 \le i \le s$ (для пустого α_{s+1} это условие при $i = s$ опускается).

Необходимо оценить сложность перехода от представлений языков выражениями линейного вида к представлениям выражениями правильного линейного вида.

В статье мы сначала даем определения и основные свойства этих выражений, а также формулируем задачу об оценке выражений.

Ранее в работе [1] изучался этот вопрос и была получена квадратичная верхняя оценка. Также в работе [2] для случая несоизмеримых

[1] *Дергач Пётр Сергеевич* — к.ф.-м.н., м.н.с. каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: dergachpes@gmail.com.

Dergach Peter Sergeevich — Ph.D., junior researcher, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

[1] *Сальцова Диана Александровна* — выпускник Филиала МГУ имени М. В. Ломоносова в городе Ташкенте, e-mail: saltsovadiana@gmail.com.

Saltsova Diana Alexandrovna — Graduate of the M. V. Lomonosov Moscow State University Branch in Tashkent.

выражений линейного вида (частный случай) была получена линейная верхняя оценка $4n$.

Далее в статье мы улучшаем общую верхнюю оценку на сложность перехода от регулярного выражения линейного вида к регулярному выражению правильного линейного вида.

Также отдельно рассмотрен случай, когда язык имеет соизмеримый вид. Получена верхняя оценка $\frac{n^2}{4}$.

В заключении мы подводим итоги и обсуждаем возможные направления для будущих исследований в данной области.

2. Основные определения и результаты

Большая часть определений данного раздела взята из источников [3]-[5].

Определение 1. Дадим определение регулярного множества. Множество P, называется *регулярным* в алфавите A, если его можно получить из пустого множества и одноэлементных однобуквенных множеств $\{a\}$, $a \in A$ следующим образом

$\emptyset$ — регулярное множество в алфавите A;

$\{a\}$ — регулярное множество в алфавите A, a — произвольная буква алфавита A;

Если $P_1,\ P_2$ — регулярные множества в алфавите A, то и множества $P_1 \cup P_2,\ P_1 \cdot P_2,\ (P_1)^*$ будут регулярными множествами в алфавите A.

Определение 2. Пусть $P \subseteq A^*$ и $n \in \mathbb{N}$. Через $P_{\leq}(n)$ обозначаем множество всех слов из P длины не больше n. Через T_P обозначаем функцию $T_P : N \to N_0$, где

$$T_P(n) := |P_{\leq}(n)|$$

для всех $n \in \mathbb{N}$. Называем T_P *функцией роста* для P. Говорим, что P имеет *полиномиальную функцию роста*, если $T_P(n)$ ограничена сверху полиномом от n.

Определение 3. *Регулярным выражением* в алфавите A, называется слово из алфавита $A \cup \{^*,\ \cdot,\ \vee,\ (,\ ),\ \lambda\}$, которое определяется следующим образом:

λ — регулярное выражение в алфавите A;

Буквы алфавита A — регулярные выражения в алфавите A;

Если $P_1,\ P_2$ — регулярные выражения в алфавите A, то выражения $P_1 \cup P_2,\ P_1 \cdot P_2,\ (P_1)^*$ — будут регулярными выражениями в алфавите A.

Определение 4. *Длиной регулярного выражения* P в алфавите A будем называть количество букв из алфавита A, которые входят в это регулярное выражение с учетом кратности.

Определение 5. Слово $\alpha \in A^*$ называется *измельчением* слова $\beta \in A^*$, если при некотором $k \in \mathbb{N}$ выполняется равенство $\beta = \alpha^k$.

Определение 6. Слово $\alpha \in A^*$ называется *минимальным измельчением* непустого слова $\beta \in A^*$, если его длина минимальна среди всех измельчений слова β.

Определение 7. Два слова $\beta_1,\ \beta_2 \in A^*$ называются *соизмеримыми*, если их минимальные измельчения совпадают. В противном случае будем называть их *несоизмеримыми*.

Определение 8. Семейство слов $\beta_1, \beta_2, \ldots, \beta_n \in A^*$ называется *соизмеримым в совокупности*, если у них есть общее измельчение.

Определение 9. Регулярное выражение P в алфавите A имеет *линейный вид*, если оно представимо в виде конечного объединения выражений вида

$$P = \alpha_1 \cdot (\beta_1)^* \cdot \alpha_2 \ldots \alpha_s \cdot (\beta_s)^* \cdot \alpha_{s+1},$$

где $s \in \mathbb{N}_0,\ \alpha_1, \ldots, \alpha_{s+1} \in A^*,\ \beta_1, \ldots, \beta_s \in A^* \setminus \{\lambda\}$.

Если при этом $\alpha_2, \ldots, \alpha_s$ не пусты и первые буквы слов $\beta_i,\ \alpha_{i+1}$ при всех $1 \le i \le s$ различны (для пустого α_{s+1} это условие при $i = s$ опускается), то будем называть такое выражение *выражением правильного линейного вида.*

Определение 10. Будем говорить, что выражение линейного вида *несоизмеримо*, если все соседние пары β_i, β_{i+1} в нём несоизмеримы.

Определение 11. Пусть регулярное множество P задано конечным объединением регулярных выражений линейного вида. *Линейной сложностью такого представления* называем максимальную из длин этих выражений.

Определение 12. Пусть регулярное множество P задано конечным объединением регулярных выражений правильного линейного вида. *Правильной линейной сложностью такого представления* называем максимальную из длин этих выражений.

Определение 13. Пусть $P \subseteq \mathbb{N}$. Называем такое множество периодическим, если существуют $T_0,\ T \in \mathbb{N}$ такие, что начиная с $k \ge T_0$ из условия $k \in P$ следует условие $k + T \in P$. Число T_0 называем *предпериодом*, а число T — *периодом.*

Определение 14. Пусть $P \subseteq \mathbb{N}$ — периодическое множество. Через $T_{\text{предп}}(P)$ будем обозначать его минимальный предпериод.

Определение 15. Пусть $A = (\nu^{a_1})^*(\nu^{a_2})^* \dots (\nu^{a_s})^*$. Тогда через $X_\nu(A)$ обозначаем множество $X_\nu(A) = \{l \mid \nu^l \in A\}$.

Определение 16. Пусть $a, b \in \mathbb{N}$. Если a делится нацело на b, то пишем $b|a$. Через $Z_{a,b}$ обозначаем множество:

$$Z_{a,b} := \{n \in \mathbb{N} |\ n \geq a,\ b|n\}.$$

Теорема 1. *Пусть регулярный язык задаётся регулярным выражением следующего линейного вида*

$$A = (\nu^{a_1})^*(\nu^{a_2})^* \dots (\nu^{a_s})^*$$

со сложностью не выше n. Тогда его можно представить регулярным выражением правильного линейного вида с правильной линейной сложностью не выше $\frac{n^2}{4} + n$.

Теорема 2. *Пусть регулярный язык задаётся регулярным выражением следующего линейного вида*

$$A = \beta_1^* \beta_2^* \dots \beta_s^*$$

со сложностью не выше n. Тогда его можно представить регулярным выражением правильного линейного вида с правильной линейной сложностью не выше $\frac{n^2}{2} + n$.

3. Вспомогательные утверждения

Лемма 1. *Пусть $x_1, \dots, x_k$ — натуральные числа и $r := НОД(x_1, \dots, x_k)$, $H := \{a_1x_1 + \dots + a_kx_k \mid a_1, \dots, a_k \in \mathbb{N}_0\}$. Тогда существует $n_0 \in \mathbb{N}$ такое, что $Z_{n_0,r} \subseteq H$.*

Доказательство. Доказательство леммы приведено в главе 2 работы [1]. □

Лемма 2. *Пусть A — конечный алфавит и P — выражение линейного вида сложности не выше n. Тогда его можно представить выражением правильного линейного вида сложности не выше n^2.*

Доказательство. Данная лемма взята из главы 5 работы [1], там она приводится как лемма 3. □

Лемма 3. *Пусть A — конечный алфавит, $\alpha, \beta \in A^* \setminus \{\Lambda\}$ и $\alpha^k = \beta^m$ для некоторых $k, m \in \mathbb{N}$. Тогда существует $\nu \in A^* \setminus \{\Lambda\}$ такое, что*

$$|\nu| = НОД(|\alpha|, |\beta|), \alpha = \nu^{\frac{|\alpha|}{|\nu|}}, \beta = \nu^{\frac{|\beta|}{|\nu|}}.$$

Доказательство. Доказательство леммы приведено в главе 2 работы [1]. □

Лемма 4. *Пусть $a, b \in \mathbb{N}$, $a + b \leq n$. При таких ограничениях максимальное значение произведения ab не превосходит $\frac{n^2}{4}$.*

Доказательство. Рассмотрим два случая — когда сумма чётна и нечётна.

1) Пусть $a + b$ — чётно. Тогда для доказательства этой леммы воспользуемся неравенством между средним арифметическим и геометрическим:
$$\frac{a+b}{2} \geq \sqrt{ab},$$
где a и b — неотрицательные числа. Из этого неравенства следует, что
$$ab \leq \left(\frac{a+b}{2}\right)^2 = \frac{n^2}{4}.$$

2) Пусть $a + b$ — нечётно. Если a и b отличаются хотя бы на 2 и, без ограничения общности, $a \leq b$, то мы можем заменить a на $a+1$, а b — на $b-1$. От этого значение произведения может только вырасти. Таким образом, максимальное значение произведения достигается, когда $a = k, b = k + 1$. И окончательно получаем
$$k(k+1) \leq \frac{(2k+1)^2}{4} = \frac{n^2}{4},$$
так как
$$4k^2 + 4k \leq 4k^2 + 4k + 1.$$

□

Лемма 5. *Пусть $A = (0^{a_1})^*(0^{a_2})^* \dots (0^{a_s})^*$, $a_1 + a_2 + \dots + a_s \leq l$ и a_1, a_2 — взаимно просты. Тогда $T_{предп}(X_0(A)) \leq a_1 a_2$.*

Доказательство. Рассмотрим следующие числа:

$$a_1,\ 2a_1,\ 3a_1\ \dots, a_2 a_1.$$

Они все входят в $X_0(A)$. Рассмотрим остатки этих чисел по модулю a_2. Из взаимной простоты a_1 и a_2 следует, что все эти остатки различны. А

значит в этом списке встретятся все остатки. Самое большое число здесь — a_2a_1. Поэтому у любого другого числа $t \geq a_2a_1$ будет остаток, который мы уже получили ранее.

А раз у нас уже есть в списке число с таким остатком, то мы можем прибавить к нему a_2 несколько раз и получить число t. То есть, начиная с момента a_2a_1 период для X будет равен единице. А это значит, что $T_{\text{предп}}(X_0(A)) \leq a_1a_2$. □

Лемма 6. *Пусть* $A = (\nu^{a_1})^*(\nu^{a_2})^* \dots (\nu^{a_s})^*, s \geq 2$ *и выполнено неравенство* $(a_1+a_2+\dots+a_s)|\nu| \leq l$. *Тогда, для множества* $X_\nu(A) = \{l \mid \nu^l \in A\}$ *верно, что*

$$T_{предп}(X_\nu(A)) \leq \frac{l^2}{4|\nu|}.$$

Доказательство. Сначала докажем утверждение для случая, когда система чисел $a_1, a_2, \dots a_s$ взаимно проста в совокупности. Проведем доказательство методом обобщенной математической индукции по параметру s.

База индукции $(s = 2)$.

$$A = (\nu^{a_1})^*(\nu^{a_2})^*.$$

Рассмотрим вспомогательный случай $B = (0^{a_1})^*(0^{a_2})^*$.
Мы имеем $a_1 + a_2 \leq \frac{l}{|\nu|}$. Тогда по лемме 5 получаем

$$T_{\text{предп}}(X_0(B)) \leq a_1a_2.$$

И по лемме 4 это число не превосходит $\frac{l^2}{4|\nu|^2}$. Поэтому

$$T_{\text{предп}}(X_\nu(A)) \leq T_{\text{предп}}(X_0(B))|\nu| \leq \frac{l^2}{4|\nu|}.$$

Переход индукции. $(2, 3, \dots, s-1 \Rightarrow s)$
Докажем для s. Мы имеем

$$A = (\nu^{a_1})^*(\nu^{a_2})^* \dots (\nu^{a_s})^*.$$

Без ограничения общности упорядочим $a_1, a_2, \dots, a_s$ следующим образом

$$a_1 \leq a_2 \leq \dots \leq a_{s-1} \leq a_s.$$

Рассмотрим вспомогательное множество $\hat{A} = (0^{a_1})^*(0^{a_2})^* \dots (0^{a_s})^*$.
Тогда для множества

$$B = (0^{a_1})^*(0^{a_2})^* \dots (0^{a_{s-1}})^*$$

предпериод $T_{\text{п}}$ множества $X_0(B)$ не превосходит $\frac{(a_1+a_2+\cdots+a_{s-1})^2}{4}$ по предположению индукции. Обозначим через d_1 наибольший общий делитель системы чисел $a_1, a_2, \ldots, a_{s-1}$. По лемме 5 мы знаем, что начиная с момента $T_{\text{п}}$ множество $X_0(B)$ имеет период d_1. Рассмотрим вспомогательное множество $C = (0^{d_1})^*(0^{a_s})^*$. Мы знаем, что d_1 и a_s взаимно просты, поэтому по предположению базы предпериод множества $X_0(C)$ не превосходит $\frac{(d_1+a_s)^2}{4}$. Значит предпериод множества $X_0(\hat{A})$ не превосходит суммы предпериодов $X_0(B)$ и $X_0(C)$:

$$\frac{(a_1 + a_2 + \cdots + a_{s-1})^2}{4} + \frac{(d_1 + a_s)^2}{4}.$$

Докажем, что это выражение не превосходит $\frac{l^2}{4|\nu|}$.

Для доказательства этого неравенства достаточно доказать, что

$$\frac{(a_1 + a_2 + \cdots + a_{s-1})^2}{4} + \frac{(d_1 + a_s)^2}{4} \leq \frac{(a_1 + a_2 + \cdots + a_{s-1} + a_s)^2}{4},$$

что равносильно следующему неравенству

$$d_1^2 + 2d_1a_s \leq 2a_s(a_1 + \cdots + a_{s-1}).$$

Что, очевидно, верно, так как $d_1 \leq a_1 \leq a_2 \leq \cdots \leq a_s$ и $s - 1 \leq 2$. Возвращаясь к исходному множеству A, получаем

$$T_{\text{предп}}(X_\nu(A)) \leq T_{\text{предп}}(X_0(\hat{A}))|\nu| \leq \frac{l^2}{4|\nu|}.$$

Переход индукции доказан.

Теперь рассмотрим случай, когда НОД$(a_1, a_2, \ldots, a_s) = d > 1$.

Обозначим $\hat{\nu} = \nu^d$ и представим наше множество A в виде:

$$A = (\hat{\nu}^{\frac{a_1}{d}})^*(\hat{\nu}^{\frac{a_2}{d}})^* \ldots (\hat{\nu}^{\frac{a_s}{d}})^*.$$

Для такого представления множества мы уже выше доказали, что

$$T_{\text{предп}}(X_{\hat{\nu}}(A)) \leq \frac{l^2}{4|\hat{\nu}|} = \frac{l^2}{4d|\nu|}.$$

Поэтому

$$T_{\text{предп}}(X_\nu(A)) \leq T_{\text{предп}}(X_{\hat{\nu}}(A))d \leq \frac{l^2}{4d|\nu|}d = \frac{l^2}{4|\nu|}.$$

Лемма доказана. □

Лемма 7. *Пусть $A = \beta_1^*\beta_2^* \ldots \beta_s^*$. Если соседние пары β_i в этом выражении соизмеримы, то тогда все β_i соизмеримы в совокупности.*

Доказательство. Данное утверждение легко доказывается индукцией по s. При $s = 2$ утверждение очевидно. Докажем переход индукции. Пусть мы уже доказали утверждение для множества $\beta_1^*\beta_2^* \dots \beta_{s-1}^*$. Тогда по индукционному предположению

$$\beta_1^*\beta_2^* \dots \beta_s^* = (\nu^{a_1})^*(\nu^{a_2})^* \dots (\nu^{a_{s-1}})^*\beta_s^*.$$

Осталось заметить, что из соизмеримости β_{s-1}, β_s и леммы 3 следует соизмеримость ν, β_s. Значит у слов $\beta_1, \beta_2, \dots, \beta_s$ есть общее измельчение.

Лемма доказана. □

4. Доказательство основных утверждений

Теорема 1. *Пусть регулярный язык задаётся регулярным выражением следующего линейного вида*

$$A = (\nu^{a_1})^*(\nu^{a_2})^* \dots (\nu^{a_s})^*$$

со сложностью не выше n. Тогда его можно представить регулярным выражением правильного линейного вида с правильной линейной сложностью не выше $\frac{n^2}{4} + n$.

Доказательство. При $s = 1$ утверждение очевидно, поэтому далее будем считать, что $s \geq 2$.

Для доказательства этой теоремы необходимо разобраться в доказательстве леммы 3 главы 5 работы [1]. Там показано, что наше представление задаётся в виде периодического семейства степеней ν, которое может быть получено конечным объединением элементов из предпериода и периодической части с длиной периода НОД$(a_1, a_2, \dots, a_s)$.

Каждое множество из этого объединения уже имеет правильный линейный вид. Самое большое по длине из этих множеств задаётся в следующем виде:

$$\nu^{T_{\text{предп}}(X_\nu(A))}(\nu^{\text{НОД}(a_1,a_2,\dots,a_s)})^*. \qquad (*)$$

Оценим его длину. По лемме 6

$$T_{\text{предп}}(X_\nu(A)) \leq \frac{n^2}{4|\nu|}.$$

И НОД$(a_1, a_2, \dots, a_s) \leq a_1 \leq a_1 + \dots + a_s \leq \frac{n}{|\nu|}$.

Поэтому длина выражения (*) не превосходит

$$|\nu|(T_{\text{предп}}(X_\nu(A)) + \text{НОД}(a_1, a_2, \dots, a_s)) \leq |\nu| \left(\frac{n^2}{4|\nu|} + \frac{n}{|\nu|} \right) = \frac{n^2}{4} + n.$$

Теорема доказана. □

Теорема 2. *Пусть регулярный язык задаётся регулярным выражением следующего линейного вида*

$$A = \beta_1^* \beta_2^* \dots \beta_s^*$$

со сложностью не выше n. Тогда его можно представить регулярным выражением правильного линейного вида с правильной линейной сложностью не выше $\frac{n^2}{2} + n$.

Доказательство. Для доказательства данного утверждения нужно глубже погрузиться в доказательство леммы 3 главы 4 работы [1]. В ней рассматриваются процедура "расщепления", позволяющая преобразовывать соседнюю пару итераций $\beta_1^* \beta_2^*$ в выражение правильного линейного вида.

На идейном уровне эту процедуру можно описать так: из итерации β_2^* вытягивается несколько раз слово β_2 так, чтобы результат нельзя было целиком перебросить через итерацию β_1. И показывается, что длина вытянутого слова не превосходит $|\beta_1||\beta_2|$.

С помощью этого факта можно доказать утверждение нашей теоремы. Для этого разобьём выражение $\beta_1^* \beta_2^* \dots \beta_s^*$ в последовательность из блоков $B_1, B_2, \dots, B_k$, где слова в каждом из блоков B_i и B_{i+1} попарно соизмеримы, а значит по лемме 7 все блоки B_i представимы в виде

$$B_i = (\nu_i^{a_1(i)})^* (\nu_i^{a_2(i)})^* \dots (\nu_i^{a_s(i)})^*.$$

При этом ν_i и ν_{i+1} несоизмеримы в силу разбиения на блоки. Обозначим длину блока B_i через l_i. Из теоремы 1 знаем, что каждый блок B_i можно заменить на представление правильного линейного вида с длиной не больше чем $\frac{l_i^2}{4} + l_i$. При этом очень важно отметить, что в новом линейном виде все подъитерационные слова являются степенями слова ν_i.

Дальше в ход идёт процедура "расщепления" для пары несоизмеримых звёзд, позволяющая "развести" соседние блоки B_i. Для этого, как было показано в доказательстве теоремы 1 потребуется увеличить длину выражения на не более чем $l_1 l_2 + l_2 l_3 + \dots + l_{s-1} l_s$ символов.

Совмещая вместе идеи о "расщеплении" и приведении блоков в правильный линейный вид окончательно получаем оценку на длину нового линейного вида:

$$\frac{l_1^2}{4} + l_1 + \dots + \frac{l_s^2}{4} + l_s + (l_1 l_2 + l_2 l_3 + \dots + l_{s-1} l_s) =$$

$$= \frac{l_1^2 + l_2^2 + \dots + l_s^2 + 4 l_1 l_2 + 4 l_2 l_3 + \dots + 4 l_{s-1} l_s}{4} + (l_1 + l_2 + \dots + l_s) \leq$$

$$\leq \frac{(l_1 + l_2 + \cdots + l_s)^2}{4} + \frac{2l_1l_2 + 2l_2l_3 + \cdots + 2l_{s-1}l_s}{4} + (l_1 + l_2 + \cdots + l_s) \leq$$

$$\leq \frac{(l_1 + l_2 + \cdots + l_s)^2}{4} + \frac{(l_1 + l_2 + \cdots + l_s)^2}{4} + (l_1 + l_2 + \cdots + l_s) =$$

$$= \frac{(l_1 + l_2 + \cdots + l_s)^2}{2} + (l_1 + l_2 + \cdots + l_s).$$

Правильность полученного линейного вида следует из выше отмеченного факта о том, что подъитерационные слова в заменённых блоках по прежнему начинаются с ν_i и не портят условие правильности при использовании процедуры "расщепления".

Теорема доказана. □

5. Заключение

В данной статье было проведено исследование о представлении регулярных языков с полиномиальной функцией роста с использованием регулярных выражений линейного вида. Основной целью публикации было разработать новую оценку сложности перехода от линейного вида регулярных языков к правильному линейному виду.

В ходе исследования были рассмотрены существующие оценки сложности перехода и проведен анализ их эффективности. Было обнаружено, что существующие оценки не всегда достаточно точно оценивают сложность перехода. Это стало основной мотивацией для разработки новой оценки.

На основе проведенного исследования была сформулирована и доказана основная теорема, утверждающая, что регулярный язык, задаваемый регулярным выражением линейного вида с линейной сложностью не превышающей n, может быть задан регулярным выражением правильного линейного вида с линейной сложностью не превышающей $\frac{n^2}{2} + n$. Это означает, что предложенная новая оценка более точно оценивает сложность перехода и может быть применена для оптимизации представления регулярных языков.

Список литературы

[1] Дергач П.С., "Алфавитное кодирование регулярных языков с полиномиальной функцией роста", *Кандидатская диссертация*, 2016, 1–213.

