ISSN 1818-1015
eISSN 2313-5417

2023 Volume 30 No 4

MODELING AND ANALYSIS OF INFORMATION SYSTEMS

SCIENTIFIC JOURNAL

Start date of publication — 1999
Published quarterly

FOUNDER
P.G. Demidov Yaroslavl State University

EDITORIAL OFFICE
14 Sovetskaya str., Yaroslavl 150003, Russian Federation

Website: http://mais-journal.ru
E-mail: mais@uniyar.ac.ru
Phone: +7 (4852) 79-77-73

2023 Том 30 № 4

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ

НАУЧНЫЙ ЖУРНАЛ

Издается с 1999 года
Выходит 4 раза в год

УЧРЕДИТЕЛЬ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Ярославский государственный университет им. П. Г. Демидова»

РЕДАКЦИЯ
ул. Советская, 14, Ярославль, 150003, Российская Федерация
Website: http://mais-journal.ru
E-mail: mais@uniyar.ac.ru
Телефон: +7 (4852) 79-77-73

Свидетельство о регистрации СМИ ПИ № ФС77-66186 от 20.06.2016 выдано Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций. Подписной индекс в каталоге «Урал-Пресс» – 31907. Технический редактор, компьютерная вёрстка – К. В. Лагутина. Подписано в печать 04.12.2023. Дата выхода в свет 20.12.2023. Формат 200×265 мм. Объем 146 с. Тираж 32 экз. Свободная цена. Заказ 23173. Издатель и его адрес: Ярославский государственный университет им. П. Г. Демидова; ул. Советская, 14, Ярославль, 150003, Россия. Типография и ее адрес: ООО «Филигрань»; ул. Свободы, 91, Ярославль, 150049, Россия.
Содержание предназначено для детей старше 12 лет.

Contents

Содержание

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 4, 2023
journal homepage: www.mais-journal.ru

DISCRETE MATHEMATICS IN RELATION TO COMPUTER SCIENCE

Algorithm for link prediction in self-regulating network with adaptive topology based on graph theory and machine learning

E. Y. Pavlenko[1]

DOI: 10.18255/1818-1015-2023-4-288-307

[1]Peter the Great St. Petersburg Polytechnic University, 29 Polytechnicheskaya str., St. Petersburg 195251, Russia.

MSC2020: 93B70; 68R10
Research article
Full text in Russian

Received August 7, 2023
After revision October 24, 2023
Accepted November 2, 2023

The paper presents a graph model of the functioning of a network with adaptive topology, where the network nodes represent the vertices of the graph, and data exchange between the nodes is represented as edges. The dynamic nature of network interaction complicates the solution of the task of monitoring and controlling the functioning of a network with adaptive topology, which must be performed to ensure guaranteed correct network interaction. The importance of solving such a problem is justified by the creation of modern information and cyber-physical systems, which are based on networks with adaptive topology. The dynamic nature of links between nodes, on the one hand, allows to provide self-regulation of the network, on the other hand, significantly complicates the control over the network operation due to the impossibility of identifying a single pattern of network interaction.

On the basis of the developed model of network functioning with adaptive topology, a graph algorithm for link prediction is proposed, which is extended to the case of peer-to-peer networks. The algorithm is based on significant parameters of network nodes, characterizing both their physical characteristics (signal level, battery charge) and their characteristics as objects of network interaction (characteristics of centrality of graph nodes). Correctness and adequacy of the developed algorithm is confirmed by experimental results on modeling of a peer-to-peer network with adaptive topology and its self-regulation at removal of various nodes.

Keywords: modeling; networks with adaptive topology; graph model; link prediction; centrality metrics

INFORMATION ABOUT THE AUTHORS

Evgeny Y. Pavlenko
corresponding author

orcid.org/0000-0003-1345-1874. E-mail: pavlenko@ibks.spbstu.ru
Associate Professor, Ph.D. in Technical Sciences.

Funding: The research is funded by the Russian Science Foundation, project no. 22-21-20008. The research is funded by the grant of the St. Petersburg Science Foundation in accordance with the agreement of April 15, 2022 № 61/220.

For citation: E. Y. Pavlenko, "Algorithm for link prediction in self-regulating network with adaptive topology based on graph theory and machine learning", *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 288-307, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 4, 2023
сайт журнала: www.mais-journal.ru

DISCRETE MATHEMATICS IN RELATION TO COMPUTER SCIENCE

Алгоритм предсказания связей в саморегулирующейся сети с адаптивной топологией на базе теории графов и машинного обучения

Е. Ю. Павленко[1]

DOI: 10.18255/1818-1015-2023-4-288-307

[1]Санкт-Петербургский политехнический университет Петра Великого, ул. Политехническая, д. 29, г. Санкт-Петербург, 195251 Россия.

УДК 519.17
Научная статья
Полный текст на русском языке

Получена 7 августа 2023 г.
После доработки 24 октября 2023 г.
Принята к публикации 2 ноября 2023 г.

В статье представлена графовая модель функционирования сети с адаптивной топологией, где узлы сети представляют собой вершины графа, а обмен данными между узлами представлен в виде ребер. Динамический характер сетевого взаимодействия осложняет решение задачи мониторинга и контроля функционирования сети с адаптивной топологией, которую необходимо выполнять для обеспечения гарантированно корректного сетевого взаимодействия. Значимость решения такой задачи обосновывается созданием современных информационных и киберфизических систем, в основе которых лежат сети с адаптивной топологией. Динамический характер связей между узлами, с одной стороны, позволяет обеспечивать саморегуляцию сети, с другой стороны, существенно осложняет контроль за работой сети в связи с невозможностью выделения единого шаблона сетевого взаимодействия.

На базе разработанной модели функционирования сети с адаптивной топологией предложен графовый алгоритм предсказания связей, распространенный на случай с одноранговыми сетями. В основу алгоритма положены значимые параметры узлов сети, харатеризующие как их физические характеристики (уровень сигнала, заряд батареи), так и их характеристики как объектов сетевого взаимодействия (характеристики центральности вершин графа). Корректность и адекватность разработанного алгоритма подтверждена экспериментальными результатами по моделированию одноранговой сети с адаптивной топологией и ее саморегуляции при удалении различных узлов.

Ключевые слова: моделирование; сети с адаптивной топологией; графовая модель; предсказание связей; метрики центральности

ИНФОРМАЦИЯ ОБ АВТОРАХ

Евгений Юрьевич Павленко
автор для корреспонденции

orcid.org/0000-0003-1345-1874. E-mail: pavlenko@ibks.spbstu.ru
доцент, кандидат технических наук.

Финансирование: Исследование выполнено за счет гранта Российского научного фонда № 22-21-20008. Исследование выполнено за счет гранта Санкт-Петербургского научного фонда в соответствии с соглашением от 15 апреля 2022 г. № 61/220.

Для цитирования: E. Y. Pavlenko, “Algorithm for link prediction in self-regulating network with adaptive topology based on graph theory and machine learning”, *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 288-307, 2023.

Введение

Развитие информационных и сетевых технологий, цифровизация различных отраслей деятельности человека привели к трансформации информационных систем, расширив их состав интеллектуальными устройствами физического мира и изменив принципы их сетевого взаимодействия за счет появления беспроводных самоорганизующихся сетей (ad hoc сети, сети MANET и VANET) [1, 2].

Цифровые информационные системы, в состав которых входит большое число интеллектуальных устройств, взаимодействующих друг с другом и окружающей средой посредством сенсорных и сетевых технологий, далее будем называть киберфизическими системами (КФС). Ключевым отличием КФС от традиционных информационных систем является соединение необратимых физических и обратимых информационных процессов в рамках единой системы. При этом ведущая роль в реализации процессов остается за интеллектуальными устройствами, а влияние человека на работу КФС сведено к минимуму.

Сетевая топология КФС может варьироваться от классической клиент-серверной до динамической, с переменным составом узлов и связей между ними, а также с возможностью перемещения узлов в пространстве. К преимуществам динамической сетевой инфраструктуры следует отнести возможность саморегуляции структуры таких сетей, возможную за счет адаптивного перестроения структуры сети в случае появления/исчезновения участников сетевого взаимодействия и разрыва сетевых соединений. Это весомый фактор, в частности, для КФС, предоставляющих различные сервисы населению — медицинскую помощь, навигацию людей в опасных регионах с ограниченным числом аварийных выходов, различные транспортные сервисы [3—7].