[2] Абдурахмонов А.Н., "О правильной линейной форме", *Выпускная квалификационная работа*, 2019, 1–23.

[3] В. Б. Кудрявцев, С. В. Алешин, А. С. Подколзин., “Введение в теорию автоматов”, *М.: Наука*, 1985.

[4] Яблонский С.В., “Введение в дискретную математику”, *М.: Наука*, 1986.

[5] Виноградов И.М., “Введение в теорию чисел”, *М.: Наука*, 1972.

On the complexity of converting to a correct linear form
Dergach P.S., Saltsova D.A.

This paper is devoted to the study of the regular linear form for regular languages with polynomial growth function and improving the corresponding estimates on the complexity of the transition from the linear form to the regular linear form.

We managed to lower the previously known estimate n^2 from [1] to an estimate $\frac{n^2}{2} + n$. Also obtained an upper estimate $\frac{n^2}{4}$ for languages of the form $\beta_1^*\beta_2^* \dots \beta_s^*$, in which the words β_i are commensurable.

Keywords: regular language, growth function, regular linear form, complexity.

References

[1] Dergach P.S., “Alphabetic encoding of regular languages with polynomial growth function”, *PhD thesis*, 2016, 1–213.

[2] Abdurakhmonov A.N., “About the correct linear form”, *bachelor thesis*, 2019, 1–23.

[3] Kudryavtsev V.B., Aleshin S.V, Podkolzin A.S., “Introduction to automata theory”, 1985.

[4] Yablonskiy S.V., “Introduction to discrete math”, 1986.

[5] Vinogradov I. M., “Basics of number theory”, 1972.

Оценка степеней разделяющих многочленов для монотонных и самодвойственных функций

М. В. Носов[1]

В работе получена верхняя оценка степени многочлена с действительными коэффициентами разделяющего нули и единицы монотонной булевской функциии в нечётном случае размерности пространства. Вместе с ранее известными оценками для чётного случая и нижней оценки для нечётного получается окончательный результат. Аналогичные результаты получены для класса самодвойственных функций.

Ключевые слова: монотонная булевская функция, самодвойственная булевская функция, разделяющий многочлен.

Пусть B^n - n-мерный единичный куб, $R[x]$ - множество многочленов от n переменных с действительными коэффициентами, $x = (x_1, \ldots, x_n)$, P_2 - множество булевых функций, M_n - множество монотонных булевых функций от n переменных. Скажем, что многочлен $f(x), f(x) \in R[x]$ разделяет нули и единицы булевой функции $F(x)$ если

$$F(\alpha) = 1 \Leftrightarrow f(\alpha) \geq 0, \quad F(\alpha) = 0 \Leftrightarrow f(\alpha) < 0, \quad \alpha = (\alpha_1, \ldots, \alpha_n) \in B^n.$$

Такой многочлен будем называть разделяющим многочленом функции $F(x)$.

1.В работе [2] доказано, что для любой монотонной булевской функции от n переменных существует разделяющий многочлен степени не более $[\frac{n+1}{2}]$ и существует булевская функция, для которой степень разделяющего полинома не менее $[\frac{n}{2}]$. При чётном n верхняя и нижняя оценки совпадают, покажем, что будет такая же ситуация и при нечётном n.

Пусть $n = 2k + 1, F(x) \in M_n$, имеет место следующее представление [1]

$$F(x) = \bigvee_{(i_1,\ldots,i_m)\in U} x_{i_1} \ldots x_{i_m},$$

где $K = \{\sigma = (0, \ldots, \sigma_{i_1}, 0, \ldots, \sigma_{i_m}, 0, \ldots, 0) | \sigma_{i_j} = 1, (i_1, \ldots, i_m) \in U\}$ - множество нижних единиц функции F. Будем представление $F(x)$ записывать в виде

[1] *Носов Михаил Васильевич* — с.н.с. каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: mvnosov@rambler.ru.

Nosov Michail Vasilevich-senior researcher, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

$$F(x) = \bigvee_{\sigma \in K} x^{\sigma}.$$

Определим множества K_1 и K_2:

$$K_1 = K \bigcap \{\sigma | \, |\sigma| \leq k\}, \quad K_2 = K \bigcap \{\sigma | \, |\sigma| \geq k+1\}$$

тогда

$$F(x) = F_1(x) \bigvee F_2(x),$$

$$F_1(x) = \bigvee_{\sigma \in K_1} x^{\sigma}, \quad F_2(x) = \bigvee_{\sigma \in K_2} x^{\sigma}.$$

Определим многочлен из $R[x]$:

$$f_1(x) = \sum_{\sigma \in K_1} x^{\sigma}, \quad deg f_1 \leq k.$$

Тогда

$$F_1(\alpha) = 1 \Leftrightarrow f_1(\alpha) \geq 0, \quad F_1(\alpha) = 0 \Leftrightarrow f_1(\alpha) = 0, \quad \alpha \in B^n.$$

Если $K_2 = \emptyset$, то$(f_1(x) - 0.5)$ - разделяющий многочлен F и степень его не более k.

Рассмотрим случай $K_2 \neq \emptyset$. Определим функцию $G(x) = \overline{F_2(\overline{x})}, F_2(x) = \overline{G(\overline{x})}$. Очевидно, что $G(x)$ монотонная булевская функция и $\{\overline{\sigma} | \quad \sigma \in K_2\}$ - верхние нули функции G. Для любого $\sigma \in K_2$ имеем $|\overline{\sigma}| \leq (2k+1) - (k+1) = k$, значит, если Λ - нижние единицы $G(x)$, то для любого $\lambda \in \Lambda$, $|\lambda| \leq k+1$ и слой E^n_{k+1} лежит во множестве единиц G.

Определим множества K_3 и K_4:

$$K_3 = \Lambda \bigcap \{\lambda | \, |\lambda| \leq k\}, \quad K_4 = K \bigcap \{\lambda | \, |\lambda| = k+1\}$$

и функции

$$F_3(x) = \bigvee_{\lambda \in K_3} x^{\lambda}, \quad F_4(x) = \bigvee_{\lambda \in K_4} x^{\lambda}.$$

тогда

$$G(x) = F_3(x) \bigvee F_4(x),$$

Определим многочлен

$$f_2(x) = \sum_{\lambda \in K_3} x^{\lambda} + \delta(\sum_{i=1}^{n} x_i - k - 1),$$

где $0 < \delta < \frac{1}{2^{2k+3}}$. Тогда, если $\alpha \in B^n$ и $G(\alpha) = 0$, то $-(k+1)\delta \leq f_2(x) \leq -\delta$; если $G(\alpha) = 1$, то возможны три случая: $1)F_3(\alpha) = 1$, $F_4(\alpha) = 1$; $2)F_3(\alpha) = 1$, $F_4(\alpha) = 0$; $3)F_3(\alpha) = 0$, $F_4(\alpha) = 1$. В первом случае $1 \leq f_2(x) \leq 2^{2k+2}$, во втором случае $0 < 1 - \delta(k+1) \leq f_2(\alpha) \leq 2^{2k+1}$; в третьем случае $f_2(\alpha) = 0$. Во всех трёх случаях $f_2(\alpha) \geq 0$.

Возьмём многочлен

$$f(x) = f_1(x) - \delta f_2(1 - x_1, \ldots, 1 - x_n).$$

Если $\alpha \in B^n$ и $F(\alpha) = 0$, значит $f_1(\alpha) = 0, \quad F_2(\alpha) = 0, \quad , G(\overline{\alpha}) = 1$, значит $f_2(\overline{\alpha}) \geq 0$ и $f(\alpha) \leq 0$. Пусть $F(\alpha) = 1$, тогда могут быть три случая: $1)F_1(\alpha) = 1$, $F_2(\alpha) = 1$; $2)F_1(\alpha) = 1$, $F_2(\alpha) = 0$; $3)F_1(\alpha) = 0$, $F_2(\alpha) = 1$. В первом и втором случаях $f(\alpha) \geq 1 - \delta 2^{2k+2} \geq \frac{1}{2}$. В третьем случае $f_1(\alpha) = 0, \quad G(\overline{\alpha}) = 0, \quad -(k+1)\delta \leq f_2(\overline{\alpha}) \leq -\delta$, тогда $f(\alpha) \geq \delta^2$. Многочлен $f(x) - \frac{1}{2}\delta^2$ будет разделять нули и единицы монотонной функции $F(x)$ и $degf(x) \leq k$.

2.Для класса самодвойственных функций рассмотрим отдельно нечётный и чётный случаи. В первом случае линейная функция от всех аргументов является самодвойственной, её нули и единицы разделяются только полиномом степени n [2]. Во втором случае обе линейные функции от всех переменных не являются самодвойственными, следовательно, степень разделяющего полинома не более $n - 1$, а для линейной самодвойственной функции с одной несущественой переменной, т.е. от нечётного числа существенных переменных $n - 1$, требуется полином степени $n - 1$.

Список литературы

[1] Яблонский С.В., Лупанов О.Б.(ред.) Дискретная математика и математические вопросы кибернетики. Издательство Наука, Москва, 1974, 242.

[2] Алешин С.В. Распознавание динамических образов. Издательство Московского университета, Москва, 1996, 97.

Estimates of the degrees of separating polynomials for monotone and self-dual functions

Nosov M.V.

In this paper, we obtain an upper bound on the degree of a polynomial with real coefficients separating zeros and ones of a monotone Boolean function in the odd case of the dimension of space.Together with the previously known estimates for the even case and the lower estimate for the odd one, the final result is obtained. Similar results are obtained for the class of self-dual functions.

*Keywords:*monotone Boolean function, self-dual Boolean function, separating polynomial.

References

[1] Yablonsky S.V.,Lupanov O.B.(ed.), *Discrete mathematics and mathematical questions of cybernetics*, Nauka, Moscow, 1974 (In Russian), 242 c.

[2] Aleshin S.V., *Dynamic image recognition*, Moscow University Press, Moscow, 1996 (In Russian), 97 c.

Часть 3.
Математические модели

О верхних оценках сложности синтеза автономных автоматных плоских схем

А. С. Воротников[1]

В работе получена верхняя оценка переключательной мощности реализации периодической последовательности плоской автоматной схемой без входов. Приводится схема, реализующая произвольную наперёд заданную последовательность длины 2^n для натуральных n с переключательной мощностью не более $\frac{2^{n/2}}{n}$.

Ключевые слова: схемы из функциональных элементов, конечные автоматы, модель СБИС, плоские схемы, мощность схем, функция Шеннона, верхние оценки.

1. Введение

Впервые понятие схемы из клеточных элементов, далее так же называемой плоскими схемами, было введено в работе Кравцова С.С. [1], показавшего, что для реализации произвольной булевой функции такой схемой требуется $O(2^n)$ элементов. В последствии был получен порядок функции Шеннона площади для частичных булевых функций $O(|D|)$, где D — область определения (Д. А. Жуков [2]). Связь между площадью плоских схем и объёмом трёхмерных, реализующих булевы операторы анализировалась в работе (Н.А. Шкаликова [3]). В работах [4, 5, 6] Г. В. Калачев изучал порядок потенциала и переключательной мощности плоских схем, реализующих булевы функции и операторы. Было показано, что порядок потенциала и переключательной мощности плоской схемы, реализующей булеву функцию от n переменных, составляет $2^{n/2}$; показана связь этих мер мощности. Так же был получен порядок переключательной мощности реализацией плоскими схемами частичных булевых операторов. О. М. Касим-Заде исследовал активность схем из функциональных элементов. В качестве меры активности схемы в работе [7] он ввёл понятие P-активности: на каждом входном наборе рассматривается количество элементов, на вход которых подаётся единица или сам элемент выдаёт единицу, после этого берётся максимум по всем наборам. Касим-Заде показал, что в некотором базисе для любой булевой функции от n переменных можно построить схему, P-активность которой имеет порядок $O(n^2)$.

[1] *Воротников Алексей Сергеевич* — аспирант каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: vorotnikov.lexa@yandex.ru.

Vorotnikov Alexey Sergeevich — graduate student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

В работе [8] Ю. С. Шуткин исследовал мощность контактных схем и получил линейный порядок функции Шеннона.

Рассматриваются так же задачи сложности синтеза для некоторых классов функций и операторов. В частности были найдены порядки роста функции Шеннона максимального и среднего потенциала для монотонных функций [9]. В работе [10] получены асимптотические оценки высокой степени точности для сложности реализации контактными схемами дешифратора.

В данной работе рассматривается расширенное понятие: плоские автоматные схемы. Это схемы, чей базис клеточных элементов составляют, помимо привычных булевых функций с не более чем четырьмя входами и выходами, задержки - автоматы с одним состоянием, подающие на выход в следующий такт то, что пришло на вход в предыдущий. Корректные схемы теперь в каждом ориентированном цикле должны содержать не менее одной задержки. Функционирует данная конструкция как структурный автомат. В дальнейшем рассматриваем только автоматные плоские схемы без входов, будем их так же называть автоматными схемами. В качестве меры сложности рассматривается переключательная мощность: суммарное число изменений на выходах элементов, нормированное на длину периода последовательности, поступающей на выход. Показано, что произвольную последовательность из нулей и единиц с периодом 2^n можно реализовать автоматной схемой, обладающей переключательной мощностью не более $\frac{2^{n/2}}{n}$. В работе приводится такая схема. При построении схемы активно применялись идеи, возникавшие ранее в работах Г. В. Калачёва [4, 5].

В дальнейшем планируется получить нижнюю оценку для переключательной мощности плоских автоматных схем, реализующих периодические последовательности, так же порядок переключательной мощности для реализаций плоскими автоматными схемами произвольных автоматов.

Автор выражает благодарность д.ф.-м.н. профессору Э. Э. Гасанову за научное руководство и помощь в работе, а также к.ф.-м.н., м.н.с. Г. В. Калачёву и А. А. Ефимову за ценные замечания и предложения по тексту статьи.

2. Основные понятия и формулировка результатов

2.1. Определение плоской автоматной схемы

Вводимое определение несколько расширяет определение плоской схемы, введённое в работе [4].

Клеточным элементом будем называть автомат с не более чем двумя состоянием, у которого в сумме не более четырёх входов и выходов, причём каждому его входу и каждому выходу сопоставлена некоторая метка из множества $\{l, r, t, b\}$, причём метки не повторяются. Метки будем также называть сторонами элемента:

- l — левая сторона;
- r — правая сторона;
- t — верхняя сторона;
- b — нижняя сторона.

Клеточный элемент будем изображать в виде единичного квадрата на плоскости. При этом входам и выходам элемента сопоставляются стороны квадрата в соответствии с присвоенными им метками. Метки, присвоенные входам (выходам) автомата будем называть входами (выходами) элемента. Метки, не присвоенные ни входам, ни выходам, будем называть изоляторами. Множество входов (выходов) элемента e будем обозначать $in(e)$ ($out(e)$). Входы и выходы элемента будем называть его контактами.

Описывать элемент с одним состоянием будем уравнениями, которые задают его оператор, заменяя все переменные в них на сопоставленные им метки (l, r, t или b). Тогда в левой части каждого уравнения будет стоять выходная метка, а в правую будут входить только входные метки. На рисунке 1 приведены примеры клеточных элементов. Для удобства

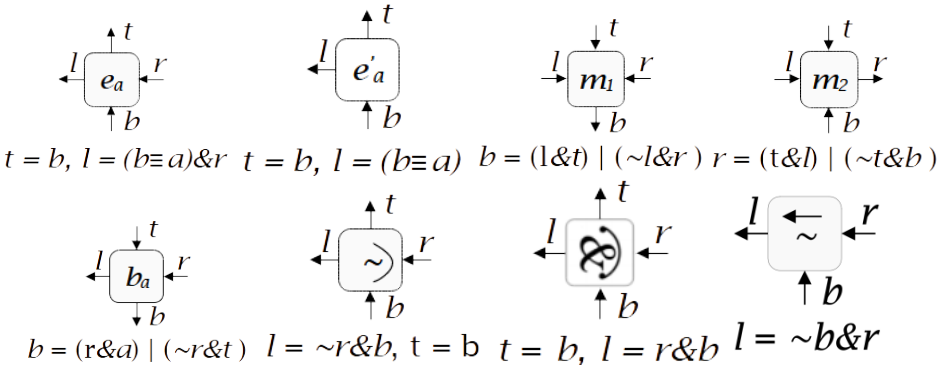


Рис. 1. Примеры логических элементов.

введём пустой клеточный элемент — изолирующий (будем обозначать λ). Всюду далее значок := будет обозначать «по определению равно». Через $\mathcal{E}$ обозначим множество всех клеточных элементов, $N_{\mathcal{E}} := |\mathcal{E}|$.

Далее везде используется базис $\mathcal{B}$, состоящий из всех элементов с одним состоянием и множества элементов с с двумя состояниями состояниями $\{(E, E, E, \phi, \psi_1), (E, E, E^2, \phi, \psi_2), (E, E, E^3, \phi, \psi_3)\}$, где $E = \{0, 1\}$,

$$\begin{cases} \varphi(1) = 0, \\ \varphi(t+1) = a(t). \end{cases}$$

$$\psi_1(t) = q(t), \quad \psi_2(t) = (q(t), q(t)), \quad \psi_3(t) = (q(t), q(t), q(t)),$$

где $a(t)$ — входной сигнал в момент времени t, $q(t)$ — состояние автомата в момент времени t. Будем называть такие элементы *задержками*. Иными словами, автомат просто играет роль задержки, «засасывая»сигнал по одному проводу и «отдавая»не более чем по трём проводам на следующем такте, рисунке 2.

Если на всех выходах элемента с одним состоянием реализуются тождественные функции, то будем называть элемент *коммутационным*, остальные элементы с одним состоянием будем называть *логическими*. *Сетью из клеточных элементов* на множестве $M \subset \mathbb{Z}^2$ над множеством

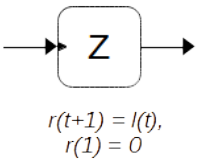


Рис. 2. Задержка.

$\mathcal{E}' \subseteq \mathcal{E}$ будем называть отображение $K : M \to \mathcal{E}'$, при этом $\mathcal{E}'$ будем называть *базисом* сети. Элемент $K(x, y)$ будем называть элементом схемы K с координатами (x, y). Элемент с приписанными ему координатами будем называть *элементом схемы*. *Левой*, *правой*, *верхней* и *нижней* сторонами элемента e с координатами (x, y) будем называть точки с координатами $(x - \frac{1}{2}, y), (x + \frac{1}{2}, y), (x, y - \frac{1}{2})$ и $(x, y + \frac{1}{2})$ соответственно (ось y направлена вниз). Будем говорить, что сеть K из клеточных элементов корректна, если для любых двух таких элементов x и y схемы K, что сторона a элемента x совпадает со стороной b элемента y, выполнено одно из условий:

- один из элементов x, y — изолирующий,
- стороны a и b являются изоляторами,
- либо одна из них является входом, другая — выходом, например, a — выход, а b — вход; в таком случае будем говорить, что выход a подключен к входу b.

Множество M будем называть *носителем* сети K. Введём понятие *графа корректной сети из клеточных элементов* K (будем обозначать его G_K): это ориентированный граф, вершинами которого являются входы и выходы элементов схемы. Если выход одного элемента подключен ко входу другого, то им будет соответствовать одна и та же вершина графа (будем говорить, что эта вершина является выходом первого элемента и входом второго). Из вершины a в вершину b ведёт ребро в том и только том случае, когда существует такой элемент e, что a является его входом, b — выходом, причём функция, реализуемая на выходе b, существенно зависит от входа a. *Плоской автоматной схемой* на множестве $M \subset \mathbb{Z}^2$ над базисом $\mathcal{E}' \subseteq \mathcal{E}$ будем называть корректную сеть из клеточных элементов, в графе которой все ориентированные циклы содержат хотя бы одну задержку. Множество M будем называть *носителем* схемы K.

Если вход (выход) элемента не подключен к выходу (входу) другого элемента, будем его называть *входом (выходом)* схемы. *Контактами* схемы будем называть её входы и выходы. Множество входов (выходов) схемы K будем обозначать $In(K)(Out(K))$. *Узлами* схемы K будем называть вершины графа G_K. Если M — носитель схемы K, то величину $|M|$, равную количеству элементов множества M , будем называть *площадью* схемы K и обозначать $|K|$.

Длиной схемы K будем называть длину наименьшего прямоугольника, содержащего все непустые элементы схемы K, обозначается $l(K)$. *Шириной схемы* K будем называть ширину наименьшего прямоугольника, содержащего все непустые элементы схемы K, обозначается $w(K)$.

Расстоянием между узлами схемы будем называть расстояние между соответствующими вершинами в G_K. Расстояние от узла a до узла b на схеме K будем обозначать $\rho_K(a, b)$. *Подсхемой* схемы K с носителем $M_0 \subseteq M$ будем называть схему $K|_{M_0}$, получающуюся из K выбрасыванием клеточных элементов, соответствующих множеству $M \backslash M_0$. Если схема K фиксирована, то иногда будем говорить просто подсхема M_0. Каждой плоской схеме K можно сопоставить структурный автомат $Circ(K)$ следующим образом:

1) каждой функции $f_{s,i}$, которую реализует i-й выход элемента s клеточной схемы, сопоставим функциональный элемент $e_{s,i}$, реализующий $f_{s,i}$; если i-й и j-й выходы являются выходами одной и той же функции, то им будет соответствовать один и тот же функциональный элемент;

2) если i-й выход s_1 подключен к j-му входу s_2, то соединим выход элемента $e_{s_1,i}$ с j-ми входами элементов $e_{s_2,k}$ для всех k, для которых $f_{s_2,k}$ зависит от j-го аргумента;

3) удалим из схемы все тождественные функции, подсоединив их вход ко всем их выходам;

4) аналогично поступаем с задержками.

Сопоставление корректно, так как правила сопоставления вкладываются в правила индуктивного построения структурных автоматов [11]. Правило обратной связи, требующее зависимость со сдвигом от замыкаемой переменной, так же выполнено в силу наличия задержки в каждом ориентированном цикле. Поскольку только ориентированные циклы могут порождать обратную связь, всё верно.

Будем говорить, что схема K реализует автомат A_K, если схема из автоматных элементов $Circ(K)$ реализует A_K.

Поскольку класс определённых здесь автоматных схем включает в себя класс плоских схем, то для синтеза автоматных схем можно пользоваться верхними оценками плоских схем. В частности, для блоков, не содержащих задержек, мы будем применять утверждения, известные для плоских схем. С другой стороны, каждую автоматную схему с n входами, m выходами и z задержками можно интерпретировать как плоскую схему, реализующую булев оператор с $n + z$ входами и $m + z$ выходами, где ровно z входов идут из пустых клеточных элементов, причём в эти же клеточные элементы приходит один выход. Иными словами, ровно z входов имеют пару в виде выхода, расположенного на той же клетке.