Беспроводные сети с адаптивной топологией можно разделить на следующие типы [1]:

1. Беспроводные децентрализованные самоорганизующиеся сети мобильных устройств (mobile ad hoc network, MANET).
2. Автомобильные самоорганизующиеся сети (vehicular ad hoc network, VANET). Используются для связи между транспортными средствами.
3. Беспроводные сети ad hoc, использующие смартфоны (smartphone ad hoc network, SPAN). После внедрения технологии сетей ad hoc группа смартфонов, находящихся в непосредственной близости друг от друга, может совместно создать сеть ad hoc.
4. Беспроводные ячеистые сети (wireless mesh network, mesh-сеть) — коммуникационная сеть, состоящая из радиоузлов, организованных в ячеистой топологии. Каждый узел в mesh-сети работает как маршрутизатор и как хост.
5. Военные тактические сети (army tactical MENT) — используется военными для связи «на ходу». Такая беспроводная тактическая сеть ad hoc полагается на дальность действия и мгновенную работу для организации сетевого взаимодействия в случае необходимости.
6. Беспроводные сенсорные сети (wireless sensor network, WSN). Датчики в беспроводных сенсорных сетях обычно представляют собой небольшие сетевые узлы с очень ограниченной вычислительной мощностью, ограниченной коммуникационной способностью и ограниченным источником питания. Таким образом, датчик может выполнять только простые вычисления и общаться с датчиками и другими узлами на небольшом расстоянии.
7. Сети ad hoc для спасения при стихийных бедствиях (disaster rescue ad hoc network). Такие ad hoc сети используются, когда случается катастрофа и установленное коммуникационное оборудование не функционирует должным образом.

Для эффективной саморегуляции сетевой инфраструктуры, при которой обеспечивается доставка данных от всех узлов сети к центру обработки данных и стабильная работа системы, построенной на базе динамической сети, необходимо выявить принципы функционирования самой сети

и объединить их со спецификой, накладываемой видом системы. Это позволит адаптировать уже заложенные архитектурно возможности саморегуляции системы к ее целевой функции, которая будет значительно варьироваться в зависимости от типа системы.

Так, для КФС мониторинга труднодоступных регионов (например, лесов, гор, мест землетрясений) целевая функция будет заключаться в качественном получении и доставке данных от всех (или максимально возможного числа) сенсоров, то есть, при разрыве связи между парой узлов необходимо, чтобы данные от узлов не потерялись [8]. В этом случае первостепенное значение имеет связность сети, и саморегуляция должна быть направлена на ее обеспечение.

Для КФС другого типа, например, промышленных объектов, ключевое значение может иметь непрекращающееся выполнение какого-либо процесса, пусть даже с потерей качества в связи с выходом из строя части узлов-участников сетевого взаимодействия. В таком случае саморегуляция должна быть направлена на выполнение этого процесса, что может быть выражено как последовательный сетевой обмен между узлами определенного типа в заданном порядке.

Целью настоящей статьи является создание алгоритма предсказания связей, работающего на базе разработанной ранее автором графовой модели функционирования сетей с адаптивной топологией [9]. В рамках модели изменчивая топология таких сетей рассматривается в динамике за счет представления состояний сети в виде «снимков» динамического графа. Значимым преимуществом данной модели среди подобных является ее способность описывать различные типы сетей с адаптивной топологией и ее ориентированность на сети, являющиеся базой для КФС, за счет единовременного учета физических и информационных характеристик узлов сети.

Разработанный алгоритм предсказания связей использует параметры, которыми оперирует модель, и выполняет предсказание появления или исчезновения ребер в «снимке» динамического графа в следующий момент времени. Область применения данного алгоритма уже, чем у модели — он ориентирован только на работу одноранговых сенсорных сетей как наиболее простого типа сетей с адаптивной топологией. Исходя из результатов экспериментальных исследований, подтвердивших его эффективность и соответствие поведению сетей с адаптивной топологией на практике, в рамках дальнейших исследований целесообразно расширить данный алгоритм, распространив его на случаи с гетерогенными сенсорными и промышленными сетями.

Статья организована следующим образом:

- в главе 1 представлен обзор связанных работ, цель которого — сравнить разработанный алгоритм с аналогами и описать в целом область применения алгоритмов предсказания связей;
- в главе 2 приводится краткое описание разработанной ранее автором графовой модели, подкрепленное практическим примером для случая с гетерогенной сенсорной сетью;
- глава 3 содержит формальную постановку задачи разработки алгоритма предсказания связей в одноранговых сенсорных сетях, включающая определение целевой функции для формирования условий к запуску алгоритма, требования к алгоритму и используемые им параметры;
- в главе 4 представлено описание работы алгоритма в виде псевдокода;
- глава 5 содержит описание проведенных экспериментальных исследований.

1. Обзор связанных работ в части алгоритмов предсказания связей

Алгоритмы предсказания связей в зарубежной литературе носят название link prediction. Области применения таких алгоритмов включают предсказание неизвестных белковых взаимодействий, прогнозирование реакции на лекарственные препараты [10], рекомендации продуктов пользователям [11], заполнение графов знаний [12]. Также они нашли широкое применение в графовых методах анализа взаимосвязей между пользователями в социальных сетях [13, 14].

Работа таких алгоритмов весьма разнообразна, однако наиболее распространенный подход, реализуемый ими, состоит в оценке схожести вершин графа в соответствии с некоторым множеством

заданных характеристик. Так, для социальных сетей к таким метрикам могут относиться: число общих друзей, одно и то же место учебы или работы в сочетании с похожим возрастом, схожие интересы и т. п.

Однако следует отметить, что подходы, основанные на сходстве, имеют некоторые проблемы, связанные с потерей информации об узлах и обобщающей способностью индексов сходства.

Работы, посвященные предсказанию связей в различных сетях, как правило, используют такие графовые характеристики как метрики центральности, вес пути между узлами и распределение степеней графа [15—18]. Однако часто вычисление метрик центральности используется для определения наиболее «влиятельных» узлов в сети, что не требуется для предсказания связей в одноранговых сенсорных сетях, где все узлы равноправны и одинаковы по характеристикам.

Многие алгоритмы link prediction реализуются с использованием искусственных нейронных сетей или алгоритмов машинного обучения, и следует отметить, что это направление перспективно за счет возможности более эффективной обработки больших объемов данных и выявления неявных зависимостей. Однако для всех методов и алгоритмов, использующих техники искусственного интеллекта, важны два аспекта:

- сбалансированная и репрезентативная обучающая выборка, сформированная таким образом, чтобы в полной мере отражать специфику данных исследуемой предметной области;
- грамотный выбор признаков объектов, используемых для обучения.

Как правило, наборы обучающих данных для КФС и сетей с адаптивной топологией (сенсорные сети, MANET и VANET) отражают либо физические аспекты функционирования системы (показатели мощности сигнала, заряда устройств, координаты их перемещения), либо сетевые (число отправленных и полученных сетевых пакетов, используемый протокол маршрутизации и т. п.). Крайне малое число наборов данных включает информацию о состоянии сетей с адаптивной топологией, вследствие чего исследования, посвященные решению задач мониторинга и анализа безопасности таких сетей, используют достаточно давно опубликованные наборы данных, такие как KDD-99 [19, 20].

Относительно грамотного выбора признаков объектов, в ряде публикаций наблюдается стремление авторов переложить задачу выявления значимых признаков объектов на искусственный интеллект. Так, авторы работы [21] утверждают, что в силу отсутствия знаний о некоторых эвристиках, используемых для предсказания связей, целесообразнее получить эвристики, характерные для каждой отдельно взятой сети, с использованием графовых нейронных сетей. Такой подход не подходит для предсказания связей в сетях, на базе которых функционируют КФС, поскольку для таких систем особенную значимость имеют как физические характеристики узлов сети, так и информационные характеристики сетевого обмена.

Однако часть публикаций посвящена абстрактным исследованиям сложных сетевых структур, в которых признаки графа не описываются явно, и оставляют возможность для других исследователей по включению в модель признаков различной природы. К таким исследованиям относится работа [22], авторы которой предлагают использовать графовые нейронные сети для предсказания связей, уделяя при этом особенное внимание подграфам, моделирующим отдельные части сети. Такое направление является перспективным для промышленных сетей, для которых с точки зрения графовой модели характерно наличие нескольких компонент связности.

Применительно к КФС и сенсорным сетям, алгоритмы предсказания связей часто используются для прогнозирования маршрутов в сетях, предсказания сбоев и решения подобных проблем, связанных с надежностью. Так, авторы работы [23] также используют link prediction для прогнозирования качества каналов связи как одной из важнейших задач маршрутизации в сложных сетях.