Значение всех выражений ниже, относящихся к параметрам блоков, полагается равным их округлению до целого вверх. Если при таком округлении возникают не используемые дальше провода полагается, что на них подана константа 0.

Везде ниже слово асимптотически понимается в том смысле, что $k \asymp n$ и $n \to \infty$.

Далее рассматриваем только плоские автоматные схемы без входов с единственным выходом.

2.2. Меры мощности схем

Рассмотрим плоскую автоматную схему K без входов, реализующую периодическую последовательность длины $l \in \mathbb{N}$. Последовательность, реализуемую схемой K обозначим α_K. Для каждой такой схемы K зафиксируем некоторую нумерацию её узлов. На i-м узле реализуется некоторая автоматная функция g_i. Везде далее будем считать, что схема K имеет h узлов и g_i — автоматная функция, реализуемая в i-м узле схемы K. Состоянием схемы K на такте t назовём вектор

$$s_K(t) := (g_1(t), ..., g_h(t)).$$

Величину $c_K(t) := |s_K(t) \oplus s_K(t+1)|$ назовём *затратой энергии на переключение схемы* с такта t на $t+1$. Длина вектора понимается как сумма целых чисел его компонент:

$$a = (a_1, \cdots, a_n) \in \{0,1\}^n, |a| = \sum_{i=1}^{n} a_i.$$

Схема K *функционирует циклически* с периодом l, если после последовательность на её выходе имеет период l.

Переключательной мощностью схемы K, функционирующей циклически с периодом l, назовём $W(K) = \dfrac{1}{l}\sum_{t=0}^{l} c_K(t)$.

Для подсхемы K' схемы K определим переключательную мощность как $W_K(K') = \dfrac{1}{l}\sum_{t=0}^{l} c'_K(t)$.

Переключательной мощностью последовательности α назовём величину $W(\alpha) = \min\limits_{\alpha_K=\alpha} W(K)$.

Введём функцию Шеннона для переключательной мощности последовательностей из класса $\mathcal{F}$:

$$W(\mathcal{F}, l) = \max_{\alpha \in \{0,1\}^l \cap \mathcal{F}} W(\alpha),$$

Если $\mathcal{F} = \{0,1\}^l$, то $W(\mathcal{F}, l) = W(l)$.

Теорема 1.

$$W(2^n) \preceq 12\frac{2^{n/2}}{n},\ \textit{при}\, n \to \infty$$

Причём достигнуть такой переключательной мощности можно используя не более $\frac{3}{2}n$ задержек асимптотически.

2.3. Вспомогательные утверждения

Для доказательства теоремы 1 нам потребуется следующая лемма.

Лемма 1.

$$W(2^n) \lesssim \begin{cases} 50\frac{2^{n/2}}{n}, & \textit{если } k = \frac{n}{4} + \frac{1}{2}\log_2 n, \\ 12\frac{2^{n/2}}{k}, & \textit{если } k \in (\frac{n}{4} + \frac{1}{2}\log_2 n, \frac{n}{2} - \log_2 n),\ \textit{при}\, n \to \infty \\ 36\frac{2^{n/2}}{n}, & \textit{если } k = \frac{n}{2} - \log_2 n, \end{cases}$$

используя не более $n + k + \log_2 k$ задержек, где параметр $k \in [\frac{n}{4} + \frac{1}{2}\log_2 n, \frac{n}{2} - \log_2 n] \cap \mathbb{N}$.

Докажем лемму следующим образом: сначала опишем блоки, используемые в дальнейшем для построения схемы H с требуемыми характеристиками, опишем устройство схемы H, затем оценим переключательную мощность блоков, составляющих схему H и, наконец, оценим переключательную мощность всей схемы H.

Везде ниже полагается $W_H(S) := W(S)$.

Фразу «за цикл (период) длины l »при рассмотрении некоторого блока надо воспринимать следующим образом. Или данный блок рассматривается как составной элемент некоторого другого блока, обладающего периодом l или данный блок сам обладает периодом l.

Лемма 2. *Если за цикл длины l вход схемы P меняется не чаще чем m раз, то $W(P) \leqslant \frac{1}{l}\left((2l(P)w(P) + l(P) + w(P))m\right)$. Вход понимается в расширенном смысле, как объединение собственно входов схемы и выходов задержек, содержащихся в схеме.*

Доказательство. Каждое изменение на входах схемы может привести к изменению не более чем всех узлов схемы. Осталось посчитать узлы, перемножить их с m и разделить на l. Пусть $l(P) = l, w(P) = w$. Рассмотрим прямоугольную схему S с такими размерами и всеми узлами. Число узлов схемы S не меньше числа узлов схемы P.

Рассмотрим прямоугольную схему S_l, $l(S_l) = l, w(S_l) = 1$. За исключением узлов на границе, в ней $l-1$ узел. Таким образом в строке схемы S $l-1$, а строк w. Если соединить две строки длины l, находящиеся друг над другом, то появится ещё l узлов. Всего для схемы S надо провести $w-1$ соединение. По периметру стоит не более $2(l+w)$ узлов. Итого число узлов в схеме S не больше

$$(l-1)w + l(w-1) + 2(l+w) \leqslant 2lw + l + w,$$

что завершает доказательство. □

Если схема P с периодом l состоит из схем $P_1, \cdots, P_m$, то

$$W(P) = \sum_{i=1}^{m} W(P_i).$$

Иными словами переключательная мощность аддитивна.

2.4. Описание основных блоков

Для реализации периодической последовательности нам потребуется несколько разных блоков. Опишем их характеристики.

1) Декодеры D_1, D_2 и D_3. В литературе эти блоки часто называются дешифраторами.

$$\begin{array}{lll} l(D_1) = 3(k - \log_2 k), & w(D_1) = \frac{2^k}{k}, & W(D_1) \lesssim 6\frac{2^k}{k}. \\ l(D_2) = 3k, & w(D_2) = 2^k, & W(D_2) \lesssim 6k. \\ l(D_3) = k, & w(D_3) = 2 \cdot 2^k, & W(D_3) \lesssim 4 \cdot 2^{3k-n}. \end{array}$$

2) Генератор последовательностей длины k K_α.

$$l(K_\alpha) = 2^k(1 + 3/k), \quad w(K_\alpha) = 2^{k+1}, \quad W(K_\alpha) \lesssim 4 \cdot 2^{3k-n}.$$

3) Блок дизъюнкций U.

$$l(U) = 2^k(1 + 3/k), \quad w(U) = k, \quad W(U) \lesssim 2 \cdot 2^{3k-n}.$$

4) Блок B, отвечающий за генерацию подпоследовательности длины 2^{2k}.

$$l(B) \lesssim 2^k(1 + 3/k), \quad w(B) \lesssim 2^{k+1}, \quad W(B) \lesssim 10 \cdot 2^{3k-n}.$$

5) Счётчики C_1, C_2, C_3, C_4.

$$\begin{array}{lll} l(C_1) = 6, & w(C_1) = 3(\log_2 k + 2), & W(C_1) \preceq \log_2 k. \\ l(C_2) \lesssim 3(\log_2 k + 2), & w(C_2) = 6, & W(C_2) \preceq \frac{k - \log_2 k}{k}. \\ l(C_3) = 3(k + 2), & w(C_3) = 6, & W(C_3) \preceq \frac{k}{2^k}. \\ l(C_4) = 6, & w(C_4) = 3(n - 2k + 2), & W(C_3) \preceq \frac{n-2k}{2^{2k}}. \end{array}$$

6) Блок проверки на равенство одному из записанных в его структуру чисел Q_α.

$$l(Q_\alpha) = 2^{n/2-k}, \quad w(Q_\alpha) = n - 2k + 1, \quad W(Q_\alpha) \lesssim 2(n - 2k)2^{n/2-3k}.$$

7) Энкодер E.

$$l(E) = k, \quad w(E) = 2^k, \quad W(E) \lesssim 2 \cdot 2^k.$$

8) Очередь f, способная полностью перезаписываться по сигналу.

$$l(f) = 2k, \quad w(f) = 2, \quad W(f) \lesssim 8k.$$

2.5. Реализация используемых блоков

Назовём прямоугольную схему *повторяющей*, если напротив каждого её входа x на противоположной стороне симметрично (рассматривается осевая симметрия) расположен выход y, на котором реализуется тождественная функция $y(x) = x$.

Будем описывать элементы схемы следующим образом: чтобы описать элемент e с координатами (x, y), будем писать

$$(x, y) : s_1 = f_1(s_{m+1}, ..., s_q), ..., s_m = f_m(s_{m+1}, ..., s_q),$$

где $s_j \in \{l, t, r, b\}$, причем $out(e) = \{s_1, ..., s_m\}$, $in(e) = \{s_{m+1}, ..., s_q\}$.

Введём индуктивно *множество проводов, выходящих из блока* X $\omega(X)$:

1) База индукции. Для каждого узла x из $Out(X)$ рассмотрим все узлы клеточного элемента, к которому он подключён вне X, и добавим к множеству проводов те узлы, на которых реализуется тождественная функция равная x.

2) Шаг индукции. Для каждого узла x' из множества, построенного на предыдущем шаге, рассмотрим клеточный элемент, к которому он подключён. Если клеточный элемент принадлежит X, то перейдём к рассмотрению следующего узла. Добавим к множеству те узлы этого клеточного элемента на которых реализуется тождественная функция равная x' и которых ещё нет в множестве.

Если в базе рассмотреть один единственный узел j, лежащий в схеме Y, и рассматривать только клеточные элементы из Y, то получится определение множества проводов, соединённых с входом j некоторого блока; так же будем обозначать его $\omega(j)$. Будем считать, что параллельно с построением множества $\omega(X)$ строится ориентированный граф G. На первом шаге граф состоит из вершин $Out(X)$. На каждом следующем шаге, если для узла u, находящегося среди вершин графа G, нашёлся клеточный элемент, удовлетворяющий всем требованиям определения (т.е. к нему подключён узел u, он лежит вне X), и на его узле v реализуется тождественная функция u, то добавим к G вершину v ребро (u, v). Если рассмотреть некоторый блок Y и в множестве $\omega(X)$ оставить только элементы, лежащие на ориентированных путях графа G с началами в $Out(X)$ и концами в $In(Y)$, то получится множество $\omega(X, Y)$ — *множество проводов, соединяющих блок* X *с блоком* Y.

Введём операцию транспонирования схемы из [4]. Транспонированный клеточный элемент e — элемент $Transp(e)$, получающийся из e, если поменять местами метки $l \leftrightarrow t$ и $r \leftrightarrow b$. Транспонированная схема K —

схема $K^T = Transp \circ K \circ T$, где $T : \mathbb{Z}^2 \to \mathbb{Z}^2, T(x, y) = (y, x)$. По сути, это схема, получающаяся отражением схемы K относительно прямой $y = x$.

Введём операцию отражения схемы по горизонтали. В дальнейшем для краткости операция будет называться просто отражением, так как отражение по вертикали не используется. Отражённый клеточный элемент e — элемент $Refl(e)$, получающийся из e, если поменять местами метки $l \leftrightarrow t$. Отражённая схема K — схема $K^T = Refl \circ K \circ T$, где $T : \mathbb{Z}^2 \to \mathbb{Z}^2, T(x, y) = (L + 1 - x, y)$, где L — ширина схемы K. По сути, это схема, получающаяся отражением схемы K относительно вертикальной прямой, проведённой через середину схемы.

Введём целый параметр $k \in [\frac{n}{4} + \frac{1}{2}\log_2 n, \frac{n}{2} - \log_2 n]$. Часть блоков будем строить сразу в зависимости от параметра k.

1) $KI(a_1, \cdots, a_s)$ из [4] и $KI'(a_1, \cdots, a_s)$ — повторяющие схемы размером $s \times 1$ и $3s \times 1$ соответственно, принимающие снизу переменные $x_1, ..., x_s$ и выдающие справа и слева соответственно элементарную конъюнкцию $x_1^{a_1}, x_2^{a_2}, \cdots, x_s^{a_s}$

$$\left(\text{здесь } x^a := x \oplus a = \begin{cases} x, a = 0; \\ \overline{x}, a = 1. \end{cases} \right).$$

$KI(a_1, \cdots, a_s)$ состоит из следующих элементов:

$(1, 1) : t = b, r = (b \oplus a_1);$
$(j, 1) : t = b, r = l(b \oplus a_j) \text{для} j = 2, ..., s.$

Его площадь равна s.

$KI'(a_1, \cdots, a_s)$ — из следующих:

$(1, 1) : t = b, l = (b \oplus a_1);$
$(2, 1) : l = r;$
$(3, 1) : l = r;$
$(j, 1) : \begin{cases} t = b, l = r(b \oplus a_l) \text{ для } j = 3l - 2, l = 2, \cdots, s. \\ l = r \text{ для } j \neq 3l - 2, l = 2, \cdots, s. \end{cases}$

Площадь этого блока равна $3s$.

2) $D(x_1, \cdots, x_s)$ и $D'(x_1, \cdots, x_s)$ — декодеры от s переменных. $D(x_1, \cdots, x_s)$ состоит из блоков $KI(a_1, \cdots, a_s)$, стоящих на чётных позициях для всех $\{a_1, \cdots, a_s\} \in \{0, 1\}^s$. На нечётных позициях стоят коммутационные элементы (l = r, t = b), которые соединяют i-й выход предыдущего блока KI с i-м входом следующего блока, а так же передают входы блока D справа налево через весь

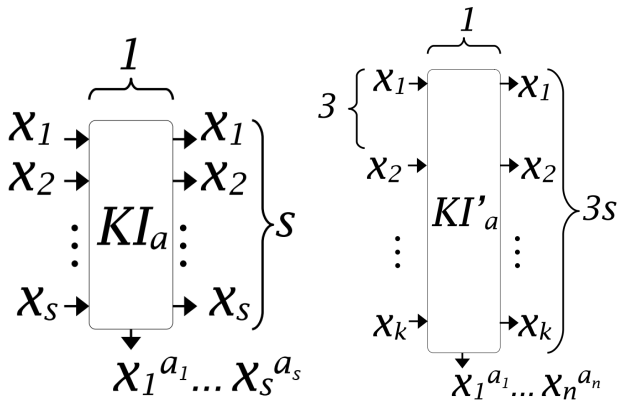


Рис. 3. Структура блоков KI_a и KI'_a, где $a = (a_1, a_2, \cdots, a_b)$, для всех $i\, a_i \in \{0, 1\}$

блок. Его площадь равна $2s2^s$. Этот блок совпадает с транспонированным и зеркально отражённым блоком Q_2, построенным в работе [4]. В русскоязычной литературе этот блок часто называется дешифратором.

$D'(x_1, \cdots, x_s)$ состоит из блоков $KI'(a_1, \cdots, a_s)$ для всех $\{a_1, \cdots, a_s\} \in \{0, 1\}^s$. Его площадь равна $3s2^s$. При построении

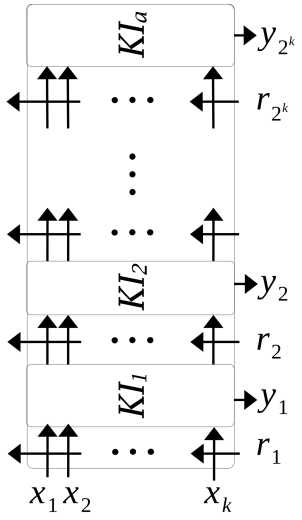


Рис. 4. Структура блока D.

схемы H использованы блоки $D_1 = D'(x_1, \cdots, x_{k-\log_2 k})$, $D_2 = D'(x_1, \cdots, x_k)$ и $D_3 = D(x_1, \cdots, x_k)$.

Лемма 3. *Пусть вход блока D_1 меняется раз в k тактов, вход блока D_2 — раз в 2^k тактов, а входы блока D_3 меняются не более чем $\frac{2^{2k}}{k}$ раз за период. Тогда*

$$\begin{array}{lll} l(D_1) = 3(k - \log_2 k), & w(D_1) = \frac{2^k}{k}, & W(D_1) \lesssim 6\frac{2^k}{k}. \\ l(D_2) = 3k, & w(D_2) = 2^k, & W(D_2) \lesssim 6k. \\ l(D_3) = k, & w(D_3) = 2 \cdot 2^k, & W(D_3) \lesssim 4 \cdot 2^{3k-n}. \end{array}$$

Доказательство, так же как и доказательства всех подобных лемм ниже, основано на применении леммы 2.

3) $e'_a(x_1, \cdots, x_{n-2k})$ и $e_a(x_1, \cdots, x_{n-2k})$ — повторяющие схемы размером $n - 2k \times 1$ и $3(n - 2k) \times 1$, принимающие снизу переменные $x_1, ..., x_{n-2k}$ и выдающие слева $\bigwedge_{i=1}^{n-2k} x_i \equiv a_i$, где $a = (a_1, \cdots, a_{n-2k}) \in \{0,1\}^{n-2k}$. Иными словами, блоки посылают налево единицу тогда и только тогда, когда вектор значений переменных на входе равен a. Устройство блоков представлено на рисунках 5 и 6.

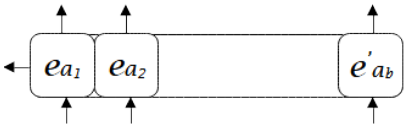


Рис. 5. Структура блока e'_a, где $a = (a_1, a_2, \cdots, a_b)$, для всех $i\, a_i \in \{0,1\}$

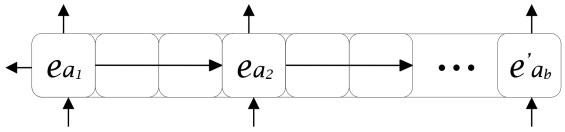


Рис. 6. Структура блока e_a

Лемма 4. *Пусть вход блока меняется не более чем 2^{n-2k} раз за период. Тогда*

$$l(e_a) = n - 2k, \quad w(e_a) = 1, \quad W(e_a) \preceq \frac{n - 2k}{2^{2k}}.$$

4) $\kappa_a(x_1, \cdots, x_k, e, r, b)$, где $a = (a_1, \cdots, a_k) \in \{0,1\}^s$ — посылает на выходы вниз последовательность a длины s если на входы e, r, b подана 1, иначе все выходы тождественно равны входам напротив них. $z = e, l = r, t = b, y_i = a_i(ebr)|x_i\overline{(ebr)}$ для $i = 1, \cdots, s$. Схема имеет вид, изображённый на рисунке 7. Клеточные элементы,

использованные при построении схемы, приведены на рисунке 1.

$$l(\kappa_a) = k + 3, \quad w(\kappa_a) = 2.$$

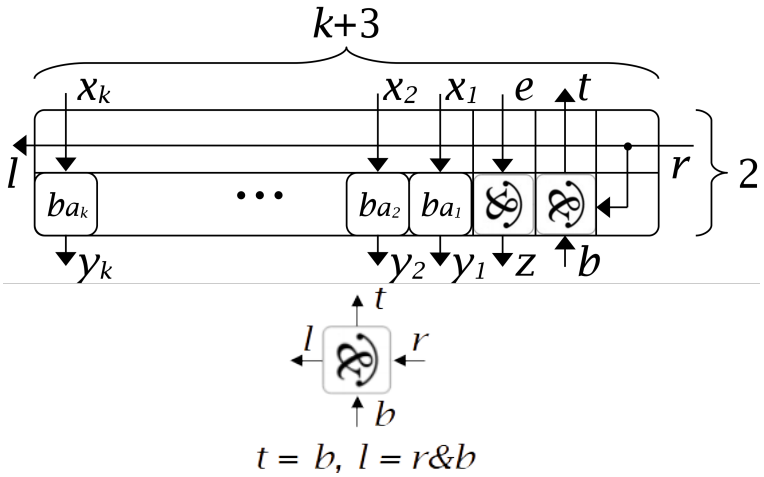


Рис. 7. Структура подблока $\kappa_a, a = (a_k, a_{k-1}, \ldots, a_2, a_1)$.

5) $K_\alpha(e_1, \cdots, e_{2^k/k}, b_1, \cdots, b_{2^k/k}, r_1, \cdots, r_{2^k})$ где $\alpha = (a_1, \cdots, a_{2^{2k}/k})$, $a_j \in \{0,1\}^k$— генератор $2^{2k}/k$ последовательностей длины k из списка α. Если все входы e_s равны 1, среди входов b_s только вход с номером j равен 1, а остальные 0, и среди входов r_s только вход с номером i равен 1, а остальные 0, то на выходе схемы снизу i-ая группа проводов будет содержать последовательность $a_{(i-1)\cdot 2^k/k+j}$, а остальные нули. Схема состоит из схем κ_a, на входы x_i схем κ_{a_j} подана константа 0 для $j = 1, 2, \cdots, 2^k/k$, то есть на верхние схемы в блоке. Схема имеет вид, изображённый на рисунке 8.

$$l(K_\alpha) = 2^k(1 + 3/k), \quad h(K_\alpha) = 2^{k+1}.$$

Лемма 5. *Рассмотрим блок*
$K_\alpha(e_1, \cdots, e_{2^k/k}, b_1, \cdots, b_{2^k/k}, r_1, \cdots, r_{2^k})$. *Пусть входы* e_s *равны единице с некоторого момента* $t_0 < 2^n - 2^{2k}$ *и не более чем* 2^{2k} *тактов подряд за период* 2^n*, причём за это время входы* b_s *и* r_s *изменяются следующим образом.*

$$b_i = \begin{cases} 1, \textit{если } t \in \cup_{s=1}^{2^k-1}[t_0 + (i-1)k + s2^k, t_0 + ik + s2^k]; \\ 0, \textit{иначе.} \end{cases}$$

$$r_i = \begin{cases} 1, \textit{если } t \in [t_0 + (i-1)2^k, t_0 + i2^k]; \\ 0, \textit{иначе.} \end{cases}$$

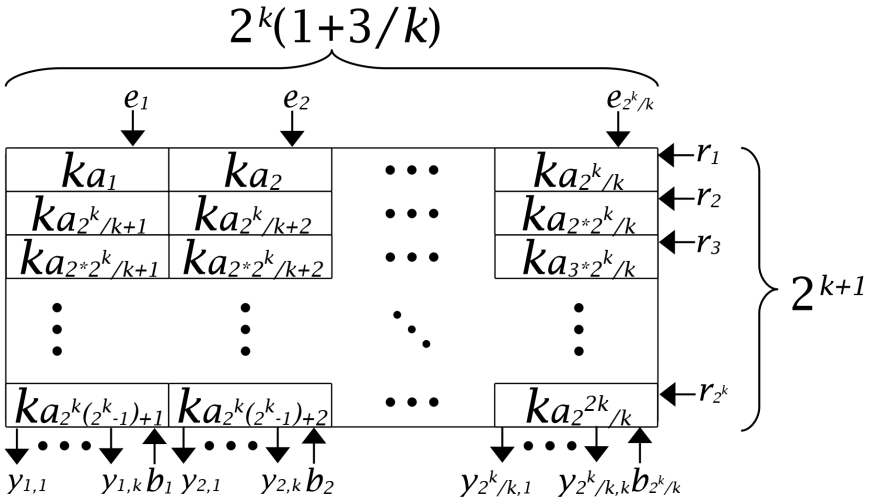


Рис. 8. Структура блока K.

Сначала на b_1 мы k тактов удерживаем 1, а на остальных 0, затем на b_2 мы k тактов удерживаем 1 и так далее. Как только на последнем $b_{2^k/k}$ происходит смена единицы на 0, единица подаётся на b_1 и всё повторяется снова, пока не пройдёт 2^{2k} с момента t_0. Начиная с такта $t_0 + 2^{2k}$ и до такта 2^n на b_i удерживается 0. На r_i всё происходит точно так же, только переключаются они через 2^k тактов и после выключения r_{2^k} на них 0.

Тогда

$$W(K_\alpha(e_1, \cdots, e_{2^k/k}, b_1, \cdots, b_{2^k/k}, r_1, \cdots, r_{2^k})) \lesssim 4 \cdot 2^{3k-n}.$$

Доказательство. При указанном изменении входов блок K раз в k тактов проводит не более чем на всю свою ширину полоску ширины k. Изменение на проводах e_s происходит дважды за период. $\sum\limits_{s=1}^{2^k/k} \omega(e_s) = 2 \cdot 2^{k+1} \frac{2^k}{k}$. Изменение на проводах r_s происходит 2^k раз за период. Каждое изменение затрагивает только два провода длины 2^k асимптотически. Изменение на проводах b_s происходит $\frac{2^{2k}}{k}$ раз за период. Каждое изменение затрагивает только два провода длины 2^{k+1} асимптотически. Тогда

$$W(K_\alpha) \lesssim \frac{1}{2^n}\left(2k \cdot 2^{k+1}\frac{2^{2k}}{k} + 2^{k+1}2\frac{2^k}{k} + 2 \cdot 2^{2k} + 2\frac{2^{2k}}{k}2^{k+1}\right) \lesssim 4 \cdot 2^{3k-n}.$$

□

6) $U(x_{1,1},\cdots,x_{k,1},x_{1,2},\cdots,x_{k,2},\cdots,x_{k,2^k/k})$ — блок дизъюнкций. Схема дизъюнктирует $\frac{2^k}{k}$ групп по k проводов в одну группу из k проводов. Имеет выходы $y_i = \bigvee\limits_{j=1}^{2^k/k} x_{i,j}, i = 1,\cdots,k$. Схема изображена на рисунке 9.

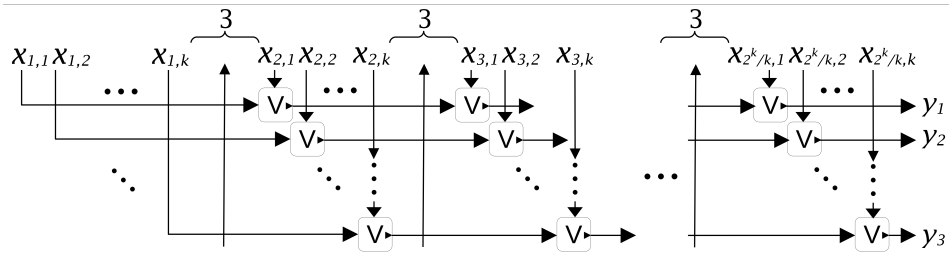


Рис. 9. Структура блока U.

Лемма 6. *Пусть вход блока меняется за период не более чем* $\frac{2^{2k}}{k}$ *раз. Тогда*

$$l(U) = 2^k(1+3/k), \quad w(U) = k, \quad W(U) \lesssim 2\cdot 2^{3k-n}.$$

7) $B_{a,\delta}(r_1,\cdots,r_{2^k},b_1,\cdots,b_{2^k/k},\eta_1,\cdots,\eta_{n-2k})$, где $a \in \{0,1\}^{n-2k}, \delta = (\delta_1,\cdots,\delta_{2^{2^s}/s}), \delta_j \in \{0,1\}^s$ — повторяющая схема, подающая на не тождественно равные входам выходы не ноль если $a = \eta = (\eta_1,\cdots,\eta_{n-2k})$. В этом случае если на входах справа $r_1,\cdots,r_{2^k}$ есть единственная единица на позиции $i \in [1,2^k]$, а на входа $b_1,\cdots,b_{2^k/k}$ есть единственная единица на позиции $j \in [1,2^k/k]$, то на выходы направо подаётся декодированная последовательность $\delta_{(i-1)2^k+j}$. Площадь блока асимптотически равна $2\cdot 2^{2k}$ при условии $n \lesssim 2^k$. Схема изображена на рисунке 10. При построении схемы H, реализующей последовательность $\alpha = \overline{\alpha_1,\cdots,\alpha_{2^{n-2k}}}$, используются блоки $B_i := B_{i,\alpha_i}, i = 1,\cdots,2^{n-2k}$.

$$l(B) \lesssim 2^k(1+3/k), \quad w(B) \lesssim 2^{k+1}.$$

Лемма 7. *Рассмотрим переключательную мощность блока* $B_{a,\delta}(r_1,\cdots,r_{2^k},b_1,\cdots,b_{2^k/k},\eta_1,\cdots,\eta_{n-2k})$ *за исключением множества проводов, проводящих* η_j*:* $\cup_{j=1}^{n-2k}\omega(\eta_j)$*. Пусть входы* η_s *равны*

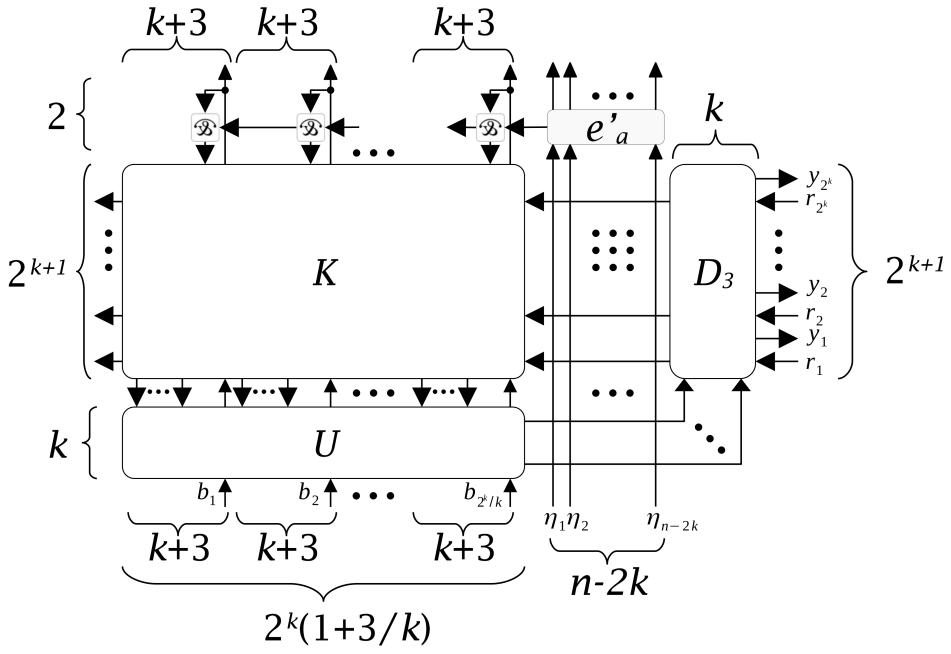


Рис. 10. Структура блока B.

а начиная с некоторого такта t_0 не более чем 2^{2k} тактов подряд за период 2^n, причём за это время входы b_s и r_s изменяются следующим образом.

$$b_i = \begin{cases} 1, \text{если } t \in \cup_{s=1}^{2^k-1}[t_0 + (i-1)k + s2^k, t_0 + ik + s2^k]; \\ 0, \text{иначе.} \end{cases}$$
$$r_i = \begin{cases} 1, \text{если } t \in [t_0 + (i-1)2^k, t_0 + i2^k]; \\ 0, \text{иначе.} \end{cases}$$

Сначала на b_1 мы k тактов удерживаем 1, а на остальных 0, затем на b_2 мы k тактов удерживаем 1 и так далее. Как только на последнем $b_{2^k/k}$ происходит смена единицы на 0, единица подаётся на b_1 и всё повторяется снова, пока не пройдёт 2^{2k} с момента t_0. Начиная с такта $t_0 + 2^{2k}$ и до такта 2^n на b_i удерживается 0. На r_i всё происходит точно так же, только переключаются они через 2^k тактов и после выключения r_{2^k} на них 0. Определим $\omega'(\eta) := \cup_{j=1}^{n-2k}\omega(\eta_j)$. Тогда

$$W(B_{a,\delta} \setminus \omega'(\eta)) \lesssim 10 \cdot 2^{3k-n}.$$

Доказательство. Доказательство не зависит от a, δ и прочих индексов блоков, поэтому опустим их при упоминании блоков.

Заметим, что для всех блоков в составе блока B переключательные мощности считались именно при тех входных последовательностях, которые получатся на входах блоков при выполнение условий леммы. Сигнал от блока e_a по пути до блока K затрагивает площадь не более $2 \cdot 2^k(1+3/k)$, обозначим элементы в этой площади $\omega'(e_a, K)$; эта же часть схемы проводит изменение по вертикальным проводам, а значит изменение на ней происходит не чаще чем $\frac{2^{2k}}{k}$ раз за цикл. Так же надо учесть провода, проходящие сквозь D_3 и U. В первом случае изменение происходит не чаще чем $2^k/k$ раз за период на двух проводах и расстоянии $n-k$. Во втором — не чаще $2^{2k}/k$ раз на двух проводах и расстоянии k. Обозначим эти провода $\omega'(r) = \cup_{j=1}^{2^k}\omega(r_j)$ и $\omega'(b) = \cup_{j=1}^{2^k/k}\omega(b_j)$ соответственно. Итого

$$W(B \setminus \omega'(\eta)) \lesssim$$
$$W(K) + W(U) + W(D_3) + W(e) + W(\omega'(e,K)) + W(\omega(U, D_3) +$$
$$W(\omega'(r)) + W(\omega'(b)) \lesssim 4 \cdot 2^{3k-n} + 2 \cdot 2^{3k-n} + 4 \cdot 2^{3k-n} + C\frac{n-2k}{2^{2k}} +$$
$$\frac{1}{2^n}\left(2^k(1+3/k) \cdot 2\frac{2^{2k}}{k} + 2k2^{2k}\right) \lesssim 10 \cdot 2^{3k-n}$$

при $n \lesssim 2^k, k < \frac{n}{2}$. □

8) $C^a(T)$ — счётчик от 0 до a. $a \in \{0,1\}^s$ является натуральным числом в двоичном представлении. Пока на входе T удерживается единица, счётчик каждый такт увеличивает своё значение до тех пор, пока не достигнет значения a. В тот такт, на котором счётчик достиг значения a, на выход O подаётся единица. В следующий такт счётчик имеет значение 0. Блок имеет площадь $6 \cdot 3(\lceil \log_2 a \rceil + 2)$. При построении схемы H используются как обычные, так и транспонированные или зеркально отражённые блоки $C^a(T)$. Устройство схемы изображено на рисунке 11. Выход O счётчика C_j будем обозначать O_{C_j}. $C_1 := C^k(1)$, C_2 — транспонированная схема $C^{k-\log_2 k}(O_{C_1})$, C_3 — транспонированная схема $C^k(O_{C_2})$, C_4 — зеркально отражённая схема $C^{n-2k}(O_{C_3})$.

Лемма 8.

$l(C_1) = 6,$	$w(C_1) = 3(\log_2 k + 2),$	$W(C_1) \preceq \log_2 k.$
$l(C_2) \lesssim 3(\log_2 k + 2),$	$w(C_2) = 6,$	$W(C_2) \preceq \frac{k-\log_2 k}{k}.$
$l(C_3) = 3(k+2),$	$w(C_3) = 6,$	$W(C_3) \preceq \frac{k}{2^k}.$
$l(C_4) = 6,$	$w(C_4) = 3(n-2k+2),$	$W(C_3) \preceq \frac{n-2k}{2^{2k}}.$

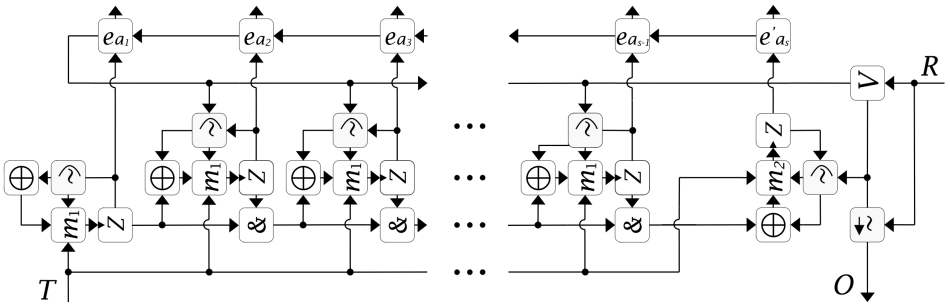


Рис. 11. Структура блока C.

Доказательство. Для оценки переключательной мощности воспользуемся леммой 2. Для корректного применения леммы будем рассматривать каждую задержку как один выход и несколько входов схемы. В такой интерпретации входы блока C_1 меняются каждый такт, блока C_2 — раз в k тактов, блока C_3 — раз в 2^k тактов, а входы блока C_4 меняются 2^{n-2k} раз за период. □

9) $Q_\alpha(x_1, \cdots x_{n-2k})$, где $\alpha = (a_1, \cdots, a_{2^{n/2-k}}), a_i \in \{0,1\}^{n-2k}$ — повторяющая схема. На единственный не тождественно равный входам выход подаётся единица тогда и только тогда, когда на вход подано одно из чисел a_j. Блок состоит из транспонированных блоков $e_a(x_1, \cdots, x_{n-2k})$ и дизъюнкций. Структура блока представлена на рисунке 12.

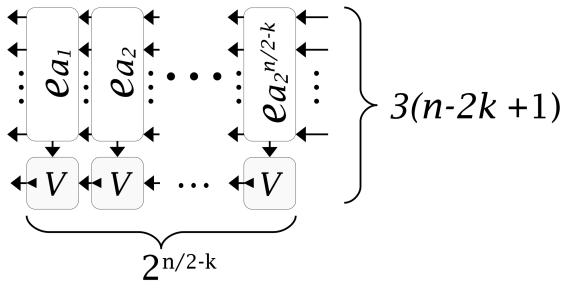


Рис. 12. Структура блока Q_α, где $\alpha = (a_1, a_2, \cdots, a_{2^{n/2-k}})$, для всех i $a_i \in \{0,1\}^{n-2k}$

Лемма 9. *Пусть вход схемы меняется не чаще, чем* 2^{n-2k} *раз за цикл. Тогда*

$$l(Q_\alpha) = 2^{n/2-k}, \quad w(Q_\alpha) = n - 2k + 1, \quad W(Q_\alpha) \lesssim 2(n-2k)2^{n/2-3k}.$$

10) $E(x_1, \cdots, x_{2^k})$ — энкодер. Номера входов возрастают сверху вниз. Схема реализует на выходе номер в двоичном представлении самого нижнего входного провода, по которому поступила единица. Структура представлена на рисунке 13.

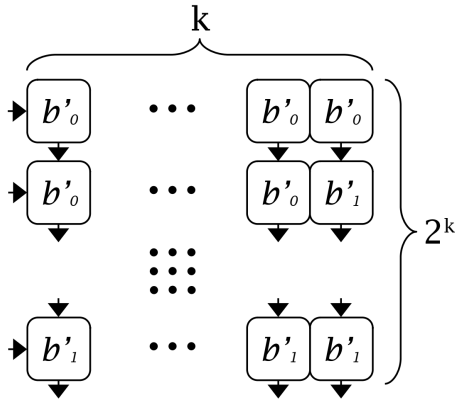


Рис. 13. Структура блока E.

Лемма 10. *Пусть вход блока меняется не чаще, чем раз в k тактов. Тогда*

$$l(E) = k, \quad w(E) = 2^k, \quad W(E) \lesssim 2 \cdot 2^k.$$

11) $f(x_1, \cdots, x_k, T)$ — очередь длины k, запоминающая по сигналу T значения входов $x_1, \cdots, x_k$. Каждый такт блок подаёт на выход следующее значение в очереди. Структура блока представлена на рисунке 14. С учётом замечания относительно задержек, сделанного при рассмотрении блоков C, можно считать, что блок работает каждый такт.

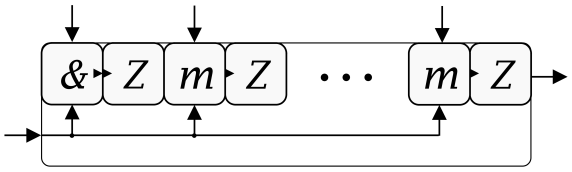


Рис. 14. Структура блока f.

Лемма 11.

$$l(f) = 2k, \quad w(f) = 2, \quad W(f) \lesssim 8k.$$

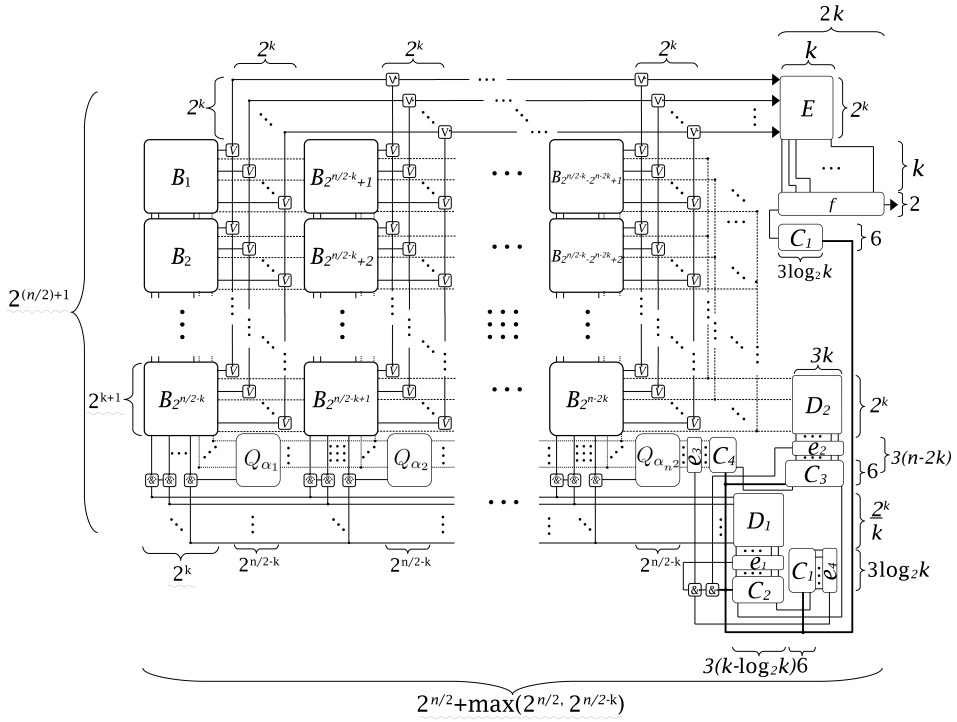


Рис. 15. Структура схемы H, реализующей последовательность α. Волнистой линией подчёркнуты асимптотические значения.

2.6. Описание схемы

Схема H представлена на рисунке 15. На рисунке так же обозначены размеры некоторых блоков и размеры всей схемы. Числа, подчёркнутые волнистой линией - асимптотические значения размеров.

Опишем схему H. Она состоит из 2^{n-2k} блоков B, расположенных в $2^{n/2-k}$ столбцах по $2^{n/2-k}$ блоков, пяти счётчиков C_1, C_1, C_2, C_3, C_4, декодеров D_1, D_2, энкодера E, очереди f и блоков $Q_{\alpha_1}, \cdots, Q_{\alpha_{2^{n/2-k}}}$.

Счётчик C_1 работает всё время, каждые k тактов он посылает сигнал счётчику C_2. Счётчик C_2 определяет номер в строке подблока K активного блока B. Этот номер декодируется и доводится до каждого блока B. Как только C_2 переполнился,то есть достиг последнего номера, он обнуляется и отправляет сигнал в счётчик C_3. Счётчик C_3 определяет номер строки подблока K активного блока B. Этот номер так же декодируется и доводится до всех блоков B. Как только C_3 переполнился, он обнуляется и отправляет сигнал в счётчик C_4. Последний счётчик определяет номер активного блока B. Как только он переполняется, схема

начинает заново генерировать последовательность, так как в это момент все счётчики так же сброшены. Провода, по которым идёт номер последовательности в строке изображены на рисунке 15 пунктирной линией, провода, по которым идёт номер строки — прерывистыми.

Сигналы от счётчиков C_2 и C_3 декодируются и доводятся до блоков B. Блоки B устроены так, что они активны, то есть изменяются вне коммутационных элементов, только если число на выходах C_4 равно номеру блока. В любом состоянии блок проводит насквозь без изменений сигнал от C_4 и декодированные сигналы от C_2 и C_3. Рассмотрим работу активного блока. В K приходят две единицы: одна по какому-то проводу r_i вторая по какому-то проводу b_j. На «пересечении»этих проводов в блоке κ содержится некоторая полоска длины k. Она отправляется вниз, с помощью U приходит в D_3, где декодируется. В итоге циклическое перещёлкивание входов некоторого блока B последовательно заставляет его посылать на выход декодированные полоски длины k. В одном блоке B суммарная длина полосок составляет 2^{2k}, тогда всего блоков будет 2^{n-2k}. Как только очередной блок отдал все свои полоски, увеличивается значение на счётчике C_4, и активным становится следующий в столбце снизу блок. Когда в столбце заканчиваются блоки, активным становится верхний блок следующего столбца и так далее. Декодированная полоска передаётся к блоку E и превращается в ту самую полоску длины k, которая лежала в блоке B. По сигналу от счётчика C_1 очередь f запоминает во все свои задержки биты этой полоски, после чего на протяжении k тактов посылает их последовательно на выход всей схемы.

Блоки Q обеспечивают прохождение сигнала от C_2 (это наиболее часто меняющийся сигнал от счётчиков, проходящий достаточно большое расстояние) только в столбец с активным блоком B. Таким образом изменение сигнала затрагивает число элементов, равное по порядку периметру схемы, а не её площади.