В работе [24] авторы прогнозируют состояния каналов для сетей подводных акустических датчиков. Они отмечают, что акустический канал значительно меняется с течением времени, однако есть ряд физических характеристик, по которым можно оценить состояние канала. В рамках алгоритма предсказания связей авторы используют такие параметры как интенсивность сигнала, интенсивность шума и расстояние между узлами. Таким образом, графовые метрики здесь не учитываются, набор характеристик ограничен физическими параметрами. В основе алгоритма лежит получение краткосрочного прогноза с использованием фильтра Калмана, что говорит также о значительной сложности настройки параметров фильтра для конкретной исследуемой сети.

Исследование авторов работы [25] посвящено прогнозированию связей в мобильных ad hoc сетях (MANET), решение этой задачи существенно осложняется подвижностью узлов в пространстве. Для предсказания связей авторы предлагают алгоритм, учитывающий длительность связей между узлами, топологию сети и корреляции между узлами. По их мнению, это позволяет получить прогноз с учетом пространственно-временных характеристик узлов. Однако авторы не учитывают физические параметры мобильных узлов, что может значительно повлиять на результат прогноза. Так, атака физического подавления сигнала в определенном радиусе исключит из сети сразу несколько узлов, а устройство с разряженной батареей в любой момент может отключиться и перестать принимать и передавать данные.

Отличительной чертой предложенного в данной работе алгоритма является одновременный учет им как физических параметров устройств, составляющих сеть, так и тех параметров, которые оказывают большое влияние на информационные процессы. Эти параметры выражаются посредством графовых метрик центральности.

Значимость данного алгоритма с практической точки зрения обоснована его соответствием поведению сетей с адаптивной топологией, которые способны к саморегуляции. Кроме того, условия начала работы алгоритма тесно связаны с целевой функцией той системы, основой для работы которой является исследуемая сеть. Следует отметить, что разработанный алгоритм предназначен для единовременного учета физических, информационных и функциональных свойств сети, что отличает его от аналогичных алгоритмов предсказания связей.

2. Графовая модель функционирования сетей с адаптивной топологией

Модель функционирования сетей с адаптивной топологией базируется на динамической теории графов [26]. На взгляд автора, для решения задач мониторинга и контроля состояния сети с адаптивной топологией в первую очередь необходимо иметь возможность наблюдения за сетью — ее структурой и характеристиками узлов и ребер — в динамике, что позволит сравнивать состояния сети в различные моменты времени. Динамический граф удобно рассматривать как дискретную последовательность статических графов, изучая свойства каждого «снимка» графа с использованием методов, ориентированных на статические графы.

Подробное описание процесса разработки модели представлено в более ранней статье автора [9]. К значимым параметрам, которые следует учесть в модели, относятся как характеристики вершин и ребер графа, так и характеристики графа в целом (например, число компонент связности).

2.1. Описание модели

Сеть с адаптивной топологией представляется в виде ориентированного графа $G = (V, E)$, где $V = v_1, v_2, \ldots, v_n$ — множество узлов сети, $E = e_1, e_2, \ldots, e_m$ — множество дуг, $E \subseteq (VV)$.

Начальное состояние сети обозначается G^0, а некоторое состояние сети в момент времени p обозначается G^p.

DG — динамический граф, представляющий собой набор (серию) «снимков» графа G, $DG = (S^0, S^1, \ldots)$, где S^{step} — «снимок» графа. Любой «снимок» графа S^{step} включает атрибуты:

- предыдущее состояние сети, выраженное в виде графа $G^{t_{step}}$. На первом шаге t_{step} принимает значение t_0, «снимок» графа на нулевом шаге представляет собой изначальное состояние исследуемой сети;
- набор выполненных преобразований Op^{step};
- временная метка t_n.

Таким образом, $S^{step} = (G^{t_n}, Op^{step}, t_n)$.

1. Op^{step} — кортеж, характеризующий изменения, совершенные при преобразовании предыдущего «снимка» к текущему; $Op = (action, V, E)$, где $action$ — набор булевых признаков, демонстрирующий, какие унарные операции над графом применяются на данном шаге, V — вершины, к которым были применены операции $action$, E — дуги, к которым были применены операции $action$; $action = (VA, VD, EA, ED)$, где:
   - VA — операция добавления вершины, $VA = 0$, если вершина не была добавлена, $VA = 1$, если вершина была добавлена).
   - VD — операция удаления вершины, $VD = 0$, если вершина не была удалена, $VD = 1$, если вершина была удалена).
   - EA — операция добавления дуги, $EA = 0$, если дуга не была добавлена, $EA = 1$, если дуга была добавлена).
   - ED — операция удаления дуги, $ED = 0$, если дуга не была удалена, $ED = 1$, если дуга была удалена).

   Результатом применения каждой из операций к вершине или паре вершин является бинарный ответ: 0 или 1, в зависимости от того, было выполнено преобразование или нет.

   При преобразовании предыдущего «снимка» графа каждая унарная операция ассоциируется с временной меткой. В ходе практической реализации модели предполагается использование базы данных, в которую будут сохраняться сведения о конфигурациях графа и внесенных изменениях. В базе данных также предполагается хранение пар совершенных унарных операций и соответствующих каждой из них временных меток. Такая упорядоченность во времени обеспечивает согласованность операций и не позволяет, например, добавить ребро к удаленной вершине.

   В описании модели предлагается сосредоточиться именно на характере внесенных изменений, в связи с чем на первый план выходит множество типов совершенных преобразований при переходе к новому «снимку».
2. «Снимок» графа G^{step} описывается как общими структурными характеристиками графа Str, так и характеристиками отдельных вершин ($Vrtx$) и дуг (Ed), $G^{step} = (Str, Vrtx, Ed)$.
   2.1 Структурные характеристики: $Str = (Cn, D, |V|, |E|)$, где:
      - Cn — связность графа, определяет число компонент сильной связности num;
      - D — наибольший диаметр компонент связности, $D = d(v_a, v_b)$, где v_a и v_b — наиболее удаленные друг от друга вершины графа;
      - $|V|$ — число вершин графа;
      - $|E|$ — число дуг графа.

   2.2 Характеристики отдельных вершин $Vrtx$. На практике множество вершин, характеристики которых будут храниться в памяти системы и анализироваться, может варьироваться от всего множества вершин графа, моделирующего сеть, до множества некоторых критических вершин, характеризующих наиболее значимые узлы сети. Зададим множество вершин $V^c \subseteq V$, данные о которых будут сохраняться и отслеживаться при практической реализации модели. Для этого множества вершин V^c в модели определены следующие характеристики: $Vrtx = (Cent, Pow)$, где:

- *Cent* — значения метрик центральности для множества вершин V^c. Таким образом, для каждой вершины $v_i \in V^c$ множество *Cent* представляет собой вектор значений $Cent^i = (Cent_1, Cent_2, Cent_3)$, где $Cent_1$ — степень вершины, $Cent_2$ — степень посредничества, $Cent_3$ — степень близости;
- *Pow* — мощность вершин из множества V^c, характеризующих узлы моделируемой сети с точки зрения их энергоемкости.

2.3 Характеристики отдельных дуг *Ed*. По аналогии с вершинами, выделим некоторое множество дуг E^c, характеристики которых будут постоянно контролироваться при дальнейшем анализе функционирования системы. Для них в модели определены следующие характеристики: $Ed = (W, TS)$, где:

- W — вес дуги, характеризующий качество сетевого соединения. Как было отмечено ранее, практическая интерпретация данной характеристики может варьироваться в зависимости от типа системы;
- TS — множество временных рядов для этой дуги за время, прошедшее с момента предыдущего «снимка» сети.

2.2. Пример описания реальной сети в терминах модели

Рассмотрим применение модели к реальной сети с адаптивной топологией на примере беспроводной гетерогенной сети из источника [27]. Зафиксируем ее структуру в момент времени t_0 (рисунок 1).