Возможен случай, когда в последнем столбце блоков B есть пустые блоки, то есть блоки, не содержащие ни одного кусочка генерируемой последовательности. Так же возможен случай, когда есть блок B, в котором заполнены не все строки и/или не все полоски в строке. Для этого предусмотрен механизм прерывания. По заданным 2^n и k легко вычислить номер последнего блока B, в котором есть значащие полоски, так же легко вычислить номер последней горизонтальной полосы и полоски длины k, содержащих часть генерируемой последовательности. Перед каждым управляющим счётчиком ставится блок e_j, проверяющий на равенство номеру последнего блока для счётчика C_4, номеру полосы для счётчика C_3, номеру полоски длины k и номеру от начала полосы. Когда все эти равенства выполнены, то схема H выдала последний бит последовательности. В этот же такт по проводам, отмеченным на рисунке 15

жирной линией, проходит сигнал на сброс счётчиков в начальное состояние.

2.7. Оценка переключательной мощности

Оценим переключательную мощность всех блоков B. Легко заметить что входы блоков B в схеме H ведут себя в точности так, как описано в лемме 7. Как следует из устройства схемы, на каждый блок последовательно подаются снизу декодированные числа от 0 до $\frac{2^k}{k}$, потом справа декодированное число увеличивается на 1 и всё повторяется снова до достижения декодированного числа 2^k на проводах справа.

Провода из D_1 проходят

$$(2^{n/2} + \max(2^{n/2}, 2^n - 2k) + \frac{2^{n/2}}{k} + 2^{n/2+1}.$$

Первые два слагаемых отвечают за длину горизонтальной шины, — за шины, идущие от горизонтальной шины к конъюнкциям, а последние два — за длину одной вертикальной шины. Такой результат достигается за счёт блокировки сигнала на входе во все столбцы, кроме столбца с "активным" блоком B. Аналогично провода из D_2 проходят $2^{n/2+1} + (2^k + \max(2^{n/2-k}, 2^k))2^{n-2k} \approx 2^{n/2+1} + 2^{n-k} + \max(2^{3n/2-3k}, 2^{n-k})$ асимптотически.

Провода, соединяющие E и f, имеют длину не более $2k$, их ровно k штук, значение на них меняется не более чем $\frac{2^n}{k}$ раз.

Провод, ведущий от Q_{α_j} к конъюнкциям под столбцом блоков B, имеет дину $n - 2k$, значение на нём меняется дважды за цикл, так как номера блоков B перебираются последовательно.

Наконец, сама декодированная подпоследовательность, генерируемая блоком B, проходит длину не более $3 \cdot 2^{n/2} + \max(2^{n/2}, 2^n - 2k)$ асимптотически раз в k тактов, то есть $\frac{2^n}{k}$ раз.

Назовём $\omega'(D_1)$ — провода, ведущие от блока D_1 к конъюнкциям под столбцами блоков B и провода, выходящие из этих конъюнкций. $\omega'(B)$ — все коммутационные элементы и дизъюнкции, выходящие из блоков B_j направо. Проще говоря, это те элементы схемы, через которые из блоков B_j в блок E_1 проходят декодированные подпоследовательности. В итоге

переключательная мощность схемы H оценивается как

$$W(H) \lesssim 2^{n-2k}W(B) + 2W(C_1) + W(C_2) + W(C_3) + W(C_4) + W(D_1) + W(D_2) + W(E) + W(f) + W(\omega(C_2, C_3)) + W(\omega(C_3, C_4)) + W(\omega'(D_1)) + W(\omega(D_2)) + W(\omega(E, f)) + W(\omega(C_4)) + W(\omega'(B)) \lesssim$$

$$10 \cdot 2^k + \frac{1}{2^n}\Bigg(36 \cdot 2^n \log_2 k + 36(k - \log_2 k)\frac{2^n}{k} + 36k2^{n-k} + 36(n-2k)2^{n-2k} + 6 \cdot 2^k \frac{2^n}{k} + 6 \cdot 2^k k 2^{n-k} + 2 \cdot 2^k k \frac{2^n}{k} + 8k2^n + \frac{2^k}{k}2^{n-k} + 2 \cdot 2^k 2^{n-2k} + 2\left(3 \cdot 2^{n/2} + \max\left(2^{n/2}, 2^{n-2k}\right)\right)\frac{2^n}{k} + 2\left(2 \cdot 2^{n/2} + 2^{n-k} + \max\left(2^{n-k}, 2^{3n/2-3k}\right)\right)2^{n-k} + k \cdot 2k\frac{2^n}{k} + k\left(2^{n/2} + \max\left(2^{n/2}, 2^{n-2k}\right) + 2 \cdot 2^{n-k}\right)2^{n-2k} + 2\left(3 \cdot 2^{n/2} + \max\left(2^{n/2}, 2^{n-2k}\right)\right)\frac{2^n}{k}\Bigg) \lesssim$$

$$10 \cdot 2^k + 6\frac{2^k}{k} + 2 \cdot 2^k + 3\frac{2^{n/2}}{k} + \max\left(\frac{2^{n/2}}{k}, \frac{2^{n-2k}}{k}\right)2 \cdot 2^{n/2-k} + 2^{n-2k} + \max\left(2^{n-2k}, 2^{3n/2-4k}\right) + 6\frac{2^{n/2}}{k} + \max\left(2\frac{2^{n/2}}{k}, 2\frac{2^{n-2k}}{k}\right) \lesssim$$

$$12 \cdot 2^k + 9\frac{2^{n/2}}{k} + 3\max\left(\frac{2^{n/2}}{k}, \frac{2^{n-2k}}{k}\right) + 2^{n-2k} + \max\left(2^{n-2k}, 2^{3n/2-4k}\right) \tag{1}$$

Вспомним, что $k \in [\frac{n}{4} + \frac{1}{2}\log_2 n, \frac{n}{2} - \log_2 n]$. Легко заметить, что при других значениях k порядок $W(H)$ увеличится: при уменьшении k увеличится порядок пятого слагаемого в 1, а при увеличении вырастет порядок первого слагаемого в 1. Тогда

$$W(H) \lesssim 12 \cdot 2^k + 12 \cdot \frac{2^{n/2}}{k} + 2 \cdot 2^{n-2k}.$$

$$W(H) \lesssim \begin{cases} 50\frac{2^{n/2}}{n}, & \text{если } k = \frac{n}{4} + \frac{1}{2}\log_2 n, \\ 12\frac{2^{n/2}}{k}, & \text{если } k \in (\frac{n}{4} + \frac{1}{2}\log_2 n, \frac{n}{2} - \log_2 n), \\ 36\frac{2^{n/2}}{n}, & \text{если } k = \frac{n}{2} - \log_2 n. \end{cases}$$

В схеме использовано $n + k + \log_2 k$ задержек. Лемма 1 доказана.

Таким образом при выборе $k = \frac{n}{2} - 2\log_2 n$ достигается минимум по асимптотике $12\frac{2^{n/2}}{n}$ при использовании $\frac{3}{2}n$ задержек асимптотически.

Однако, если взять $k = \frac{n}{4} + \log_2 n$, то достаточно использовать $\frac{5}{4}n$ задержек асимптотически. Платой будет ухудшение асимптотики до $48\frac{2^{n/2}}{n}$.

3. Заключение

В работе показано, что применение задержек в плоских схемах позволяет улучшить оценку переключательной мощности относительно тривиального использования результата, известного для плоских схем, реализующих булевы функции. В дальнейшем планируется с одной стороны двигаться в сторону получения нижней оценки переключательной мощности реализации автоматными схемами периодических последовательностей, а с другой в сторону получения оценок переключательной мощности реализации автоматными схемами произвольных автоматов.

Список литературы

[1] Кравцов С. С., "О реализации функций алгебры логики в одном классе схем из функциональных и коммутационных элементов", *Проблемы кибернетики*, **19** (1967), 285–293.

[2] Жуков Д. А., "О вычислении частичных булевых функций клеточными схемами", *Дискретный анализ и исследование операций*, **11**:1 (2004), 32-40.

[3] Шкаликова Н. А., "О реализации булевых функций схемами из клеточных элементов.", *Математические вопросы кибернетики*, **2** (1989), 177–197.

[4] Г. В. Калачев, "Порядок мощности плоских схем, реализующих булевы функции", *Дискрет. матем.*, **26**:1 (2014), 49–74.

[5] Г. В. Калачев, "Обобщение оценок мощности плоских схем, реализующих частичные булевы операторы", *Вестн. Моск. Ун-та.*, **3**, Сер. 1 Математика. Механика. (2018), 60-64.

[6] Г. В. Калачев, "Нижние оценки мощности плоских схем, реализующих частичные булевы операторы", *Интеллектуальные системы. Теория и приложения*, **18**:2 (2014), 279-322.

[7] Касим-Заде О. М., "Об одной мере активности схем из функциональных элементов.", *Математические вопросы кибернетики*, **4** (1992), 218–228.

[8] Шуткин Ю. С. , “Асимптотически оптимальная реализация булевых функций информационными графами.”, *Дискретная математика*, **23**:4 (2011), 80–102.

[9] Г. В. Калачев, “Оценки мощности плоских схем, реализующих монотонные функции”, *Интеллектуальные системы. Теория и приложения*, **21**:2 (2017), 163–192.

[10] С. А. Ложкин, В. С. Зизов, “Уточнённые оценки сложности дешифратора в модели клеточных схем из функциональных и коммутационных элементов”, *Учёные записки казанского университета. Серия физико-математические науки*, **162**:3 (2020), 322-334.

[11] Кудрявцев В. Б. , Алёшин С. В. , Подколзин А. С. , *Введение в теорию авоматов: Монография*, 2, Издательство Московского университета, Москва, 2019, ISBN: 978-5-19-011370-9.

Upper estimations of autonomous automata planar scheme
Vorotnikov A.S.

Switchable power of flat automatic schema without inputs implementation by periodical sequence is recieved in this work. A scheme is given that implements arbitrary pre-defined sequence of length 2^n for positive integer n with switchable power no more than $\frac{2^{n/2}}{n}$.

Keywords: boolean circuit , finite automata, models of VLSI, planar circuit, circuit power, Shannon function, upper estimations.

References

[1] S. Kravtsov, “About realisation boolean functions in one class logic circuits over the set of functional and commutation elements”, *Problems of cybernetics*, **19** (1967), 285-293 (In Russian).

[2] D. Zhukov, “On the calculation of partial Boolean functions by cellular schemes”, *Discrete analysis and operations researchs*, **11**:1 (2004), 32-40 (In Russian).

[3] N. Shkalikova, “On the implementation of Boolean functions by schemes of cellular elements”, *Mathematical questions of cybernetics*, **2** (1989), 177-197 (In Russian).

[4] G. Kalachev, “The power order of planar circuits implementing Boolean functions”, *Discrete mathematics*, **26**:1 (2014), 49-74 (In Russian).

[5] Kalachev G., "Generalization of power estimates of planar schemes implementing partial Boolean operators", *Bulletin of the Moscow University*, **3**, Series 1 Mathematics. Mechanics. (2018), 60-64 (In Russian).

[6] Kalachev G., "Lower power estimates of planar schemes implementing partial Boolean operators", *Intelligent systems. Theory and applications*, **18**:2 (2014), 279-322 (In Russian).

[7] Kasim-Zade O., "About one measure of the activity of circuits from functional elements.", *Mathematical questions of cybernetics*, **4** (1992), 218-228 (In Russian).

[8] Shutkin Y., "Asymptotically optimal implementation of Boolean functions by information graphs.", *Discrete mathematics*, **23**:4 (2011), 80-102 (In Russian).

[9] G. Kalachev, "Power estimates of flat circuits implementing monotone functions", *Intelligent systems. Theory and applications*, **21**:2 (2017), 163-192 (In Russian).

[10] S. Lozhkin, V. Zizov, "Refined estimates of the complexity of the decoder in the model of cellular circuits from functional and switching elements", *Scientific notes of Kazan University. Series of physical and mathematical sciences*, **162**:3 (2020), 322-334 (In Russian).

[11] Kudriavtsev V.B., Aleshin S.V., Podcolzin A.S., *Introduction to automata theory: Monograph*, 2, Moscow University Publishing, Moscow, ISBN: 978-5-19-011370-9 (In Russian).

Анализ графов-кактусов с использованием автоматов: свойства и время распознавания

А. А. Демидова[1]

Данная работа посвящена исследованию применения автоматов со стираемыми красками для определения того, является ли произвольный связный плоский простой неориентированный граф кактусом. Приводится алгоритм для определения данного свойства, а также нижняя и верхняя оценки числа шагов, которое должен совершить автомат для завершения обхода.

Ключевые слова: Автоматы, графы, графы-кактусы.

1. Введение

Подробный обзор области теории автоматов, связанной с обходами лабиринтов, был представлен в работе [1]. Функционирование автомата в лабиринтах и на графах отличается, поскольку в последнем случае отсутствует компас, который позволил бы получать дополнительную информацию об окружающей среде [2]. Работа [3] посвящена известным результатам в области автоматов, осуществляющих обход графов.

Обходы прямоугольных лабиринтов и графов автоматами с красками рассматривались, в частности, в работах [4] – [6].

В работе [7] исследовалось применение автоматов с красками для определения того, является ли произвольный связный плоский простой неориентированный граф деревом и псевдодеревом, и было доказано, что для определения этих свойств автомату достаточно двух стираемых красок.

В данной работе рассматривается исследование автоматом того, является ли граф, обход которого он осуществляет, графом-кактусом. Графом-кактусом является такой связный граф, в котором любое ребро принадлежит не более чем одному циклу, а любые два цикла могут иметь не более одной общей вершины.

Графы-кактусы могут применяться при моделировании сетевых структур, которые не получится отразить с помощью деревьев. При этом дерево, как и псевдодерево, является частным случаем графа-кактуса, и

[1] *Демидова Анна Андреевна* — аспирант каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: anna.dem98@mail.ru.

Demidova Anna Andreevna — graduate student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

изначально рассматриваемый тип графов назывался деревьями Хусими в честь работавшего над ними Коди Хусими (название было дано Ф. Харари и Дж. Ю. Уленбеком в [8]).

Графы-кактусы могут использоваться для декомпозиции набора геномов на ряд цепочек и подсетей для дальнейшего индексирования и сравнения [9]. Графы этого класса также могут применяться в теории коммуникационных сетей [10] и других областях.

2. Основные понятия и формулировка результатов

Обозначим через $\mathbf{G}$ класс всех связных плоских простых неориентированных графов. Будем считать, что до начала обхода графа все его рёбра имеют серый цвет.

В данной работе рассматривается автомат $\mathcal{A}_c$, осуществляющий обход графа из класса $\mathbf{G}$ с целью определения того, является ли этот граф кактусом. Во время исследования графа автомат обладает частичной информацией о посещаемых вершинах и инцидентных им рёбрах. По умолчанию обход осуществляется по правилу левой руки.

В распоряжении автомата есть 5 стираемых красок, а также серая краска, которой изначально были окрашены все рёбра графа.

Основным результатом является Теорема 1.

Теорема 1. *Существует автомат $\mathcal{A}_c$ с 6 красками, который сможет установить, является ли произвольный граф $\mathbf{g} \in \mathbf{G}$ кактусом.*

Одним из частных случаев графа-кактуса является дерево. Далее будем предполагать, что в рассматриваемом графе есть циклы.

Произвольный цикл делит плоскость на две грани, и до установления его наличия автомат обходит подграфы, имеющие общие вершины с этим циклом и лежащие только в одной их них.

Максимальные подграфы, имеющие общие вершины с рассматриваемым циклом и лежащие в той грани, которую автомат обходил до установления его наличия, будем называть **левыми ветвлениями**.

Единственное левое ветвление, обход которого мог быть не закончен к моменту обнаружения цикла, является первым – тем, из которого автомат попал на цикл.

Максимальные подграфы, имеющие общие вершины с рассматриваемым циклом и лежащие в той грани, которая до обнаружения цикла не была посещена автоматом, будем называть **правыми ветвлениями**.

Перед обходом правых ветвлений очередного цикла автомат меняет направление движения по правилу левой руки на противоположное тому,

в котором изначально обходил этот цикл. После того, как исследование правых ветвлений цикла завершено, автомат снова меняет направление обхода.

«Галочка» – пара соседних рёбер, лежащих на цикле и перекрашиваемых автоматом в фиолетовый или красный цвет перед началом обхода правого ветвления, исходящего из вершины, являющейся общей для этих рёбер.

Число «галочек» на цикле совпадает с количеством принадлежащих ему вершин. Соседние «галочки» на одном и том же цикле имеют общее ребро, поэтому при постановке очередной «галочки» автомату нужно будет перекрашивать только одно ребро.

Алгоритм, которому следует автомат $\mathcal{A}_c$, можно описать следующим образом:

1) Автомат осуществляет обход графа по правилу левой руки и наносит чёрную краску на все рёбра, по которым проходит, пока не обнаружит цикл. Наличие цикла устанавливается в ситуации, когда автомат только что покрасил некоторое ребро в чёрный цвет и оказался в вершине, которой инцидентны другие не серые рёбра.

2) Автомат ставит «галочку» в вершине, в которой было установлено наличие цикла. При постановке первой «галочки» на цикле автомат должен перекрасить последнее ребро, по которому он проходил до момента обнаружения цикла, а также первое не серое ребро справа. Оба эти ребра до перекрашивания являются чёрными, а выбор первого не серого ребра справа обусловлен тем, что правее него ещё может быть не обойдённое правое ветвление. Если рассматриваемый цикл – самый первый из обнаруженных автоматом, то для размещения «галочки» используется красный цвет, а в противном случае – фиолетовый.

3) При постановке первой «галочки» очередного цикла автомат также перекрашивает в синий или голубой цвет первое левое ребро в первом левом ветвлении этого цикла. Данное ребро будет указывать направление движения после завершения обхода правых ветвлений цикла. Автомат использует синюю краску для перекрашивания чёрных рёбер и голубую – для фиолетовых. Если соответствующее ребро является серым или красным, то перекрашивание не происходит.

4) Автомат приступает к обходу правых ветвлений рассматриваемого цикла, поменяв при этом направление движения по правилу левой руки на противоположное. Перед обходом очередного правого

ветвления автомат ставит «галочку» в вершине цикла, из которой это ветвление исходит, причём выбирать нужно будет первое не серое слева, поскольку направление движения после начала обхода правых ветвлений цикла сменилось на противоположное. Все «галочки» на графе, кроме самой первой, поставленной на первом обнаруженном цикле, имеют фиолетовый цвет.

5) После завершения обхода правых ветвлений цикла автомат переходит в его первое левое ветвление по отмеченному ранее синему, голубому или серому ребру. В случае, если возможных рёбер несколько, выбирается самое левое из них. Если ребро было голубым, то автомат перекрашивает его в фиолетовый цвет. В противном случае автомат использует чёрную краску.

6) Автомат устанавливает, что граф не является кактусом, и завершает обход в 4 ситуациях, которые будут описаны в разделе 5.

7) Если автомат не обнаружил признаков того, что граф не является кактусом, то обход завершается при возвращении из первого левого ветвления первого обнаруженного цикла к «галочке» красного цвета. В случае, если у данного цикла не было первого левого ветвления, исследование графа завершается после завершения обхода всех правых ветвлений данного цикла.

3. Возможное взаимное расположение циклов, имеющих общие рёбра, в графе, не являющемся кактусом

Лемма 1. *Допустим, что автомат при обходе графа установил наличие цикла, у которого на самом деле есть общие рёбра с другими циклами графа. Тогда эти новые циклы могут лежать только в правых ветвлениях только что обнаруженного цикла.*

Доказательство. Предположим противное – в левом ветвлении только что обнаруженного цикла есть другой цикл, имеющий с ним по крайней мере одно общее ребро. Поскольку все левые ветвления обнаруженного цикла, кроме, возможно, самого первого, уже были обойдены, а граф является плоским, то из первого левого ветвления нельзя сразу попасть ни в одну вершину уже обнаруженного цикла, не миновав ту, из которой это ветвление исходит. Следовательно, в первом левом ветвлении обнаруженного цикла не может быть цикла, который имел бы с ним общие рёбра. Таким образом, рассматриваемые циклы могут лежать только в правых ветвлениях обнаруженного цикла. □

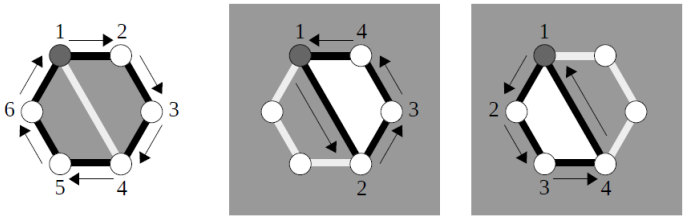


Рис. 1. Возможные варианты обхода графа, не являющегося кактусом, до обнаружения первого цикла, а также области, в которых могут лежать левые и правые ветвления

На Рисунке 1 представлены 3 варианта обхода одного и того же графа, не являющегося кактусом, в которых автомат начинает из одной и той же вершины, но в первый момент времени начинает двигаться в разных направлениях. Области, в которых лежат правые ветвления обнаруженных циклов, выделены цветом. Во всех 3 ситуациях второй цикл, мешающий графу быть кактусом, лежит в правом ветвлении.

4. Правые ветвления и «галочки»

Допустим, что автомат установил наличие цикла и приступил к обходу правых ветвлений. Перед тем, как начать исследование очередного правого ветвления, автомат перекрашивает в фиолетовый цвет рёбра, инцидентные текущей вершине цикла и лежащие на этом цикле. На Рисунках 2а-г представлены различные ситуации, отражающие результаты обхода правого ветвления, соответствующего вершине, в которой только что была поставлена «галочка». На Рисунке 2, как и на всех, которые будут представлены далее, рёбра, раскрашенные в цвета кроме серого и чёрного, отмечены соответствующими буквами: красные ребра помечены буквой «К», фиолетовые — буквой «Ф». Ситуация на Рисунке 2д невозможна. Рассмотрим все эти случаи подробнее.

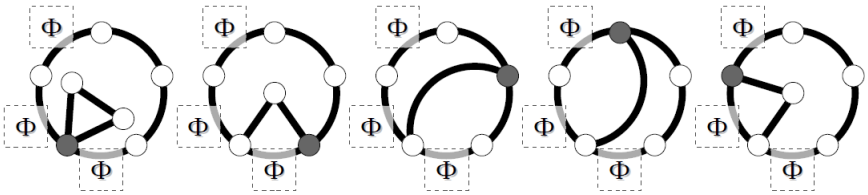


Рис. 2. Разбор ситуаций

На Рисунке 2а представлен граф-кактус. В правом ветвлении может не быть цикла, но автомат в любом случае попадает в вершину, которой инциденты 2 фиолетовых ребра, и видит слева ребро этого цвета. При этом автомат мог только что красить некоторое ребро в чёрный цвет (если в ветвлении есть цикл, имеющий общую вершину с исходным, и он был только что обойдён) или нет (если таких циклов в ветвлении нет или же цикл есть, но он не имеет общих вершин с рассматриваемым).