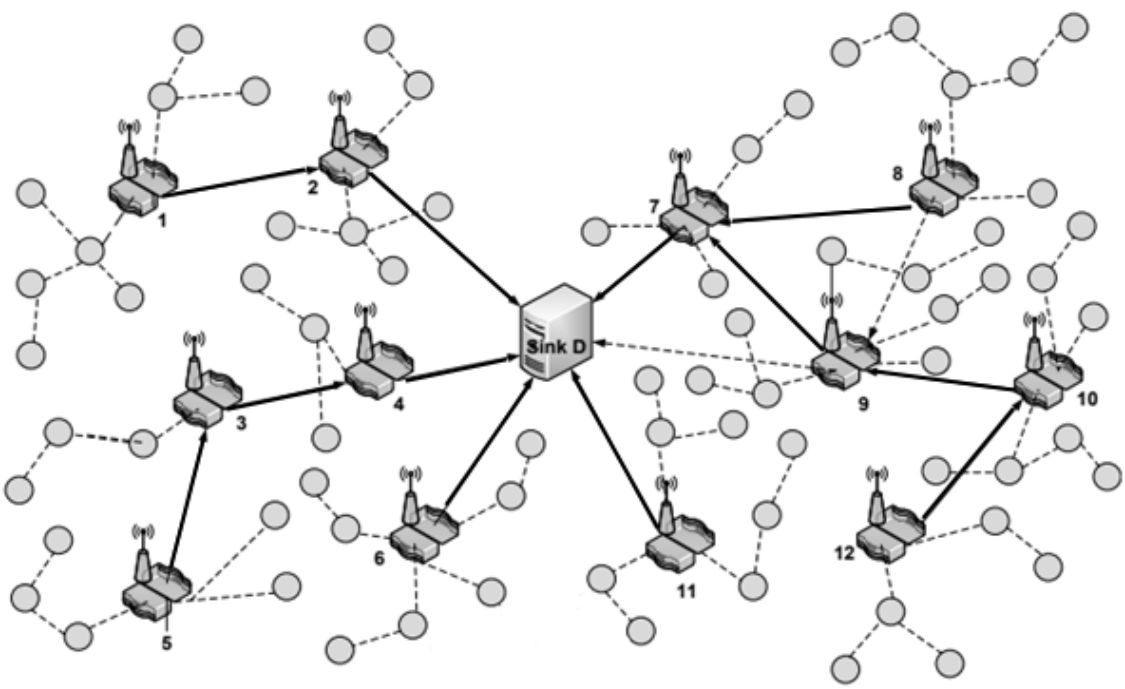


Fig. 1. Sensor network structure at time t_0

Рис. 1. Структура сенсорной сети в момент времени t_0

Узлы с номерами от 1 до 12 представляют собой так называемые «супер-узлы», их мощность сигнала выше, чем у обычных сенсоров, обозначенных на рисунке серыми кругами. За счет этого они могут агрегировать показания от множества сенсоров и передавать их на узел *D* (*sink D* на ри-

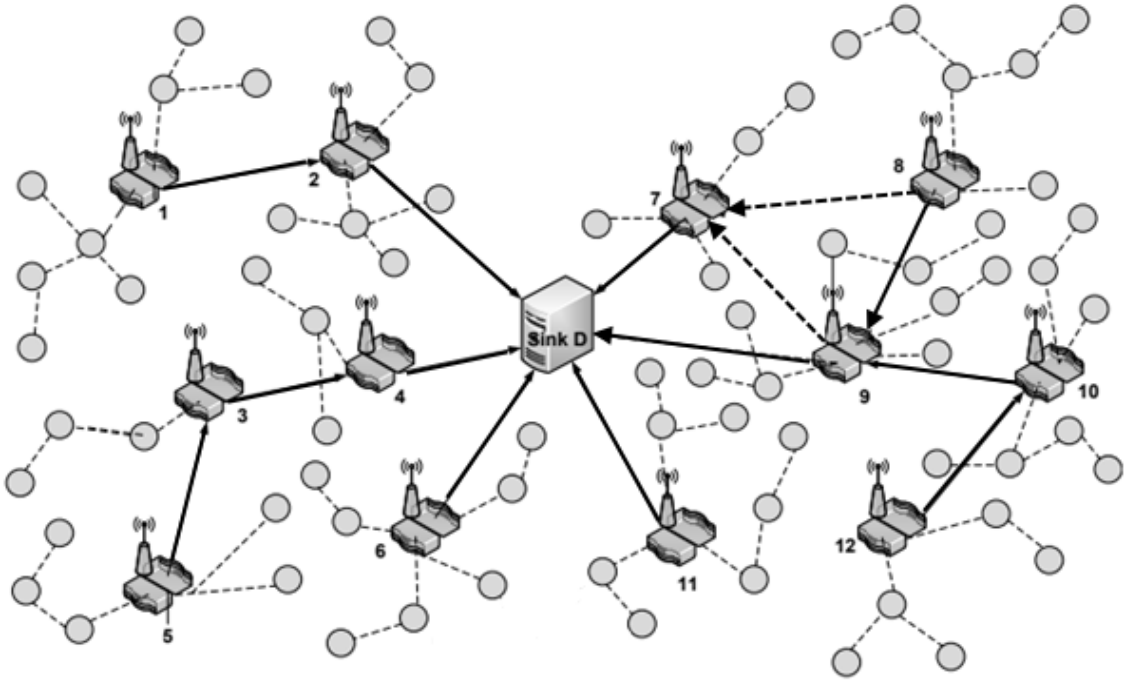
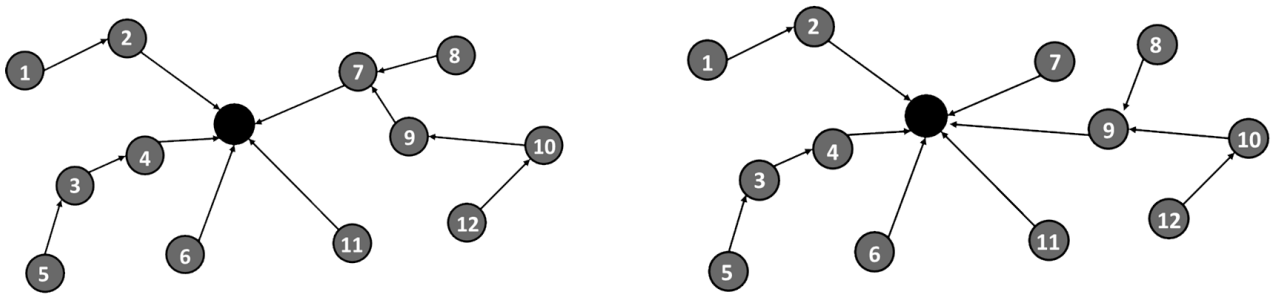


Fig. 2. Sensor network structure at time t_1

Рис. 2. Структура сенсорной сети в момент времени t_1



a) b)

Fig. 3. Snapshots of dynamic graph

Рис. 3. «Снимки» динамического графа

сунке). Этот узел с точки зрения графа является стоком, физически он представляет собой сервер или точку доступа.

Связи между супер-узлами и сервером могут быть двух типов: прямой линией обозначена активная связь, в соответствии с которой идет обмен данными, пунктирной линией обозначены альтернативные пути обмена данными. Исследуем эту сеть в динамике, активируя альтернативные пути и получая «снимок» графа G^1 в момент времени t_1. Полученный результат представлен на рисунке 2.

Представим эти два состояния сети в виде двух «снимков» динамического графа, для удобства не визуализируя узлы-сенсоры (рисунок 3). Тогда в терминах модели, $DG = (S^0, S^1)$. Рассмотрим «снимок» $S^1 : S^1 = (G^1, Op^0, t_1)$. Здесь из четырех возможных унарных операций над статическим графом G^0 выполнены только две: добавление и удаление дуг. Распишем Op^0:

$$Op^0 = ((VA, VD, EA, ED), \{\}, \{\{(v_8, v_9), (v_9, v_D)\}, \{(v_9, v_7), (v_8, v_7)\}\});$$
$$Op^0 = ((0, 0, 1, 1), \{\}, \{\{(v_8, v_9), (v_9, v_D)\}, \{(v_9, v_7), (v_8, v_7)\}\}).$$

Характеристики графов: $Str(G^0) = ((1, 84), 12, 84, 81)$, $Str(G^1) = ((1, 84), 11, 84, 81)$, видим, что изменился диаметр графа, несмотря на то, что число ребер в графе осталось прежним.

Значимыми вершинами в данном случае будут все 12 супер-узлов, и для них будут меняться только показатели центральности, поскольку мощность *Pow* в данном случае одинакова и постоянна. Так, изменились степени входа вершин 7 и 9: для «снимка» графа G^0: $deg(v_7) = 5$, $deg(v_9) = 4$, для «снимка» графа G^1 степени входа: $deg(v_7) = 3$, $deg(v_9) = 5$, степени выхода остались прежними.

Следует отметить, что альтернативные пути обмена данными между супер-узлами (пунктирная линия) не учитывались при вычислении степеней, учитывались только связи с сенсорами и активные связи (прямая линия) между супер-узлами.

Веса дуг в данном примере предполагаются одинаковыми, а анализ временных рядов, характеризующих обмен данными между узлами сети, выходит за рамки данной статьи и здесь не рассматривается.

Таким образом, представленная модель, базирующаяся на математическом аппарате динамической теории графов, описывает функционирование адаптивной сетевой топологии и предоставляет основу для разработки методов и алгоритмов мониторинга и контроля состояния системы. К таким алгоритмам и относится разработанный алгоритм предсказания связей, детально описанный в следующей главе.

3. Постановка задачи разработки алгоритма предсказания связей в одноранговых сенсорных сетях

Для создания алгоритма предсказания связей между узлами сети с учетом специфики, накладываемой способностью сети к саморегуляции, необходимо:

1. Задать условия, при возникновении которых в сети должен быть запущен процесс саморегуляции. Эти условия будут исходить из нарушений в целевой функции системы, определение которой дано далее.
2. Выбрать параметры, которые на практике будут наиболее значимыми при выполнении сетью саморегуляции.
3. Разработать алгоритм предсказания связей, работа которого инициируется при возникновении хотя бы одного условия из п. 1, и базирующийся на выбранных в п. 2 параметрах.

3.1. Целевая функция системы

Поскольку рассматриваемые сети за счет своей адаптивной топологии способны к саморегуляции, этот процесс будет направлен на реализацию целевой функции системы. Определим понятие целевой функции (ЦФ) системы, исходя из чего сформулируем условия для выполнения саморегуляции.

Сеть с адаптивной топологией рассматривается как инструмент реализации ЦФ той КФС, которая построена на ее основе. Предполагается, что любая КФС обладает ЦФ, выражаемой множеством процессов, которые обязательно реализует рассматриваемая система. При этом каждый такой процесс характеризуется определенными диапазонами значений для характеристик узлов сети и процессов передачи данных.

Для различных типов сетей с адаптивной топологией ЦФ будет варьироваться. Если рассматривать только сети, узлы которых не меняют своего положения в пространстве, их можно разделить на 3 группы:

1. Одноранговые сенсорные сети — включают множество однотипных маломощных датчиков (сенсоров), выполняющих функции измерения, приема и передачи данных.
2. Гетерогенные сенсорные сети — помимо маломощных датчиков, включают так называемые «суперузлы», обладающие мощным сигналом и способностью к агрегации и обработке дан-

ных, поступающих от множества сенсоров. Рассмотренная в п. 1.2 сеть относится к данному типу.

3. Промышленные сети — обладают выраженной иерархической структурой, включают несколько типов устройств, варьирующихся по своему функционалу и сложности. Возможно перераспределение нагрузки между более сложными узлами (концентраторами, хабами и т. д.).

Представленное исследование касается саморегуляции только одноранговой сетевой топологии, выполняющей простейшую ЦФ — сбор и передачу данных от всех сенсоров от одного конца сети до другого, подключенного к системе хранения данных. Далее будем считать, что число вершин n в графе не меняется. Также будем рассматривать граф как неориентированный, в связи с тем, что каждый узел выполняет одинаковые функции и способен как принимать, так и отправлять данные, возможность этого определяется только нахождением в заданном радиусе с соседними узлами.

Ребро e_{ij} между парой вершин v_i, v_j существует только в том случае, если они удалены друг от друга на расстояние $d(v_i, v_j)$, не большее r, $d(v_i, v_j) = \sqrt{((x_i - x_j)^2 + (y_i - y_j)^2)}$ $\forall v_i, v_j \in V$, (x_i, y_i), $i = \overline{(1, N)}$, где $(x_i; y_i)$ и $(x_j; y_j)$ — координаты вершин v_i и v_j в двумерном евклидовом пространстве, соответственно.

Такая ЦФ характерна для сенсорных сетей, расположенных в труднодоступных местах, где ключевая задача — собрать информацию и передать ее в хранилище данных для дальнейшего анализа [28, 29].

Выразим ЦФ в виде набора следующих условий, при возникновении которых в сети должен быть запущен процесс саморегуляции:

1. Граф G является связным: $\nexists v_i : deg(v_i) = 0$.
2. Все вершины графа функционально эквивалентны и выполняют одинаковые функции $F = measure, send, receive$ — измерение, отправку (измеренного значения или значений, полученных от соседних узлов) и прием измеренных значений от соседних узлов, находящихся на расстоянии r : $\forall i : f(v_i) = (1, 1, 1)$. Здесь f — оператор, сопоставляющий каждой вершине графа двоичный вектор, соответствующий операциям множества F: 1 — если вершина выполняет данную функцию, 0 — в противном случае.
3. Распределение степеней графа G, моделирующего сетевую структуру рассматриваемой КФС, близко к равномерному. Такое предположение выдвинуто из-за предполагаемого равномерного покрытия сенсорами всей заданной области и во избежание перегрузки некоторых узлов, что характерно в случае с реализацией сетевых атак. Например, атаки отказа в обслуживании и атаки типа «черная дыра» (sinkhole) характеризуются увеличением степени вершин графа. Для этого зададим максимальную степень вершин графа $deg_{max}(v_i), i = \overline{(1, n)}$.

Триггером для выполнения саморегуляции сети является нарушение хотя бы одного из вышеописанных условий.

Важно отметить, что вышеописанные условия являются обязательными для саморегуляции, однако в зависимости от типа КФС, саморегуляция может быть выполнена и при наличии дополнительных факторов, таких как снижение заряда батареи устройства или уровня его сигнала. Представленный алгоритм описан только для трех ключевых условий.

3.2. Формальная постановка задачи

Задача состоит в разработке алгоритма *Pred* предсказания связей между узлами неориентированного графа, узлы которого функционально эквивалентны и выполняют одинаковый набор функций.

Определим условия для запуска алгоритма *Pred*:

1. $\exists v_i : deg(v_i) = 0$.

2. $\exists v_i : f(v_i) \neq (1,1,1)$.
3. $\nexists v_k : deg(v_k) = deg_{max}(v_i)$.

На вход алгоритм *Pred* принимает следующие параметры:

- текущий «снимок» динамического графа S^{step};
- множество *Vrtx* значимых параметров вершин графа, моделирующего сеть;
- максимальную степень вершин графа $deg_{max}(v_i)$;
- радиус установления связи между узлами r;
- пороговое значение *threshold* для принятия решения о добавлении/удалении ребра.

Разрабатываемый алгоритм должен быть таким, что спрогнозированный «снимок» динамического графа на шаге $step + 1$ максимально соответствует реальному «снимку» графа на шаге $step + 1$. Отклонение от предсказания вычисляется на основе симметрической разности множеств ребер для спрогнозированного и реального «снимков», результатом которой является множество, состоящее только из тех ребер, которые входят ровно в одно из множеств. Мощность множества и представляет собой отклонение от предсказания.

$$Pred : Pred(S^{step}, Vrtx, deg_{max}(v_i), r) \rightarrow {S_{pred}}^{step+1}, {S_{pred}}^{step+1} : |{S_{pred}}^{step+1} - S^{step+1}| = 0.$$

Учитывая, что рассматривается случай с постоянным числом узлов в сенсорной сети, разрабатываемый алгоритм *Pred* должен предсказывать только появление или исчезновение ребер в графе. Целью настоящего исследования является разработка такого алгоритма *Pred*, чтобы он выполнял эту функцию на основе вычисления вероятностей наличия ребра между заданной парой вершин.

Требования, накладываемые на алгоритм:

1. Алгоритм работает с неориентированным графом.
2. Для повышения скорости работы алгоритма, он не реализует пересчет вероятностей для ребер, инцидентных вершинам, которые не подпадают под условия саморегуляции.
   2.1 В случае с появлением изолированной вершины, алгоритм вычисляет вероятности появления ребер между этой вершиной и ее ближайшими соседями, и добавляет в порождаемый им граф ${S_{pred}}^{step+1}$ ребро, для которого вероятность была наибольшей.
   2.2 В случае, когда вершина v_i графа не выполняет одну или несколько из своих функций, алгоритм *Pred* работает для множества вершин, смежных с v_i. Так, при проблемах с приемом и передачей данных от соседних узлов, саморегуляция сети будет заключаться в построении альтернативных маршрутов, обеспечивающих передачу данных в нужном направлении.
   2.3 В случае с возникновением в сети «перегруженных» узлов, саморегуляция сети будет заключаться в попытке приблизить распределение степеней графа к равномерному за счет удаления ребер, инцидентных «загруженной» вершине v_i и создании новых ребер для некоторого подмножества вершин, смежных с v_i. Поэтому алгоритм *Pred* будет работать для некоторого случайно выбранного подмножества вершин, инцидентных v_i.
3. Для работы алгоритма требуется обученная модель логистической регрессии. Задача предсказания связей трактуется как задача бинарной классификации, где в одном классе собраны ребра, «рекомендованные» алгоритмом к построению в будущем графе, а во втором классе — ребра, которых, по прогнозу алгоритма, не будет в будущем графе. Выбор логистической регрессии обоснован ее эффективностью и удобством с точки зрения генерации значений вероятности принадлежности объекта к классу.