На Рисунке 2б представлен граф, не являющийся кактусом. Автомат, приступив к обходу правого ветвления некоторой вершины исходного цикла, попадает в другую вершину цикла, в которой ещё не была поставлена фиолетовая «галочка», однако «галочка» уже была проставлена в предыдущей вершине цикла.

На Рисунке 2в также представлен граф, не являющийся кактусом. Автомат, приступив к обходу правого ветвления некоторой вершины исходного цикла, попадает в другую вершину цикла. Этот случай отличается от представленного на Рисунке 2б тем, что «галочки» нет даже в предыдущей вершине цикла.

Представленный на Рисунке 2г случай отличается от 2б только тем, что автомат попадает из правого ветвления в вершину, правое ветвление которой должно было быть исследовано в последнюю очередь. Одно из рёбер, инцидентных этой вершине, уже было ранее окрашено в фиолетовый цвет, потому это было необходимо для того, чтобы оставить «галочку» в самой первой вершине, правое ветвление которой исследовал автомат.

Ситуация, представленная на Рисунке 2д, не может возникнуть при обходе, поскольку, если бы в графе действительно был расположенный так второй цикл, то ещё на стадии исследования правого ветвления той вершины, в которую якобы попадает автомат, он оказался бы в одной из уже рассмотренных ранее ситуаций.

5. Признаки того, что граф не является кактусом

Опишем 4 ситуации, в которых автомат будет устанавливать, что граф не является кактусом.

1) **Автомат обнаруживает новый цикл и тут же видит ровно одно фиолетовое или красное ребро** (Рисунок 3).

   У автомата есть возможность поставить первую «галочку» на обнаруженном цикле, однако он сразу прерывает обход. Если фиолетовое ребро — первое не серое справа, то следует перейти к следующему случаю.

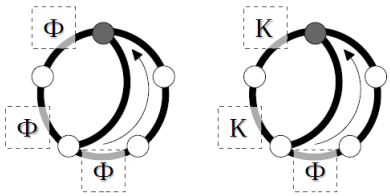


Рис. 3. Автомат устанавливает, что граф не является кактусом, и не ставит первую «галочку» на только что обнаруженном цикле

2) **Автомат обнаруживает новый цикл, и ему негде ставить второе ребро первой «галочки», поскольку первое не серое ребро справа не является чёрным** (Рисунок 4).

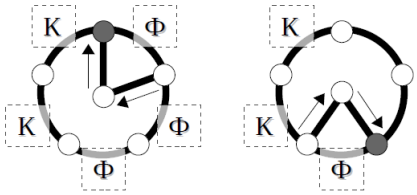


Рис. 4. Автомат устанавливает, что граф не является кактусом, так как негде поставить одно из рёбер первой «галочки» на только что обнаруженном цикле

3) **Автомат обнаруживает новый цикл, успешно ставит первую «галочку» на нём, но обнаруживает, что на данном цикле уже есть по крайней мере одно фиолетовое ребро** (Рисунок 5).

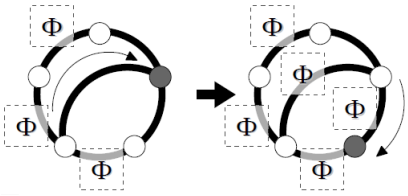


Рис. 5. Автомат устанавливает, что первая «галочка» на обнаруженном цикле на самом деле не является первой

Автомат ставит «галочку» и сталкивается с одиночным фиолетовым ребром или нечётным числом фиолетовых рёбер. Одинокое фиолетовое ребро (оно же — первое не серое справа) должно лежать на том же цикле — следовательно, это не первая «галочка» на цикле, что означает, что граф не является кактусом.

4) **Автомат ставит последнюю «галочку» на цикле (первое не серое ребро слева — фиолетовое), но, если пройти по первому не серому ребру слева, то обнаружится, что там, помимо этого ребра, чётное число фиолетовых рёбер, то есть нет следующего ребра на данном цикле — следовательно, решение автомата о том, что он завершил обход очередного цикла, является ошибочным** (Рисунок 6).

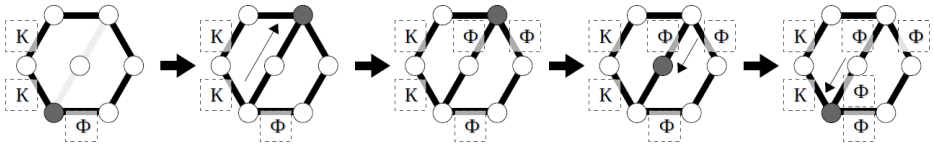


Рис. 6. Автомат устанавливает, что цикл на самом деле не обойдён до конца

Покажем, что такая ситуация не может возникнуть при обходе графа-кактуса. Допустим, что автомат завершил обход правых ветвлений некоторого цикла. В графе-кактусе «галочки», лежащие на разных циклах, не могут иметь общих рёбер. В кактусе после завершения обхода правых ветвлений цикла первое фиолетовое ребро слева будет лежать либо на том же цикле, либо в его первом правом ветвлении. Поскольку данное ветвление должно было быть обойдено полностью, оно добавляет в текущей вершине чётное число фиолетовых рёбер. При этом текущий цикл также добавляет чётное число фиолетовых или красных рёбер. Следовательно, в графе-кактусе такая ситуация не возникнет.

6. Обеспечение наличия левого ветвления в некоторой вершине рассматриваемого цикла

В алгоритме для псевдодеревьев в [7] автомат должен был оставлять нестандартную метку в одном из левых ветвлений. При повторном обходе обнаруженного цикла с целью установления наличия левых и правых ветвлений автомат получал информацию о том, есть ли какие-то ветвления. Если все вершины, посещённые автоматом, имели степень 2, то никаких ветвлений не было и граф просто являлся циклом. Если были вершины степени больше 3, причём автомат был в ситуациях, когда вершине были инцидентны хотя бы 2 чёрных ребра, то в графе были левые ветвления. Если же автомат посещал вершины степени 3, но признаков левых ветвлений не обнаружил, то в цикле есть только правые ветвления. В таком случае граф перекрашивал всё обратно в серый цвет (в

силу отсутствия левых ветвлений «всё» включало в себя только обнаруженный цикл), после чего менял направление движения по правилу левой руки на противоположное, тем самым превращая старые правые ветвления в новые левые, и начинал обход заново.

В алгоритме для псевдодеревьев признаком окончания обхода было «левое ребро чёрное, правое ребро серое, сейчас автомат ничего не красил».

Рассмотрим следующую ситуацию: автомат нашёл цикл и начал исследование ещё не обойдённых его ветвлений, где обнаружил ещё один цикл. Если новый цикл лежит в первом левом ветвлении первого, то направление обхода не менялось, и для нового цикла первый будет в левом ветвлении. Если новый цикл лежит в одном из правых ветвлений исходного, то направление обхода к этому моменту уже поменялось, так что исходный цикл опять же будет лежать в левом ветвлении нового.

Наличие первых левых ветвлений у циклов необходимо для того, чтобы была возможность размещать синие или голубые рёбра. Ситуация, в которой автомату будет негде ставить эти рёбра, может возникнуть только в том случае, если наличие цикла было обнаружено в вершине степени 2 или вершине большей степени, но такой, что все инцидентные рёбра лежат в правых ветвлениях. Такая ситуация может возникнуть только в той вершине, откуда начинался обход.

7. Порядок изменения цветов рёбер при обходе графа автоматом

Автомат может перекрашивать рёбра следующим образом:

1) **Серый** → **Чёрный**: стандартная операция при обходе новых рёбер;

2) **Чёрный** → **Фиолетовый**: размещение «галочек»;

3) **Чёрный** → **Красный**: размещение первой «галочки» на первом обнаруженном цикле;

4) **Чёрный** → **Синий**: размещение синего ребра;

5) **Фиолетовый** → **Голубой**: размещение голубого ребра.

6) **Синий** → **Чёрный** и **Голубой** → **Фиолетовый**: автомат идёт туда, где ещё нужно что-то обойти, после завершения обхода правых ветвлений некоторого цикла.

Таблица 1. Возможные изменения цвета ребра за один шаг

Старый\новый цвет	Сер.	Чёрн.	Фиол.	Красн.	Син.	Гол.
Серый	X	+	−	−	−	−
Чёрный	−	X	+	+	+	−
Фиолетовый	−	−	X	−	−	+
Красный	−	−	−	X	−	−
Синий	−	+	−	−	X	−
Голубой	−	−	+	−	−	X

Невозможные действия:

1) **Серый** → **Фиолетовый** и **Серый** → **Красный**: только рёбра циклов могут быть перекрашены в фиолетовый или красный цвет. Если ребро серое, то его принадлежность какому-либо циклу ещё не установлена;

2) **Серый** → **Синий** и **Серый** → **Голубой**: в синий цвет могут быть перекрашены только чёрные рёбра, а в голубой – только фиолетовые. Использование этих красок на сером ребре могло бы привести к тому, что новый цикл будет обнаружен при перекрашивании какого-то ребра не в чёрный цвет;

3) **Любой цвет** → **Серый**: перекрашивание обратно в серый цвет невозможно;

4) **Фиолетовый** → **Чёрный** и **Фиолетовый** → **Красный**: фиолетовые рёбра не могут быть перекрашены обратно в чёрный цвет. Красных рёбер в графе всего 2, и они сразу были перекрашены из чёрных в красные;

5) **Синий** → **Фиолетовый** и **Синий** → **Красный**: синие рёбра могут сразу стать только чёрными;

6) **Голубой** → **Чёрный** и **Голубой** → **Красный**: голубые рёбра могут сразу стать только фиолетовыми;

7) **Чёрный** → **Голубой** и **Фиолетовый** → **Синий**: синяя и голубая краски нужны для того, чтобы отличать рёбра, которые раньше были чёрными и фиолетовыми;

8) **Красный** → **Любой цвет**: красные рёбра остаются красными до конца обхода;

9) **Синий** → **Голубой** и **Голубой** → **Синий**.

8. Оценки времени установления автоматом свойства графа быть графом-кактусом

В качестве одного шага алгоритма будем рассматривать проход автомата по некоторому ребру в одну сторону вместе со всеми действиями, которые могут сопровождать этот переход. Такие действия, в частности, подразумевают выбор ребра и то, нужно ли его красить. Вне зависимости от того, выбирает автомат первое ребро слева или, например, первое не серое ребро справа, на это действие вместе с проходом по ребру и возможным его перекрашиванием уходит один шаг алгоритма. Анализ ситуации в текущей вершине происходит мгновенно, то есть автомату не нужен дополнительный шаг для того, чтобы установить, что он обнаружил цикл, и так далее.

1) Рёбра, не лежащие на циклах, нужно проходить 2 раза (больше шагов нужно только для тех рёбер, которые потом становятся синими). Первое левое ветвление первого цикла в худшем случае нужно будет полностью обходить 2 раза, то есть проходить по каждому его ребру 4 раза;

2) По каждому ребру каждого цикла необходимо пройти по крайней мере 4 раза (больше шагов нужно только для тех рёбер, которые потом становятся голубыми): один раз – когда автомат проходит по нему в первый раз и красит в чёрный цвет, два раза – когда автомат ставит очередную «галочку» и возвращается в её центр, и ещё один раз – когда автомату нужно переместиться в следующую вершину цикла;

3) Каждому циклу нужно по одному синему или голубому ребру: 2 шага уходят на то, чтобы его поставить, и ещё 1 – чтобы потом перекрасить это ребро в чёрный или фиолетовый цвет;

4) В конце обхода правых ветвлений необходимо проверить, что следующее ребро на этом цикле действительно фиолетовое или красное. Для этого необходимо осуществить 2 шага.

Пусть в графе есть k рёбер и m циклов, а общее число рёбер, содержащихся в циклах, равно n. Нижняя оценка для числа шагов совпадает с полученной в работе [7] для деревьев и псевдодеревьев и следует из того, что по каждому ребру автомат пройдёт хотя бы 2 раза. Верхняя оценка числа шагов:

$$T \leq 4(k-n) + 4n + 3m + 2m, \tag{1}$$

где $4(k-n)$ – верхняя оценка числа шагов для обхода рёбер, не лежащих на циклах (умножение на 4 стоит из-за первого левого ветвления первого цикла – неизвестно, сколько там рёбер по сравнению с k),
$4n$ – число шагов для рёбер, принадлежащих циклам, от первого прохода до завершения обхода этих циклов;
$3m$ – число шагов для синих и голубых рёбер;
$2m$ – проверка.

Если в графе есть хотя бы один цикл, то $3 \leq n \leq k$. Кроме того, в каждом цикле есть по крайней мере 3 ребра, причём предполагается, что общих рёбер у циклов нет, – следовательно, $3m \leq n \leq k \Rightarrow m \leq \frac{n}{3} \leq \frac{k}{3}$. Тогда:

$$T \leq 4k + 5m \leq 4k + \frac{5k}{3} = \frac{17k}{3}. \quad (2)$$

9. Выводы

Представленный в данной работе алгоритм для автоматов, осуществляющих обход связных плоских простых неориентированных графов, позволяет с использованием 6 красок установить, является ли рассматриваемый граф кактусом.

Список литературы

[1] Кудрявцев В. Б., Килибарда Г., Ушчумлич Ш., “Системы автоматов в лабиринтах”, *Интеллектуальные системы*, **10**:1–4 (2006), 449–562.

[2] Blum M.,Kozen D., “On the power of the compass (or, why mazes are easier to search than graphs)”, *19th Annual Symposium on Foundations of Computer Science (sfcs 1978)*, 1978, 132–142.

[3] Okhotin A., “Graph-walking automata: from whence they come, and whither they are bound”, *International Conference on Implementation and Application of Automata*, 2019, 10–29.

[4] Насыров А. З., “Об обходе лабиринтов автоматами, оставляющими нестираемые отметки”, *Дискретная математика*, **9**:1 (1997), 123–133.

[5] Голованов А. В., “Об обходе лабиринтов автоматами, оставляющими след в вершинах лабиринта”, *Интеллектуальные системы*, **3**:3–4 (1998), 193–212.

[6] Голубев Д. В., “Об обходе графов автоматами с одной нестираемой краской”, *Интеллектуальные системы*, **4**:1–2 (1999), 243–272.

[7] Демидова А. А., “Автоматный анализ свойств графа быть деревом и псевдодеревом”, *Интеллектуальные системы. Теория и приложения*, **25**:2 (2021), 111–127.

[8] Harary F., Uhlenbeck G. E., “On the number of Husimi trees: I”, *Proceedings of the National Academy of Sciences*, **39**:4 (1953), 315–322.

[9] Paten B. et al., “Cactus graphs for genome comparisons”, *Journal of Computational Biology*, **18**:3 (2011), 469–481.

[10] Zmazek B., Zerovnik J., “Estimating the traffic on weighted cactus networks in linear time”, *Ninth International Conference on Information Visualisation (IV'05)*, 2005, 536–541.

Analysis of cactus graphs using automata: properties and recognition time
Demidova A.A.

This paper is devoted to the study of using automata with erasable colors to determine whether an arbitrary connected plane simple undirected graph is a cactus. An algorithm is given for determining this property, as well as lower and upper bounds of the number of steps that the automaton must take to complete the traversal.

Keywords: Automata, graphs, cactus graphs.

References

[1] Kudryavtsev V. B., Kilibarda G., Ušćumlić Š., “Automata systems in labyrinths”, *Intelligent Systems*, **10**:1–4 (2006), 449–562 (In Russian).

[2] Blum M.,Kozen D., “On the power of the compass (or, why mazes are easier to search than graphs)”, *19th Annual Symposium on Foundations of Computer Science (sfcs 1978)*, 1978, 132–142.

[3] Okhotin A., “Graph-walking automata: from whence they come, and whither they are bound”, *International Conference on Implementation and Application of Automata*, 2019, 10–29.

[4] Nasyrov A. Z., “On traversing labyrinths by automata that leave not-erasable marks”, *Discrete mathematics*, **9**:1 (1997), 123–133 (In Russian).

[5] Golovanov A. V., "On traversing labyrinths by automata that leave a trail at the vertices of the labyrinth", *Intelligent Systems*, **3**:3–4 (1998), 193–212 (In Russian).

[6] Golubev D. V., "On graph traversal by automata with one not-erasable paint", *Intelligent Systems*, **4**:1–2 (1999), 243–272 (In Russian).

[7] Demidova A. A., "Automaton analysis of the properties of a graph to be a tree and a pseudo-tree", *Intelligent Systems. Theory and Applications*, **25**:2 (2021), 111–127 (In Russian).

[8] Harary F., Uhlenbeck G. E., "On the number of Husimi trees, I", *Proceedings of the National Academy of Sciences*, **39**:4 (1953), 315–322.

[9] Paten B. et al., "Cactus graphs for genome comparisons", *Journal of Computational Biology*, **18**:3 (2011), 469–481.

[10] Zmazek B., Zerovnik J., "Estimating the traffic on weighted cactus networks in linear time", *Ninth International Conference on Information Visualisation (IV'05)*, 2005, 536–541.

О сложности A-выразимости элементарного базиса для A-замыкания в классах линейных автоматов над конечными полями

И. Ю. Ильин[1]

В предыдущей работе нами был получены оценки сложности реализации элементарного базиса в классе одноместных линейных автоматов, сохраняющих нулевую последовательность. В данной работе мы получили верхнюю оценку сложности реализации элементарного базиса через операции A-замыкания в классе линейных автоматов над конечным полем [4].

Ключевые слова: линейные автоматы, A-замыкание, A-выразимость, временная сложность алгоритма.

Мы продолжаем исследовать сложность реализации функций в различных классах линейных автоматов. В предыдущей работе нами были получены оценки сложности реализации элементарного базиса в классе линейных автоматов, сохраняющих нулевую последовательность. В данной работе мы будем решать задачу сложности реализации элементарного базиса для A-замыкания в классе линейных автоматов над конечным полем E_k [4], предварительно получив результаты для поля E_2. Для упрощения приведем используемые нами обозначения, определения и теоремы, подробные доказательства которых читатель сможет найти в работах [1] [2] [5].

§1. Оценка сложности τ-приближения функций элементарного базиса в классе линейных автоматов над полем E_2.

Мы рассмотриваем множество линейных автоматов L_2 над E_2 с операциями композиции:

1. Подстановки.
2. Переименования переменных.
3. Отождествления переменных.
4. Обратной связи.

[1] *Ильин Иван Юрьевич* — аспирант каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: vanyail@yandex.ru.

Ilin Ivan Yurievich— graduate student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

Обозначим $K(M)$ – замыкание множества M по операциям перечисленным операциям.

Введем множества T_a, $a \in E_2$ следующим образом:

$T_a = \{f | f \in L_2, f$ сохраняет a в начальный момент времени. $\}$ [1]

Переменная x_j функции $f(x_1, ..., x_n) = \sum_1^n \mu_i x_i + \mu_0, \mu_i \in E_2'(\xi)$ называется непосредственной, если $\mu_j(0) = 1$.

Введем ещё два множества:

$V_1 = \{f | f \in L_2, f$ имеет не более одной непосредственной переменной$\}$.

$V_2 = \{f | f \in L_2, f$ имеет нечетное число непосредственных переменных$\}$.

Обозначим $U(f) = \{\mu_i, i = 1, ..., n\}$.

Введем множество

$$M(\xi) = \{f | f \in L_2, \forall \mu \in U(f), \mu + \mu(0) = a_2\xi^2 + a_3\xi^3 + ...\}.$$

Множества $T_0, T_1, V_1, V_2, M(\xi)$ являются K-замкнутыми и предполными в L_2 [1].

Пусть $M \subseteq L_2, f \in L_2, \tau \in \mathbf{Z}_+$. Тогда функция $f = f(x_1, ..., x_n)$ является τ-выразимой через M, если существует функция $g(x_1, ...x_n) \in K(M)$ такая, что

$$\forall \alpha_i, \alpha_i = a_{0i}a_{2i}...a_{\tau i}, i = 1, 2, ..., n$$

$$f(\alpha_1, ..., \alpha_n) = g(\alpha_1, ..., \alpha_n).$$

Функция f называется A-выразимой через M, если $\forall \tau \in \mathbb{Z}_+$ функция f τ-выразима из M [5].

Пусть $M \subseteq L_2$ тогда A-замыканием M называется множество $A(M) = \{f | f \in L_2, f$ - A-выразима из $M\}$.

Множества $T_0, T_1, V_1, V_2, M(\xi)$ являются A-замкнутыми, далее будем обозначать $J_a = \{T_0, T_1, V_1, V_2, M(\xi)\}$.

Лемма 1. *Пусть множество $M \subseteq L_2$, M не содержится целиком ни в одном из множеств $\Theta \in J_a$. Пусть n - максимальная арность функций из M. Тогда мы можем получить константы γ_0, γ_1 такие, что $\gamma_0(0) = 0, \gamma_1(0) = 1$, используя $O(n)$ операций сложения и умножения, а также 1 операцию обратной связи.*

Доказательство. Так как M целиком не содержится в V_1, V_2, то существует функция $f(x_1, ..., x_n) = \sum_{i=1}^n \mu_i x_i + \mu_0$ имеющая четное количество непосредственных переменных. Рассмотрим функцию

$$g(x) = f(x, x, ..., x) = \sum_{i=1}^{n} \mu_i x + \mu_0 = \mu x + \mu_0,$$

которую мы получаем за $O(n)$ операций сложения. В силу того, что у функции f четное количество непосредственных переменных, $\mu(0) = 0$, а значит к функции g применима обратная связь. $F_b(g(x)) = \gamma$ – константа.

Предположим, что $\gamma(0) = 0$, тогда $\gamma \in T_0$. В M выберем функцию h не принадлежащую T_0 и подставим в неё константу γ: $h(\gamma, \gamma, ..., \gamma) = \gamma_1$, для чего используем $O(n)$ операций сложения и умножения.

Схожим образом поступим, если $\gamma(0) = 1$, в этом случае $\gamma \in T_1$. В M выберем функцию h' не принадлежащую T_1 и подставим в неё константу γ: $h'(\gamma, \gamma, ..., \gamma) = \gamma_0$, для чего используем $O(n)$ операций сложения и умножения.