Выбор параметров, которые на практике будут наиболее значимыми при выполнении сетью саморегуляции, зависит от типа сети. В случае с сенсорной сетью, узлы которой неподвижны в пространстве, значимыми предполагается следующее множество характеристик вершин *Vrtx*:

- мощность сигнала устройства сети, Pow_{v_i};
- заряд батареи узла Bat_{v_i};
- возможности узла по обмену данными, выражаемые значением метрики центральности по степени (степень вершины) ${Cent_1}^{v_i}$;
- критичность узла, выражаемая метрикой центральности по посредничеству ${Cent_2}^{v_i}$;
- пороговое значение *threshold* для принятия решения о внесении изменений в матрицу смежности графа.

Следует учитывать, что адаптивная сетевая топология является основой для функционирования некоторой КФС, а такие системы отличаются от информационных и компьютерных наличием взаимосвязанных информационной и физической составляющих. Поэтому, на взгляд автора, отнести к значимым параметрам вершин как физические характеристики узлов (мощность сигнала и заряд батареи), так и характеристики, важные для информационного обмена (метрики центральности).

4. Описание алгоритма в виде псевдокода

4.1. Появление изолированной вершины

Алгоритм вычисляет всех соседей изолированной вершины v_i, с использованием логистической регрессии определяет вероятность появления ребра между v_i и каждым ее соседом, выбирает ребро, вероятность появления которого максимальна, добавляет его к текущему графу и присваивает получившийся граф графу для следующего шага S^{step+1}.

START
READ входные параметры (S^{step}, $Vrtx$={Pow_{v_i}, Bat_{v_i}, ${Cent_1}^{v_i}$, ${Cent_2}^{v_i}$}, $deg_{max}(v_i)$, r, *threshold*)
COMPUTE множество V^x вершин, находящихся на расстоянии, не большем чем r, от v_i
FOR каждой вершины из множества V^x
    SAVE параметры *Pow*, *Bat*, $Cent_1$, $Cent_2$
    COMPUTE вероятность $P(e_{ij})$ появления ребра между вершиной v_i и текущей вершиной
COMPUTE ребро с максимальным значением вероятности
ADD ребро с максимальным значением вероятности к «снимку» графа S^{step}
SET $S^{step+1} \leftarrow S^{step}$
END

4.2. Нарушение в работе вершины

Для вершины v_i, функционирование которой нарушено, алгоритм определяет все вершины, смежные с ней, на основе матрицы смежности графа A. Для каждой такой вершины он определяет множество соседних вершин, вычисляет вероятности появления ребра между текущей вершиной и каждой из соседних. На основе выбранных максимальных значений изменяет матрицу смежности, для выбранных узлов меняя значения 0 и 1 на соответствующие вероятности. Затем алгоритм вычисляет разность между изначальной и измененной матрицами смежности, и если абсолютное значение разности превышает некоторый заданный пользователем порог *threshold*, меняет в матрице смежности значение на противоположное (0 вместо 1 и наоборот). На основе обновленной матрицы смежности строится граф, соответствующий состоянию сети на следующем шаге S^{step+1}.

START
READ входные параметры (S^{step}, $Vrtx$={Pow_{v_i}, Bat_{v_i}, ${Cent_1}^{v_i}$, ${Cent_2}^{v_i}$}, $deg_{max}(v_i)$, r, *threshold*)
INIT матрицу вероятностей PA появления ребер в графе, размером $n \times n$
COMPUTE множество V^y вершин, смежных с v_i
SET $PA \leftarrow A$, где A — матрица смежности графа
FOR каждой вершины из множества V^y

COMPUTE множество V^z вершин, находящихся на расстоянии, не большем чем r, от текущей вершины

COMPUTE вероятность $P(e_{yz})$ появления ребра между текущей вершиной и вершинами из множества V^z

COMPUTE ребро с максимальным значением вероятности для текущей вершины

SET $P(e_{yz}) \leftarrow$ значение вероятности, полученное на предыдущем шаге

COMPUTE $\Delta \leftarrow A - PA$

FOR каждого элемента матрицы смежности a_{kj}

IF $|\Delta_{kj}| > threshold$ THEN

SET $a_{kj} \leftarrow 1 - a_{kj}$

ENDIF

COMPUTE «снимок» графа S^{step+1} на основе полученной матрицы смежности A

END

4.3. Возникновение «перегруженной» вершины

Алгоритм работает аналогично случаю 2, с тем лишь отличием, что для «разгрузки» вершины v_i графа выбирается некоторое подмножество смежных с ней вершин, для которых в дальнейшем и идет перестроение.

START

READ входные параметры (S^{step}, $Vrtx$={Pow_{v_i}, Bat_{v_i}, ${Cent_1}^{v_i}$, ${Cent_2}^{v_i}$}, $deg_{max}(v_i)$, r, $threshold$)

COMPUTE множество V^y вершин, смежных с v_i

COMPUTE подмножество V^{y_1} множества V^y

INIT матрицу вероятностей PA появления ребер в графе, размером $n \times n$

SET $PA \leftarrow A$, где A — матрица смежности графа

FOR каждой вершины из множества V^{y_1}

COMPUTE множество V^z вершин, находящихся на расстоянии, не меньшем чем r, от текущей вершины

COMPUTE вероятность $P(e_{yz})$ появления ребра между текущей вершиной и вершинами из множества V^z

COMPUTE ребро с максимальным значением вероятности для текущей вершины

SET $P(e_{yz}) \leftarrow$ значение вероятности, полученное на предыдущем шаге

COMPUTE $\Delta \leftarrow A - PA$

FOR каждого элемента матрицы смежности a_{kj}

IF $|\Delta_{kj}| > threshold$ THEN

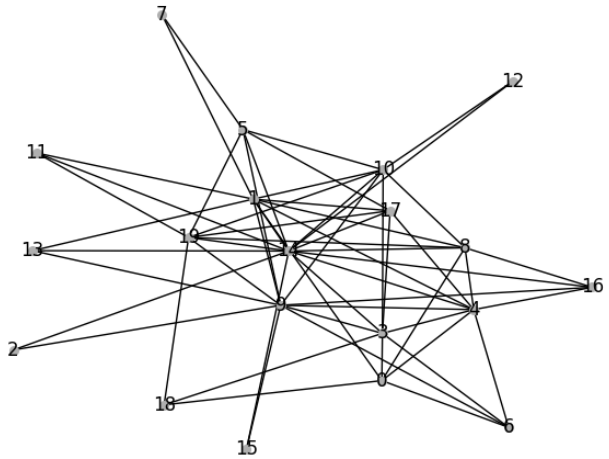
SET $a_{kj} \leftarrow 1 - a_{kj}$

ENDIF

COMPUTE «снимок» графа S^{step+1} на основе полученной матрицы смежности A

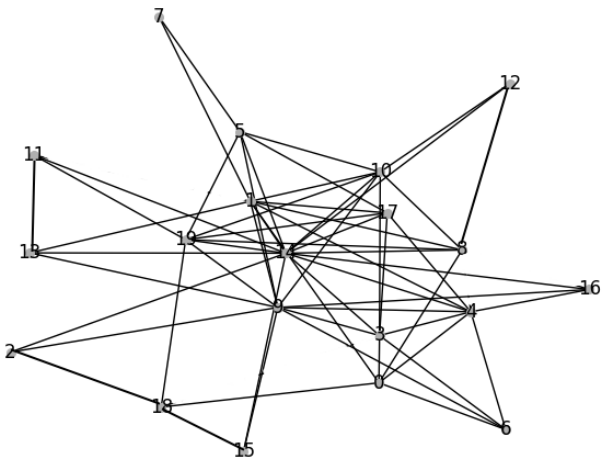
END

Алгоритм выбора подмножества V^{y_1} базируется на том, что после определения соседних вершин для каждой загруженной вершины выполняется ранжирование соседних вершин по двум параметрам: степень и уровень заряда батареи. В приоритете для разрыва соединения оказываются вершины с невысокой степенью и высоким уровнем заряда батареи. Такой выбор основан на том, что для таких вершин процесс переподключения пройдет с меньшими потерями с точки зрения качества работы сети, чем для вершин, которые имеют большое число связей и при этом не обладают запасом заряда батареи.



a)

Fig. 4. 2 snapshots of a dynamic graph



b)

Рис. 4. 2 «снимка» динамического графа

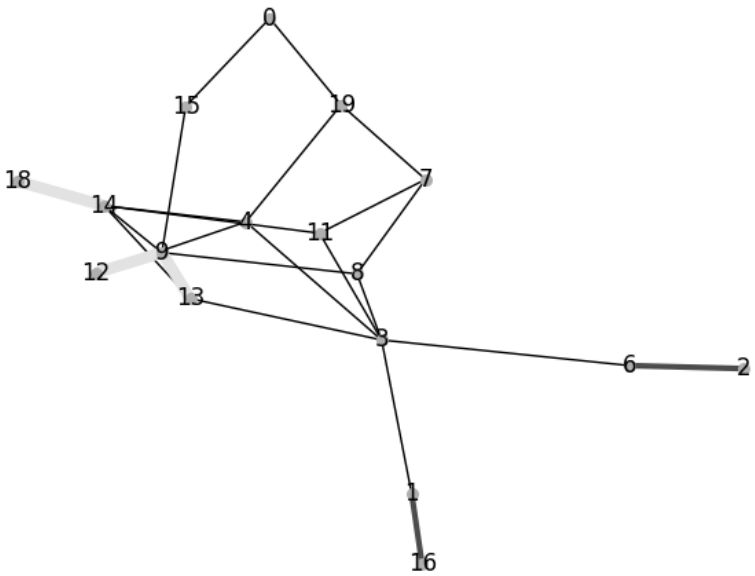


Fig. 5. Links prediction

Рис. 5. Предсказание связей в графе

5. Эксперименты по предсказанию связей в графе

Для проведения экспериментальных исследований в части оценки качества разработанного алгоритма на языке Python была смоделирована одноранговая сеть в виде неориентированного графа. Для визуального удобства размер сети составлял 20 узлов, пронумерованных от 0 до 19.

Смоделированная одноранговая сеть построена на базе разработанной модели функционирования сетей с адаптивной топологией. На рисунке 4 показаны 2 последовательных «снимка» a) и b) динамического графа, моделирующего одноранговую сеть.

Визуализация предсказанного разработанным алгоритмом графа представлена на рисунке 5. Жирные светлые линии обозначают появление связей в графе на следующем шаге, а жирные темные линии — исчезновение связей между вершинами на следующем шаге.

Стартом к запуску алгоритма предсказания ребер, помимо трех ключевых условий саморегуляции, являлись также:

- снижение уровня заряда батареи узла более чем на 5 %;
- ухудшение мощности сигнала больше чем на 10 %.

В условиях сети с маломощными сенсорами, на взгляд автора, показатель заряда батареи является более значимым для сохранения работоспособности сети, чем мощность сигнала, на которую могут влиять посторонние факторы. Исходя из этого, были заданы соответствующие пороговые значения для изменений: 5 % и 10 %, соответственно.

Модель логистической регрессии, прогнозирующая вероятность появления или исчезновения ребра на следующем шаге, была обучена на 10 000 «снимках» графа, показатель *AUC* для логистической регрессии составил 95,8 %. Подобранные моделью коэффициенты показали, что наиболее значимым признаком из всех является уровень заряда батареи устройства.

Для оценки качества работы алгоритма были смоделированы случаи появления изолированных и «загруженных» вершин. Пороговое значение *threshold* было выбрано эмпирически, *threshold* = 0.7.

5.1. Появление изолированных вершин

На графе было смоделировано появление изолированных вершин под номерами 2 и 8, при одновременном снижении заряда батареи у вершины 19 на 4 %.

Результаты работы алгоритма представлены на рисунке 6.

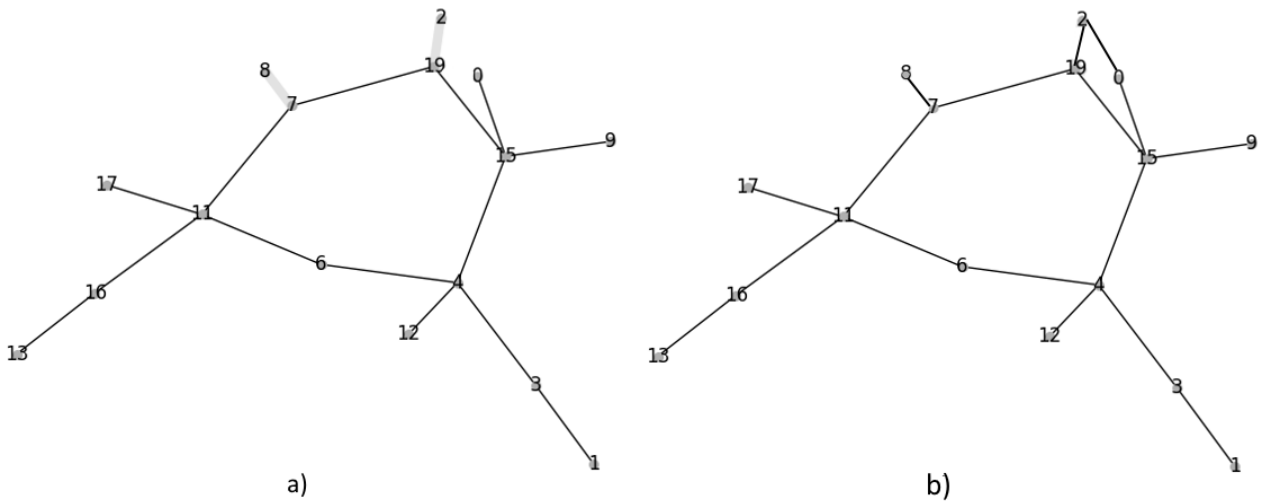


Fig. 6. Links prediction

Рис. 6. Предсказание связей в графе

Алгоритм работал следующим образом:

1. Определил ближайших соседей изолированных вершин:
   1.1 Для вершины 8: вершина 7.
   1.2 Для вершины 2: вершины 0 и 19, оценка расстояния r показала, что вершина 19 расположена чуть ближе к вершине 2.
2. Использовал обученную модель логистической регрессии для вычисления вероятностей появлений ребер (0, 2) и (0, 19) — они составили 0,98 и 0,87, соответственно.
3. Выбрал ребро с максимальным значением вероятности — между вершинами (0, 2).
4. Добавил к текущему «снимку» графа ребра (7, 8) и (0, 2).
5. Присвоил полученный на шаге 4 граф графу для следующего шага.

Из визуализации работы смоделированной сети, действительно, видно что оба спрогнозированных ребра появились на следующем шаге.