А значит, мы получаем две константы γ_0, γ_1 такие, что $\gamma_0(0) = 0$, $\gamma_1(0) = 1$. □

Лемма 2. *Пусть множество $M \subseteq L_2$, M не содержится целиком ни в одном из множеств $\Theta \in J_a$, n - максимальная арность функций из M. Пусть также нами уже были получены две константы γ_0, γ_1 из леммы 1. Тогда мы можем получить функцию $\hat{f}(x_1, x_2, x_3)$, 2^s-эквивалентную $x_1 + x_2 + x_3$ за $O(s + n)$ операций.*

Доказательство. Существует функция $f'(x_1, ..., x_n)$, принадлежащая $M \setminus V_1$. Без ограничения общности, x_1, x_2 - непосредственные переменные. Подставим константу γ_0 вместо всех переменных, кроме x_1, x_2,

$$f'(x_1, x_2, \gamma_0..., \gamma_0) = \mu_1 x_1 + \mu_2 x_2 + \gamma'$$

за $O(n)$ операций сложения и умножения. Переменные x_1, x_2 - непосредственные, а значит $\mu_1(0) = \mu_2(0) = 1$. Получим функции

$$f_{\gamma_0, x_1}(x_1) = f'(\gamma_0, x_1, \gamma_0..., \gamma_0),$$

$$f_{x_2, \gamma_0}(x_2) = f'(x_2, \gamma_0, \gamma_0..., \gamma_0),$$

$$f''(x_1, x_2) = f'(f_{\gamma_0, x_1}(x_1), f_{x_2, \gamma_0}(x_2), \gamma_0..., \gamma_0),$$

$$f''(x_1, x_2) = \mu_1\mu_2 x_1 + \mu_1\mu_2 x_2 + \gamma''.$$

Для этого мы используем $O(n)$ операции сложения и умножения, $\mu = \mu_1\mu_2$ в нулевой момент принимает значение $\mu(0) = \mu_1(0)\mu_2(0) = 1$. Следовательно, μ имеет вид $1 + \xi\hat{\mu}$, где $\hat{\mu} \in E_2'(\xi)$. Получим функцию

$$f_1(x_1, x_2) = f''(f''(x_1, x_2), \gamma_0) = \mu^2 x_1 + \mu^2 x_2 + \gamma^{(1)}$$

за $O(1)$ операций сложения и умножения.

Предположим, что мы уже получили функцию $f_{s-1}(x_1, x_2)$. Построим теперь функцию $f_s(x_1, x_2)$:

$$f_s(x_1, x_2) = f_{s-1}(f_{s-1}(x_1, x_2), \gamma_0) =$$

$$= \mu^{2^{s-1}}\mu^{2^{s-1}} x_1 + \mu^{2^{s-1}}\mu^{2^{s-1}} x_2 + \gamma^{(s-1)}\mu^{2^{s-1}} =$$

$$= \mu^{2^s} x_1 + \mu^{2^s} x_2 + \gamma^{(s-1)} \mu^{2^{s-1}}$$
$$= \mu^{2^s} x_1 + \mu^{2^s} x_2 + \gamma^{(s)}$$

На каждой итерации такого процесса мы будем использовать $O(1)$ операций сложения и умножения, значит, всего на построение $f_s(x_1, x_2)$ потребуется $O(s)$ операций сложения и умножения.

Подставим функцию f_s в себя следующим образом:

$$f_s(x_1, f_s(x_1, x_3)) = \mu^{2^s} x_1 + \mu^{2^{s+1}} x_2 + \mu^{2^{s+1}} x_3 + \gamma^{(s)} + \mu^{2^s} \gamma^{(s)} =$$

$$= (1 + \xi^{2^s} \bar{\mu}) x_1 + (1 + \xi^{2^s} \tilde{\mu}) x_2 + (1 + \xi^{2^s} \mu') x_3 + \xi^{2^s} \hat{\gamma},$$

используя $O(1)$ операций сложения и умножения. Отсюда видно, что эта функция 2^s-эквивалентна $x_1 + x_2 + x_3$ и была построена за $O(s + n)$ операций сложения и умножения. Поскольку мы можем построить подобную функцию для любых s, то таким образом мы получили функцию $\hat{f}(x_1, x_2, x_3)$, 2^s-эквивалентную функции $x_1 + x_2 + x_3$. □

Заметим, что принимая $\tau = s$ в лемме 2, мы получаем функцию $\hat{f}(x_1, x_2, x_3)$, τ-эквивалентную функции $x_1 + x_2 + x_3$.

Лемма 3. *Пусть множество $M \subseteq L_2$, M не содержится целиком ни в одном из множеств $\Theta \in J_a$, n - максимальная арность функций из M. Пусть мы уже получили константы γ_0, γ_1 из леммы 1 и функцию $x_1 + x_2 + x_3$. Тогда мы можем получить функции, τ-эквивалентые задержке, сумматору $x_1 + x_2$, константе 0, используя $O(n+\tau)$ операций сложения и умножения.*

Доказательство. Из множества M выберем функцию $h(x_1, ..., x_r)$ не принадлежащую $M(\xi)$. Существует $\mu \in U(h)$ такая, что

$$\mu + \mu(0) = \xi + \xi^2 a_2 + ...$$

Будем считать, что

$$h(x_1, ..., x_r) = \mu x_1 + \mu_2 x_2 + ... + \mu_r \xi_r + \gamma.$$

Получим функцию $h_1(x) = h(x, \gamma_0, ..., \gamma_0) = \mu x + \gamma$ за $O(n)$ операций сложения.

Для того чтобы получить оценки количества операций сложения и умножения, повторим индуктивное доказательство, представленное в работе [1]. Если $\mu(0) = 1$, то получим из функции $h_1(x)$ и сумматора $x_1 + x_2 + x_3$ следующую функцию:

$$h_2(x) = h_1(x) + x + \gamma_0 = \mu x + \gamma + x + \gamma_0 = \xi \mu' x + \gamma'.$$

Если $\gamma'(0) = 1$, то прибавив к $h_2(x)$ константы γ_0, γ_1 получим

$$h_2(x) + \gamma_0 + \gamma_1 = \xi\mu' x + \tilde{\gamma}', \tilde{\gamma}'(0) = 0.$$

Таким образом мы можем считать, что в $h_2(x)$ константа $\gamma'(0) = 0$. Значит, $\gamma' = 0 + b_1\xi + b_2\xi^2 + ...$. Если $b_1 = 0$, то все доказано. Пусть $b_1 = 1$.

$$h_2(\gamma_0) = \xi\mu'\xi\tilde{\gamma} + \gamma' = \xi + b_2'\xi^2 + b_3'\xi^3 + ...$$
$$h_2(\gamma_1) = \xi\mu'\tilde{\tilde{\gamma}} + \gamma' = b_2''\xi^2 + b_3''\xi^3 + ...$$
$$\hat{h}_2(x) = h_2(x) + h_2(\gamma_0) + h_2(\gamma_1) = \xi\mu' x + \xi^2\mu'', \mu'(0) = 1.$$

Функция $\hat{h}_2(x)$, таким образом, 2-эквивалентна задержке. Теперь докажем для $\tau > 2$.

Предположим теперь, что

$$h_\tau(x) = (\xi^{\tau-1} + \xi^\tau\mu_{\tau-1})x + \xi^\tau\gamma_{\tau-1}.$$

Подставим $\hat{h}_2(x)$ в $h_\tau(x)$ и получим

$$h_\tau'(x) = (\xi^\tau + \xi^{\tau+1}\mu_\tau')x + \xi^\tau\gamma_\tau.$$

Если $\gamma_\tau(0) = 0$, то мы уже получили $h_{t+1}(x)$, Предположим, что $\gamma_\tau(0) \neq 0$, тогда подставим в сумматор $x_1 + x_2 + x_3$ функции $h_\tau'(x)$, $h_\tau'(\gamma_0), h_\tau'(\gamma_1)$ и получим,

$$h_\tau'(x) + h_\tau'(\gamma_0) + h_\tau'(\gamma_1) = (\xi^\tau + \xi^{\tau+1}\mu_\tau'')x + \xi^{\tau+1}\gamma_\tau' = h_{\tau+1}(x)$$

Заметим, что функция $h_{\tau+1}(x)$ будет τ-эквивалентна нулю. Следовательно, мы можем получить сумматор от двух переменных, подставив $h_{\tau+1}(x)$ на место любой переменной в сумматоре $x_1 + x_2 + x_3$. Суммируя функции $h_{i,c} = h_i(\gamma_1) + h_i(\gamma_0)$, мы можем получить константу с началом из $i-1$ нуля, $h_{i,c}(i) = 1$ и произвольным концом.

Снова рассмотрим функцию $\hat{h}_2(x)$ и переобозначим её как $f_1(x)$:

$$\hat{h}_2(x) = \xi\mu' x + \xi^2\mu'' = \xi x + \xi^2\mu''' x + \gamma_{1,1} = f_1(x), \mu'(0) = 1.$$

Функция $f_1(x)$ является 2-эквивалентной задержке с нулевым начальным состоянием. Получим теперь функцию $f_2(x)$:

$$f_1(x) = (\xi + \xi^2\mu''')x + \gamma_{1,1}, \mu'''(0) = a.$$
$$f_1(x) + (1 + \mu'''(0))h_3(x) = (\xi + \xi^3\mu^{(3)})x + \xi^2\gamma_{2,1}.$$
$$f_1(x) + (1+\mu'''(0))h_3(x) + (1+\gamma_{2,1}(0))h_3(\gamma_1) = (\xi+\xi^3\mu^{(3)})x + \xi^3\gamma_{2,2} = f_2(x).$$

Видно, что для получения $f_2(x)$ мы потратили $O(1)$ операций сложения и умножения. Функция $f_2(x)$ является 3-эквивалентной задержке с нулевым начальным состоянием. Продолжая данный процесс, получая $f_4, f_5, ...$ до времени τ, получим функцию $f_\tau(x)$, τ-эквивалентную задержке с нулевым начальным состоянием. На каждой итерации мы будем использовать $O(1)$ операций сложения и умножения, а значит всего, для получения $f_\tau(x)$, мы используем $O(n+\tau)$ операций сложения и умножения. □

Теорема 1. *1. Пусть множество $M \subseteq L_2$. M является A-замкнутым тогда и только тогда, когда M не содержится целиком ни в одном из множеств $\Theta \in J_a$.*

2. Пусть n - максимальная арность функций из M. Тогда за $O(\tau+n)$ операций сложения и умножения, а также одну операцию обратной связи мы можем получить функции τ-эквивалентные единице, задержке, сумматору от двух переменных.

Доказательство. Доказательство первого утверждения теоремы читатель может найти в работе [1]. Докажем второе утверждение.

В леммах 1,2,3 мы научились получать функции τ- эквивалентные задержке с нулевым начальным состоянием, константы γ_0,γ_1, где $\gamma_0(0) = 0$,$\gamma_1(0) = 1$, функцию $x_1 + x_2 + x_3$, сумматор $x_1 + x_2$, константу 0, задержку ξx за $O(n+\tau)$ операций сложения и умножения и одну операцию обратной связи. Получим функцию τ-эквивалентную единице.

Выпишем константу γ_1 до такта τ в виде ряда: $1 + a_1\xi + a_2\xi^2 + ... + a_\tau\xi^\tau + ...$

Получим константы $\gamma_1\xi^i, i = 1, ..., \tau$. Теперь прибавим к константе γ_1 константу $\gamma_1\xi$, если коэффициент $a_1 = 0$, и запишем это в таком виде:

$$\gamma_{1,1} = \gamma_1 + b_1\xi\gamma_1 = 1 + \xi + (b_1a_1 + a_2)\xi^2 + (b_1a_2 + a_3)\xi^3 + ...,$$

где $b_1 = 1$, если $a_1 = 0$, и $b_1 = 0$, если $a_1 = 1$.

Перепишем

$$\gamma_{1,1} = 1 + \xi + a_{2,1}\xi^2 + a_{3,1}\xi^3 + ...,$$

Теперь предположим, что мы уже получили $\gamma_{1,p}$, получим теперь $\gamma_{1,p+1}$:

$$\gamma_{1,p+1} = \gamma_{1,p} + b_{p+1}\xi^{p+1}\gamma_1 = 1 + \xi + ... + \xi^{p+1} + (b_{p+1}a_{p+1,p} + a_{p+2,p})\xi^{p+2} + ...,$$

где $b_{p+1} = 1$, если $a_{p+1,p} = 0$, и $b_{p+1} = 0$, если $a_{p+1,p} = 1$.

Перепишем

$$\gamma_{1,p+1} = 1 + \xi + ... + \xi^{p+1} + a_{p+2,p+1}\xi^{p+2} + ...$$

Отсюда видно, что на итерации τ константа $\gamma_{1,\tau}$ будет τ-эквивалентна единице. На каждой итерации мы используем $O(1)$ операций сложения, так как тактов всего τ, то мы используем для получения функции τ-эквивалентной единице $O(\tau)$ операций сложения и умножения. Подставляя оценки из предыдущих лемм, мы получим, что в общей сложности используется не более 1 операций обратной связи и $O(\tau + n)$ операций сложения и умножения. □

На этом доказательстве мы закончим исследование сложности τ-выразимости элементарного базиса в классе линейных автоматов над полем E_2 и перейдем к линейным автоматам над полем E_k.

§2. Оценка сложности τ-приближения функций элементарного базиса в классе линейных автоматов над полем E_k.

В данной главе под обозначением E_k мы будем понимать конечное поле, где $k = p^m, p$ - простое, $m \in \mathbb{N}$. [4]

Аналогично со случаем в E_2, определим следующие множества согласно работам [2] [1]:

$T_a = \{f | f \in L_k, f$ сохраняет $a \in E_k$ в начальный момент времени. $\}$

$V_1 = \{f | f \in L_k, f$ имеет не более одной непосредственной переменной$\}$.

$V_0 = \{f(x_1, ..., x_n) = \sum_{i=1}^{n} \mu_i x_i + \mu_0 | \sum_{i=1}^{n} \mu_i(0) = 1 (\mathrm{mod}\ p)\}$.

$U(f) = \{\mu_i, i = 1, ..., n\}$.

$M_1(\xi) = \{f | f \in L_k, \forall \mu \in U(f), \mu - \mu(0) \in \xi^2 E'_k(\xi)\}$.

$\hat{P}_1, \hat{P}_2, ..., \hat{P}_s$ - все максимальные собственные подполя в E_k.

$P_j = \{f | f \in L_k, \forall i, i = 1, ..., n\ \mu_i(0) \in \hat{P}_j\}$.

$J_k^A = \{T_a, V_0, V_1, M_1, P_j | a \in E_k,\ i \in \{1, ..., s\}\}$.

Определения τ-эквивалентности и A-полноты аналогичны случаю с автоматами над E_2.

Перечисленные выше классы J_k^A являются A-замкнутыми и предполными в L_k [1].

Аналогично со случаем L_2 мы сформулируем следующую лемму:

Лемма 4. *Пусть множество $M \subseteq L_k$, M не содержится целиком ни в одном из множеств $\Theta \in J_k^A$. Пусть также n - максимальная арность функций из M. Тогда мы можем получить константу $\gamma, \gamma(0) = a, a \in E_k$, используя $O(k + n)$ операций сложения и умножения, а также 1 операцию обратной связи.*

Доказательство. Так как M не содержится в V_1, то, без ограничения общности, будем считать, что у нас есть функция $f(x_1, ..., x_n) = \sum_{i=1}^{n} \mu_i x_i +$

μ_0 имеющая непосредственные переменные x_1, x_2. Тогда рассмотрим функцию $g(x_1, x_2, x) = f(x_1, x_2, x, ..., x) = \mu_1 x_1 + \mu_2 x_2 + \mu' x + \mu_0$, которую мы получаем за $O(n)$ операций сложения. Обозначим $\mu_i(0) = a_i, i = 1, 2$.

Так как k - порядок поля E_k, то мы можем возвести a_i в степень $k-1$, чтобы получить $a_i^{k-1} = 1$ [4]. Получим функции

$$g_1(x_1, x) = g(x_1, x, x), g_2(x_2, x) = g_2(x, x_2, x).$$

Подставим g_1, g_2 в себя $k-2$ раз:

$$h_1 = g_1(g_1(g_1(...(x_1, x)...x), x), x),$$

$$h_2 = g_2(g_2(g_2(...(x_2, x)...x), x), x).$$

Мы используем для этого $O(k)$ операций сложения и умножения, а затем подставим полученные функции в $g(x_1, x_2, x_3)$:

$$g(h_1(x_1, x), h_2(x_2, x), x) = \mu_1^{k-1} x_1 + \mu_2^{k-1} x_2 + \hat{\mu} x + \hat{\mu_0} = h(x_1, x_2, x).$$

$$\mu_1^{k-1}(0) = (\mu_1(0))^{k-1} = 1, \mu_2^{k-1}(0) = 1.$$

p-раз подставим в себя функцию h следующим образом:

$$H(x_1, ...x_{p+1}, x) = h(h(...(h(x_1, x_2, x), ..., x_p, x), x_{p+1}, x) =$$
$$= \tilde{\mu}_1 x_1 + \tilde{\mu}_2 x_2 + ... + \tilde{\mu}_{p+1} x_{p+1} + \tilde{\mu} x + \tilde{\mu}_0.$$

Таким образом, за $O(p)$ операций мы получили, что $\tilde{\mu_i}(0) = 1, i = 1, ..., p-1, \tilde{\mu}_0(0) = p\hat{\mu}(0) = 0$, причем переменная x не является непосредственной.

Теперь возьмем функцию $u(x_1, ..., x_{n'}) \in M \setminus V_0$.

$G(x) = u(x, x, ..., x) = \sum_{i=1}^{n'} \mu_i' x + \mu_0' = \mu x + \mu_0'$, получаем $G(x)$ за $O(n')$ операций сложения. $\mu(0) \neq 1$, $n' \leq n$.

Если $\mu(0) = 0$, то мы можем применить оператор обратной связи к $G(x)$, получим $Fb_x(G(x)) = \gamma$, где γ - константа.

Пусть $\mu(0) = a, a \neq 0$. Элемент a порождает подполе E' в E_k. $E_p \subseteq E'$.

Рассмотрим случай $p = 2$. Имеем $z^{k-1} - 1 = 0$, $z = 1$ - корень данного уравнения, $a^{k-1} - 1 = 0$. Следовательно, мы можем переписать уравнение в виде $(z+1)(z^{k-2} + ...) = 0$. Если количество слагаемых во второй части уравнения - чётное, то вторая часть уравнения делится на $z+1$. Будем выделять $z+1$ из второй части уравнения до того момента, когда количество слагаемых в нём не станет нечётным, получим уравнение $(z+1)^s(\sum_{i=0}^{r} a_i z^i) = 0$, $\sum_{i=0}^{r} a_i a^i = 0, a_i \in \{0, 1\}, \sum_{i=0}^{r} a_i = 1$. Так как

всего элементов в поле не более k, то на получение степеней μ^i мы используем не более $O(k)$ операций сложения и умножения. Из леммы 5 мы можем получить 3-сложение (текущая лемма не требуется для доказательства леммы 5). Используя 3-сложение и полученные степени a^i, получим дробь $\mu \sum_{i=0}^{r} a_i \mu^i$, для этого мы будем применять 3-сложение не более $O(k)$ раз. Таким образом мы получаем функцию $\hat{H}(x)$ такую, что $\hat{H}(0) = 0$, а значит к ней применима обратная связь. Применяя к $\hat{H}(x)$ обратную связь получим константу γ за $O(k+n)$ операций сложения и умножения и одну операцию обратной связи.

Рассмотрим случай $p > 2$. Умножая элемент a не более чем за $l < k$ раз, мы можем получить элемент $t \in \{2, 3, ...p-1\}$ подполя E_p [4]. Значит, подставляя $G(x)$ в себя не более чем k раз, получим функцию $G'(x) = (\mu')^l x + \mu''$, используя $O(k)$ операций сложения и умножения и одну операцию обратной связи.

Теперь подставим функцию $G'(x)$ в функцию H на место первых r' переменных, а на место остальных поставим x, где количество применений r' найдем ниже.

Получим $\hat{H}(x) = H(G'(x), ..., G'(x), x, x, ...x) = \hat{\mu}x + \tilde{\hat{\mu}}x + \hat{\mu}_0 = (\hat{\mu} + \tilde{\hat{\mu}})x + \hat{\mu}_0$. Переменная x не является непосредственной для функции $\hat{H}(x)$, т.к. $(\hat{\mu} + \tilde{\hat{\mu}})(0) = r't + (1 - r') = 0$, $r' = \frac{-1}{b-1} \in E_p$. А значит, применяя операцию обратной связи к функции $\hat{H}(x)$, получим костанту γ за $O(k+n)$ операций сложения и умножения и одну операцию обратной связи. $\square$

Заметим, что любую другую константу γ_2 мы можем получить из функции f_a, не принадлежащей классу T_γ, подставив в функцию f_a на место всех переменных константу γ, что можно сделать не более чем за $O(n)$ операций сложения.

Теперь получим операцию сложения $p+1$ элемента, т.е. докажем следующую лемму:

Лемма 5. *Пусть множество $M \subseteq L_k$, M не содержится целиком ни в одном из множеств $\Theta \in J_k^A$. Пусть также n - максимальная арность функций из M. Тогда мы можем получить функцию τ-эквивалентную функции $x_1 + x_2 + x_3 .... + x_{p+1}$ за $O(k + \tau + n)$ операций сложения и умножения.*

Доказательство. $\exists f(x_1, ..., x_{n'}) \in M \setminus V_1$. Пусть x_1, x_2 - её непосредственные переменные. Тогда подставим переменную x в остальные переменные функции f: $f_1(x_1, x_2, x) = f(x_1, x_2, x, ..., x) = \mu_1 x_1 + \mu_2 x_2 + \mu x + \mu_0$. $\mu_1(0) \neq 0, \mu_2(0) \neq 0$, используя $O(n)$ операций сложения. Подставим функцию f_1 в себя за $O(1)$ операций умножения и сложения:

$$f_2(x_1, x_2, x) = f_1(f_1(x, x_2, x), f_1(x_1, x, x), x) = \mu_1\mu_2 x_1 + \mu_1\mu_2 x_2 + \mu' x + \mu'_0.$$

Обозначим $\mu = \mu_1\mu_2, \mu(0) \neq 0$. Распишем μ в форме ряда и переобозначим этот ряд: $\mu = a_0 + a_1\xi + ... = a_0 + \xi\hat{\mu}, \hat{\mu} \in E'_k(\xi)$. Далее за $O(k)$ операций сложения и умножения получим функцию

$$f_3(x_1, x_2, x) = \mu^{k-1}x_1 + \mu^{k-1}x_2 + \tilde{\mu}x + \tilde{\mu}_0, \mu^{k-1} = \mu_3, \mu_3(0) = 1,$$

т.к. из [4] нам известно, что

$$(a+b)^p = a^p + b^p$$
$$a^{k-1} = 1, a^k = a^{p^m} = a, \forall a \in E_k.$$

Теперь, не более чем за $O(p^t)$ операций сложения и умножения, подставляя саму в себя функцию f_3 тем же способом, что и функцию f_2, получим из функции f_3 функцию $f_4(x_1, x_2, x) = \mu_3^{p^t}x_1 + \mu_3^{p^t}x_2 + \tilde{\mu}x + \tilde{\mu_0}$. Выпишем дроби $\mu_3^p, \mu_3^{p^t}$:

$$\mu_3^p = (1 + \xi\mu_4)^p = 1^p + (\xi\mu_4)^p = 1 + a'_p\xi^p + a'_{p+1}\xi^{p+1} + ...$$
$$\mu_3^{p^t} = (1 + \xi\mu_4)^{p^t} = 1^{p^t} + (\xi\mu_4)^{p^t} = 1 + a'_{p^t}\xi^{p^t} + a'_{p^t+1}\xi^{p^t+1} + ...$$

Число t мы будем выбирать так, чтобы $\tau \leq p^t$, заметим, что при $\tau \geq 2$ мы можем выбрать t такой, что $t \leq \tau$, а значит общее количество операций сложения и умножения для получения f_4 мы можем оценить как $O(p + \tau)$. Обозначим $\mu_5 = \mu_3^{p^t}$, тогда перепишем $f_4 = \mu_5x_1 + \mu_5x_2 + \mu'_5x + \mu_{05}$. Теперь будем подставлять f_4 в себя p-раз следующим образом:

$$f_4(x_1, f_4(x_2, f_4(x_3, ..., x), x), x), x) =$$
$$= \mu_5x_1 + \mu_5^2x_2 + ...\mu_5^px_p + \mu_5^px_{p+1} + (\mu'_5 + \mu'_5\mu + ... + \mu'_5\mu_5^{p-1})x +$$
$$+(\mu''_0 + \mu''_0\mu + ... + \mu''_0\mu_5^{p-1}) = F(x_1, x_2, ...x_{p+1}, x).$$
$$\hat{\mu}' = \mu''_0 + \mu''_0\mu + ... + \mu''_0\mu_5^{p-1},$$
$$\hat{\mu}'' = \mu'_5 + \mu'_5\mu + ... + \mu'_5\mu_5^{p-1}.$$

Вид μ_5 мы уже обсуждали ранее, все дроби $\mu_5^i, i = 1, ...p$ будут τ-эквивалентны единице. $\hat{\mu}''$,$\hat{\mu}'$ в свою очередь τ-эквивалентны нулю, т.к. в $\hat{\mu}'$ и в $\hat{\mu}''$ содержится p слагаемых. Следовательно, сама функция $F(x_1, x_2, ...x_{p+1}, x)$ будет τ-эквивалентна функции $x_1 + x_2 + x_3.... + x_{p+1}$, причем функцию F мы получили за $O(k+\tau)$ операций сложения и умножения. □

Сформулируем ещё одну вспомогательную лемму:

Лемма 6. *Пусть множество $M \subseteq L_k$, M не содержится целиком ни в одном из множеств $\Theta \in J_k^A$. Пусть также n - максимальная арность функций из M. Пусть также нами были получены константа $\gamma, \gamma(0) = b, b \in E_k$, а также функция $x_1+x_2+x_3....+x_{p+1}$. Тогда для любых $a \in E_k$ за $O(n)$ операций мы можем получить функцию $g_a(x) = \mu_a x + \gamma'$, где $\mu_a \in E'_k(\xi), \gamma' \in E'_k(\xi)$, $\mu_a(0) = a$.*

Доказательство. Так как множество M не содержится целиком в классах $P_l, \forall l \in \{1, ..., s\}$, то $\exists \mu_{l_1}, ..., \mu'_{l_m} \in U(M)$, $\mu'_i(0) \notin E_{k,l_i}$, где $E_{k,i}$ - подполя в E_k такие, что элемент $\mu_{l_1}(0)\mu_{l_2}(0)...\mu'_{l_m}(0)$ порождает E_k по операциям сложения и умножения. Для получения этого элемента мы потратим не более $O(k)$ операций сложения и умножения. Заметим, что так как $\mu_{l_1}(0)\mu_{l_2}(0)...\mu'_{l_m}(0)$ порождает все E_k, то мы можем получить все различные элементы E_k в нулевой момент времени за $O(k)$ операций сложения и умножения. Будем далее считать, что мы получили данные элементы и добавим $O(k)$ в общую оценку сложности в теореме 2 и не будем учитывать эту оценку в данной лемме.
Значит, $\forall a \in E_k \ \exists f(x_1, ..., x_{n'}) \in K(M)$, $\exists \mu \in U(f), \mu(0) = a$. Не ограничивая общности, будем считать, что μ - дробь перед первой переменной x_1. Подставив в f константу γ, полученную ранее, получим $f(x, \gamma, ..., \gamma) = \mu x + \gamma'$ за $O(n)$ операций сложения. □

И теперь докажем нужную нам лемму:

Лемма 7. *Пусть множество $M \subseteq L_k$, M не содержится целиком ни в одном из множеств $\Theta \in J_k^A$. Пусть также n - максимальная арность функций из M. Пусть также нами были получены константы $\gamma_a, \gamma_a(0) = a; \gamma_c, \gamma_c(0) = c; a, c \in E_k, a \neq c$, а также функция $x_1+x_2+x_3....+x_{p+1}$. Тогда мы можем получить константу $\gamma_0 \in K(M)$ такую, что $\gamma_0(0) = 0$ за $O(n+p)$ операций.*

Доказательство. Рассмотрим функцию $x_1 + x_2 + ... + x_{p+1}$. Подставим на место переменных $x_1, ...x_i$ функцию $g_b(\gamma_c)$, которую мы можем получить по лемме 6 за $O(n)$ операций сложения и умножения, на место переменных $x_{i+1}, ..., x_p$, а на место переменной x_{p+1} константу γ_c всего за $O(n+p)$ операций сложения и умножения. В итоге получим $g_b(\gamma_c) + ... + g_b(\gamma_c) + g_a(\gamma_a) + ... + \gamma_c = \gamma$. Теперь найдем такие значения i, b, что $\gamma(0) = 0$.

$$\gamma(0) = i\mu_b(0)\gamma_c(0) + (p-i)\mu_b(0)\gamma_a(0) + \gamma_c(0) + p\gamma'_c(0) = 0,$$

$$ib\gamma_c(0) + (p-i)b\gamma_a(0) + \gamma_c(0) = 0,$$

$$ib(\gamma_c(0) - \gamma_a(0)) = \gamma_c(0).$$

Получаем уравнение

$$i = \frac{\gamma_c(0)}{b(\gamma_a(0) - \gamma_c(0))}.$$

Если $\gamma_c(0) = 0$, то $i = 0$ и мы получаем необходимую нам костанту. Если $\gamma_c(0) \neq 0$, то тогда

$$b = \frac{\gamma_c(0)}{(\gamma_a(0) - \gamma_c(0))}, i = 1.$$

Количество операций для получения константы γ_0 составило $O(n + p)$. □

Лемма 8. *Пусть множество $M \subseteq L_k$, M не содержится целиком ни в одном из множеств $\Theta \in J_a^k$. Пусть также n - максимальная арность функций из M. Пусть также нами были получены константа $\gamma_0(0) = 0 \in E_k$, а также функция $x_1+x_2+x_3....+x_{p+1}$. Тогда для любых $a \in E_k$ за $O(n+p)$ операций сложения и умножения мы можем получить $\mu_a x+\gamma_0$, где $\mu_a \in E_k'(\xi), \gamma \in E_k'(\xi)$, $\mu_a(0) = a, \gamma_0(0) = 0$.*

Доказательство. Будем считать, что мы уже получили функцию $\mu_a+\gamma'$, $a \in E_k$ за $O(n)$ операций. Возьмем функцию сложения $x_1 + x_2 + ... + x_{p+1}$ и подставим на место первой переменной x_1 функцию $\mu_a x + \gamma'$, на место переменных $x_2, ...x_p$ подставим константу $\mu_a\gamma_0+\gamma'$, а на место переменной x_{p+1} константу γ_0:

$$(\mu_a x + \gamma) + \sum_{i=1}^{p-1}(\mu_a\gamma_0 + \gamma) + \gamma_0 =$$

$$= \mu_a x + \sum_{i=1}^{p-1}(\mu_a\gamma_0) + \gamma_0 = \mu_a x + \gamma', \gamma'(0) = 0.$$

Таким образом мы получили необходимую нам функцию $\mu_a x + \gamma_0$, используя не более чем $O(n + p)$ операций сложения и умножения. □

Теперь мы можем сформулировать нашу теорему:

Теорема 2. *Пусть множество $M \subseteq L_k$, M не содержится целиком ни в одном из множеств $\Theta \in J_k^A$. Пусть также n - максимальная арность функций из M. Тогда за $O((\tau + 1)(n + p + 1) + k + p)$ операций сложения и умножения, а также одну операцию обратной связи мы можем получить функции, τ-эквивалентные базисным функциям $\{1, x_1 + x_2, ax, \xi x | a \in E_k\}$.*

Доказательство. По лемме 4 мы научились получать функции, τ эквивалентные константам γ_a, γ_b за $O(n+k)$ операций сложения и умножения. По лемме 5 получим функцию, τ-эквивалентную функции $x_1 + x_2 + ... + x_{p+1}$ за $O(k+n+\tau)$ операций сложения и умножения. По лемме 7 получим функцию $\gamma_0, \gamma_0(0) = 0$ за $O(n+p)$ операций сложения и умножения, итого получим оценку в $O(n+k+\tau+p)$ операций сложения и умножения. Заметим, что оценка $O(k)$, полученная в лемме 6 не влияет на итоговую оценку.

Теперь будем получать задержку с нулевым начальным состоянием. Существует функция $g \in M \setminus M_1$, $g(x_1, ...x_n) = \sum_{i=1}^{n} \mu_i x_i + \mu_0$. Подставим во все переменные функции, кроме x_1, полученную ранее константу γ_0. Функция g примет вид

$$g(x, \gamma_0, ..., \gamma_0) = \mu_1 x + \gamma' = h(x).$$

$$\mu_1 = a(0) + a(1)\xi + a(2)\xi^2 + ..., \text{ где } a(1) \neq 0,$$

После возведения в степень k дробь μ_1 при разложении в ряд будет выглядеть следующим образом:

$$\mu_1^k = \mu_1^{p^m} = a^{p^m}(0) + a^{p^m}(1)\xi^{p^m} + a^{p^m}(2)\xi^{2p^m} + ...$$

Заметим что так как $a^{p^m} = a$, то μ_1^k будет принимать вид

$$\mu_1^k = \mu_1^{p^m} = a(0) + a(1)\xi^{p^m} + a(2)\xi^{2p^m} + ...$$

Подставим теперь полученную выше функцию $h(x)$ и $h^k(x)$, которую мы получим за $O(k)$ операций в сумматор $x_1 + ...x_{p+1}$ следующим образом:

$$h(x) + h^k(x) + ... + h^k(x) + \gamma_r' = h'(x).$$

Константу γ_r' мы получим по лемме 8 из функции $g_r(x)$, подставляя в $g_r(x)$ константу γ_0 за $O(n+p)$ операций сложения умножения. Значение r подберем так, чтобы свободный член на нулевом такте в функции $h'(x)$ был равен нулю. Для получения самой функции $h(x)$ мы потратим не более $O(n)$ операций сложения и умножения. При подстановке в сумму - не более $O(p)$. Таким образом на данном этапе мы потратим не более $O(n+k+p)$ операций сложения и умножения.

Выпишем функцию $h'(x)$ в следующем виде:

$$h'(x) = (a(1)\xi + \xi^2\hat{\mu})x + \hat{\gamma}, \hat{\gamma}(0) = 0.$$

По доказанной ранее лемме 8, мы можем получить функцию $g'(x) = \tilde{\mu}x + \gamma_0, \tilde{\mu}(0) = (a(1))^{-1}$ за $O(n+p)$ операций сложения и умножения.

Подставим её в $h'(x)$ за $O(1)$ операций сложения и умножения:

$$h'(g'(x)) = (\xi + \xi^2\mu'')x + \gamma'', \gamma''(0) = 0.$$

Обозначим $f_1(x) = h'(g'(x))$. Предположим теперь, что мы уже получили

$$f_\tau(x) = (\xi^\tau + \xi^{\tau+1}\mu_\tau)x + \gamma_\tau, \gamma_\tau = \xi^\tau\gamma'_\tau.$$

Пусть γ_τ в моменты времени $0, ..., \tau - 1$ равна нулю, а в момент времени τ константа $\gamma_\tau(\tau) = c$. По лемме 8 получим функцию $g_{-c}(x)$ за $O(n + p)$ операций сложения и умножения. Подставим в $g_{-c}(x)$ константу γ_0 и получим константу γ_{-c} такую, что $\gamma_{-c}(0) = -c$.

Подставим γ_{-c} в $f_\tau(x)$ и получим, что в моменты времени $0, ..., \tau$ функция $f_\tau(\gamma_c)$ равна нулю. Таким образом мы получили функцию $\tau + 1$-эквивалентную константе ноль. Подставив эту константу в функцию $x_1 + ...x_{p+1}$ на место переменных $x_3, ..., x_{p+1}$ получим функцию $\tau + 1$ эквивалентную сумматору $x_1 + x_2$.

Подставим теперь f_τ в $f_1(x)$. Получим функцию

$$f_\tau(f_1(x)) = (\xi^{\tau+1} + \xi^{\tau+2}\hat{\hat{\mu}})x + \hat{\hat{\gamma}}, \hat{\hat{\gamma}}(i) = 0, i \in \{0, ..., \tau - 1\}, \hat{\hat{\gamma}}(\tau) = d.$$

По лемме 8 получим функцию $g_u(x)$ за $O(n+p)$ операций сложения и умножения. Значение u найдем в ходе дальнейших вычислений. Подставим в $g_u(x)$ константу γ_0 и получим константу γ_u такую, что $\gamma_u(0) = u$. Сложим теперь $f_\tau(f_1(x))$ и $f_\tau(\gamma_u))$ и получим

$$f_\tau(f_1(x)) + f_\tau(f_1(\gamma_{-d})) = f_{\tau+1}(x).$$

$$f_{\tau+1}(x) = (\xi^{\tau+1} + \xi^{\tau+2}\hat{\hat{\mu}})x + \xi^\tau(d + c + u) + \gamma''', \gamma'''(i) = 0, i = \{0, ..., \tau\}.$$

$$u + c + d = 0, \text{ следовательно, } u = -c - d.$$

Мы получили сумматор от двух переменных и константу ноль за $O(\tau(n+p))$ операций сложения и умножения. Отсюда также видно, что на каждом такте мы можем получать константу с произвольным количеством нулей в начале и произвольным значением $u \in E_k$ за $O(n + p)$ операций сложения и умножения. Перейдем к получению задержки с нулевым начальным состоянием.

Выпишем

$$h_1(x) = (\xi + \xi^2\mu'') + \gamma'', \gamma''(0) = 0, \gamma'' = b_{1,1}\xi + b_{2,1}\xi^2 + ...$$

Прибавим к функции $h_1(x)$ константу $\gamma_{1,-b_{1,1}}$ такую, что $\gamma_{1,-b_{1,1}}(0) = 0$, $\gamma_{1,-b_{1,1}}(1) = -b_{1,1}$. Данным сложением мы обнулим значение при ξ в константе γ''.

$$h'_1(x) = (\xi + \xi^2\mu'') + \gamma'' + \gamma_{1,-b_{1,1}} = (\xi + \xi^2\mu'')x + \gamma'_1, \gamma'_1(0) = \gamma'_1(1) = 0.$$

Аналогично будем действовать с коэффициентом при степени ξ^2 в константе γ_1'. Заметим, что на уничтожение коэффициента при произвольной степени ξ в константе мы потратим $O(n+p)$ операций сложения и умножения. Теперь обнулим коэффициент в μ'' при степени ξ^2. Пусть $\mu''(0) = a'$. Тогда подставим в функцию $h_2(x)$ функцию $g_2(x) = \tilde{\mu}_2 x + \gamma_{g,2}, \tilde{\mu}_2(0) = (a')^{-1}$, полученную за $O(n+p)$ операций сложения и умножения. Затем сложим функии $h_2(g_2(x))$ и $h_1'(x)$:

$$h_2(g_2(x)) + h'(x) = \xi + \xi^3\mu_3 + \gamma_{2,0}, \gamma_{2,0}(0) = \gamma_{2,0}(1) = 0.$$

$$h_2(g_2(x)) + h_1'(x) \overset{3}{\sim} \xi x.$$

Действуя аналогично для остальных итераций, мы получим функцию τ-эквивалентную задержке с нулевым начальным состоянием за $O(\tau(n+p))$ операций сложения и умножения.

Теперь будем получать умножитель на произвольную константу. Возьмем полученный ранее по лемме 8 за $O(n+p)$ операций сложения и умножения умножитель на некоторую константу $a(0)$ и возведем его в степень k:

$$(a(0) + \xi^k\mu^k)x + \gamma, \gamma(0) = 0.$$

Эта функция может быть получена за $O(k)$ операций сложения и умножения. Подставим константу γ в задержку и получим константу γ'. Значения константы γ для тактов $1, 2....$ нам необходимо занулить до момента времени τ. Выпишем константу γ' в форме ряда:

$$\xi(\gamma(1)....) = 0\gamma(1)... = \gamma'.$$

Далее просуммируем:

$$\gamma + \gamma'(p-1) = 00\gamma''(2)...$$

Таким способом мы можем обнулить все значения γ до такта τ, используя $O(\tau p)$ операций сложения и умножения. Данным способом за τ тактов получим выражение

$$(a(0) + \xi^\tau\mu^\tau)x + \xi^\tau\gamma_\tau,$$

которое, в свою очередь, при $k \geq \tau$ будет τ-эквивалентно умножению x на константу $a(0)$. Если же $\tau > k$, будем выражение $((a_0 + \xi^k\mu^k)x + \gamma)^{k^i}$ возводить в степень k^i, где $k^i > \tau$.

Посчитаем теперь итоговое количество операций сложения и умножения. В леммах 4,5,6,7 нам потребовалось $O(p+k+n+\tau)$ операций сложения и умножения, а также одна операция обратной связи. Для получения сумматора, задежки и константы ноль мы использовали $O(\tau(n+p))$

операций сложения и умножения. Получение умножителя на константу имеет сложность $O(k+n+p\tau)$. Итого, за $O((\tau+1)(n+p+1)+k+p)$ операций сложения и умножения, а также одну операцию обратной связи мы получили функции τ-эквивалентные базису $\{x_1 + x_2, ax, \xi x | a \in E_k\}$. $\square$

Заключение.

Мы получили оценки на получение A-эквивалентного элементарного базиса для случаев E_2 и E_k. Мы получили куда более скромную оценку на количество операций, чем для случая K-замыкания над полем E_2 для линейных автоматов, сохраняющих нулевую последовательность, где оценка являлась экспоненциальной. В следующих работах представляется интересным получить нижнюю оценку на количество операций для получения A-эквивалентного элементарного базиса и проверить, можно ли улучшить верхнюю оценку.

Автор выражает благодарность своему научному руководителю А.А. Часовских.

Список литературы

[1] Часовских А. А., "О полноте в классе линейных автоматов", *Математические вопросы кибернетики*, 1991, №2, 140–166.

[2] Kudryavtsev V.B., Alyoshin S.V., Podkolzin A.S., *Introduction to automata theory*, «Science», Moscow, 1985, 320 с.

[3] Ильин И.Ю., *Интеллектуальные системы. Теория и приложения*, 2022, 164–173.

[4] Лидл Р., Нидеррайтер Г., *Конечные поля*, Мир, Москва, 1988, 430 с.

[5] В.А.Буевич, "Об алгоритмической неразрешимости распознавания A-полноты для ограниченно-детерминированных функций", *Матем. заметки*, **11**:6 (1972), 687–697.

Comlexity of implementation of A-closure elementary basis in lineary automata class on finite field

Ilin I.Y.

In the previous work we have received compexity estimation for elementary basis implementation in the class of linear automata that preserves zero-sequence. In the current work we will find the complexity estimation for elementary basis realization by A-closure operations in linear automata under finite field [4].

Keywords: linear automata, A-closure, A-expressibility, algorithm complexity.

References

[1] Chasovskikh A.A., "Completeness problem for the class of linear automata functions", *Discrete Math. Appl.*, 1991, № 2, 140–166.

[2] Kudryavtsev V.B., Alyoshin S.V., Podkolzin A.S., *Introduction to automata theory*, «Science», Moscow, 1985, 320 c.

[3] Ilin I.Y., "Comlexity of implementation of elementary basis in one-place lineary automata class that preserves zero sequence", *Intelligent Systems. Theory and applications.*, 2022, 164–173.

[4] R. Lidl, H. Niederreiter, *Finite Fields*, Mir, Moscow, 1988, 430 c.

[5] V. A. Buevich, "On algorithmic solvability of the A-completeness problem for systems of boundedly determinate functions containing all one-place boundedly determinate S-functions", *Math. notes.*, **11**:6 (1972), 687–697.

К сведению авторов публикаций в журнале «Интеллектуальные системы. Теория и приложения»

В соответствии с требованиями ВАК РФ к изданиям, входящим в перечень ведущих рецензируемых научных журналов и изданий, в которых могут быть опубликованы основные научные результаты диссертаций на соискание ученой степени доктора и кандидата наук, статьи в журнал «Интеллектуальные системы. Теория и приложения» предоставляются авторами в следующей форме:

1. Статьи, набранные в пакете LaTeX, предоставляются к загрузке через WEB-форму http://intsysjournal.org/generator_form .
2. К статье прилагаются файлы, содержащие название статьи на русском и английском языках, аннотацию на русском и английском языках (не более 50 слов), список ключевых слов на русском и английском языках (не более 20 слов), информация об авторах: Ф.И.О. полностью, место работы, должность, ученая степень и/или звание (если имеется), контактные телефоны (с кодом города и страны), e-mail, почтовый адрес с индексом города (домашний или служебный).
3. Список литературы оформляется в едином формате, установленном системой Российского индекса научного цитирования.
4. За публикацию статей в журнале «Интеллектуальные системы. Теория и приложения» с авторов (в том числе аспирантов высших учебных заведений) статей, рекомендованных к публикации, плата не взимается. Оттиски статей авторам не предоставляются. Журнал распространяется по подписке, экземпляры журнала рассылаются подписчикам наложенным платежом. Условия подписки публикуются в каталоге НТИ «Роспечать», индекс журнала 64559.
5. Доступ к электронной версии последнего вышедшего номера осуществляется через НЭБ «Российский индекс научного цитирования». Номера, вышедшие ранее, размещаются на сайте http://intsysjournal.org, и доступ к ним бесплатный. Там же будут размещены аннотации всех публикуемых статей.

Подписано в печать: 19.06.2023
Дата выхода: 30.06.2023
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).