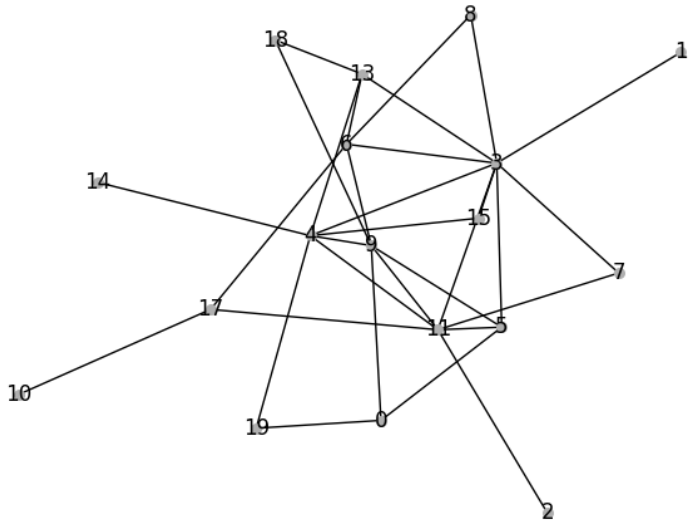


Fig. 7. Network with “overloaded” vertices

Рис. 7. Сеть с «перегруженными» вершинами

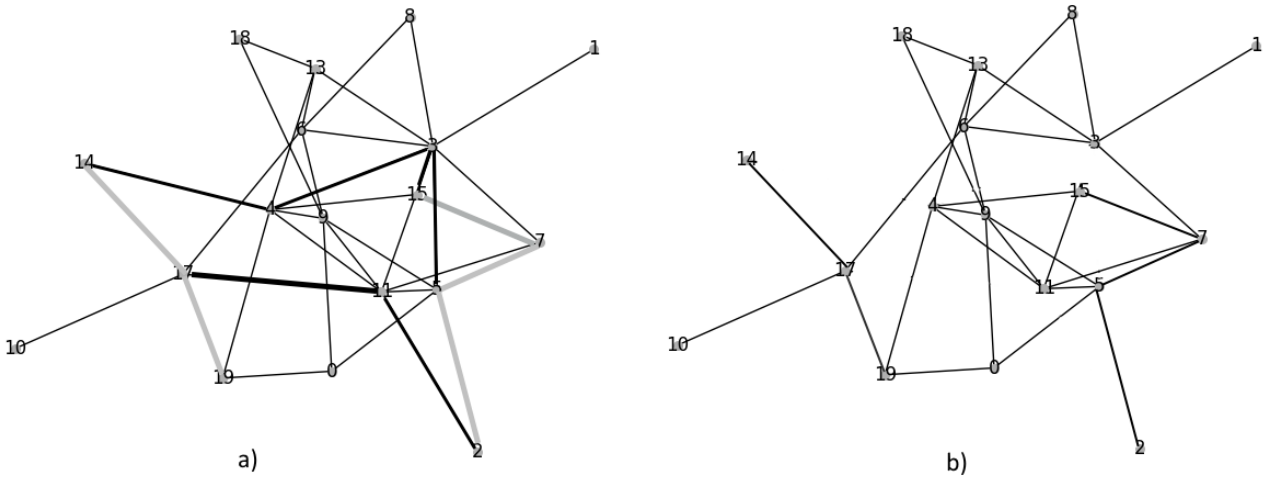


Fig. 8. Links prediction

Рис. 8. Предсказание связей в графе

5.2. Возникновение «загруженных» вершин

Было смоделировано появление вершин с более высокой степенью, чем у остальных. Это были вершины под номерами 3, 4 и 11. Изначально, $deg(v_3) = 8, deg(v_4) = 7, deg(v_{11}) = 7$. «Снимок» сети представлен на рисунке 7.

Результат работы алгоритма представлен на рисунке 8.

Алгоритм работал следующим образом:

1. Определил «загруженные вершины»: {3, 4, 11}.
2. Определил множество соседних вершин:
   - 2.1 Для вершины 3: {1, 5, 4, 6, 7, 8, 13, 15}.
   - 2.2 Для вершины 4: {3, 9, 11, 13, 14, 15, 19}.
   - 2.3 Для вершины 11: {2, 4, 5, 7, 9, 15, 17}.
3. Для каждой вершины из п.1 выбрал по 2 вершины, которые будут «переключены» на другие:
   - 3.1 Для вершины 3: {5, 15}.

3.2 Для вершины 4: {3, 14}.
3.3 Для вершины 11: {2, 17}.
4. Удалил ребра (3, 4), (3, 5), (3, 15), (4, 14), (2, 11), (11, 17).
5. Для каждой из множества вершин {2, 5, 14, 15}:
5.1 Определил ближайших соседей.
5.2 Использовал обученную модель логистической регрессии для вычисления вероятностей появлений ребер.
5.3 Выбрал ребро с максимальным значением вероятности, инцидентное текущей вершине.
6. Сохранил множество ребер, которые будут добавлены: (5, 7), (7, 15), (14, 17), (2, 5) и (17, 19).
7. Создал матрицу вероятностей путем копирования значений из матрицы смежности и обновления значений в ячейках, соответствующих вершинам, инцидентым ребрам, полученным на шаге 5.
8. Для множества ребер, полученных на шаге 5, вычислил разность матриц смежности и вероятностей.
9. Для ячеек матрицы, полученной на шаге 7, в которых абсолютное значение было меньше либо равно *threshold*:
9.1 Установил значения, противоположные значениям матрицы смежности.
9.2 Остальные значения установил в соответствии с матрицей смежности.
10. Сгенерировал граф состояния сети на следующем шаге по матрице, полученной на шаге 8.2.

Видно, что результаты работы алгоритма соответствуют поведению смоделированной сети. В результате саморегуляции, максимальная степень вершин составила 6 (у одной вершины под номером 9), и в распределении степеней вершин графа исчезли «всплески».

Таким образом, можно сделать вывод об адекватности разработанного алгоритма предсказания связей и о его соответствии реальному поведению узлов в самоорганизующейся сети.

Следует отметить, что данный алгоритм может быть незначительно доработан для предсказания связей в гетерогенных сенсорных и промышленных сетях. Тогда необходимо будет учитывать:

- типы узлов сети — они повлияют на возможность создания связей (например, типы узлов могут быть такими, что связь между узлами данных типов невозможна) и будут иметь разный приоритет при предсказании связей;
- контекст функций, выполняемых узлами — так, одно и то же ребро между узлами разных типов может иметь разное значение (отправка команды или передача данных);
- специфику, накладываемую ЦФ — это характерно для промышленных сетей, для них ЦФ может быть выражена как упорядоченная последовательность ребер, и сохранение упорядоченности также окажет влияние на результат прогноза.

Заключение

В работе представлена обобщенная графовая модель, описывающая функционирование беспроводных сетей с адаптивной топологией. Выбранный для моделирования математический аппарат динамической теории графов позволяет не только адекватно описать все изменения, происходящие в таких сетях, но и получить нужные сведения для контроля и мониторинга состояния сети.

На базе разработанной модели предложен алгоритм предсказания связей между узлами сети с адаптивной топологией, ограниченный случаем с одноранговыми сетями. Проведенные исследования связанных работ продемонстрировали, что современные исследования в основном не учитывают одновременно информационную и физическую природу киберфизических систем, и, следовательно, динамических сетей, лежащих в их основе.

Предложенный алгоритм устраняет этот недостаток, поскольку он в полной мере базируется на разработанной графовой модели и использует как физические (уровень сигнала и заряда ба-

тареи), так и информационно-структурные характеристики вершин одноранговой сети (метрики центральности по степени и посредничеству). В основе алгоритма лежит обученная автором модель логистической регрессии, показатель AUC которой составил 95,8 %, что говорит о довольно высоком качестве классификации. Визуализация работы алгоритма и последующее сравнение с полученным на следующем шаге состоянием сети продемонстрировали его соответствие принципам работы сенсорных сетей.

References

[1] V. Student and R. Dhir, "A study of ad hoc network: A review", *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 3, no. 3, pp. 135–138, 2013.

[2] V. Amulya, "Cyber physical systems by using wireless sensor networks", *International Journal of Science and Research*, vol. 7, no. 4, pp. 1380–1396, 2018.

[3] V. L. Narasimhan, A. A. Arvind, and K. Bever, "Greenhouse asset management using wireless sensor-actor networks", in *International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM'07)*, IEEE, 2007, pp. 9–14.

[4] T. Taleb, D. Bottazzi, M. Guizani, and H. Nait-Charif, "Angelah: A framework for assisting elders at home", *IEEE Journal on Selected Areas in Communications*, vol. 27, no. 4, pp. 480–494, 2009.

[5] P. Mohan, V. N. Padmanabhan, and R. Ramjee, "Nericell: Rich monitoring of road and traffic conditions using mobile smartphones", in *Proceedings of the 6th ACM conference on Embedded network sensor systems*, 2008, pp. 323–336.

[6] A. Thiagarajan *et al.*, "Vtrack: Accurate, energy-aware road traffic delay estimation using mobile phones", in *Proceedings of the 7th ACM conference on embedded networked sensor systems*, 2009, pp. 85–98.

[7] S. Mathur *et al.*, "Parknet: Drive-by sensing of road-side parking statistics", in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, 2010, pp. 123–136.

[8] J. Wang, Z. Li, M. Li, Y. Liu, and Z. Yang, "Sensor network navigation without locations", *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 7, pp. 1436–1446, 2012.

[9] E. Y. Pavlenko, "Functional model of adaptive network topology of large-scale systems based on dynamical graph theory", *Automatic Control and Computer Sciences*, vol. 56, no. 8, pp. 1016–1024, 2022.

[10] Z. Stanfield, M. Coşkun, and M. Koyutürk, "Drug response prediction as a link prediction problem", *Scientific reports*, vol. 7, no. 1, p. 40 321, 2017.

[11] T. J. Lakshmi and S. D. Bhavani, "Link prediction approach to recommender systems", *Computing*, 2023. DOI: 10.1007/s00607-023-01227-0.

[12] M. Nickel, K. Murphy, V. Tresp, and E. Gabrilovich, "A review of relational machine learning for knowledge graphs", *Proceedings of the IEEE*, vol. 104, no. 1, pp. 11–33, 2015.

[13] N. N. Daud, S. H. Ab Hamid, M. Saadoon, F. Sahran, and N. B. Anuar, "Applications of link prediction in social networks: A review", *Journal of Network and Computer Applications*, vol. 166, p. 102 716, 2020.

[14] Z. Qiu, J. Wu, W. Hu, B. Du, G. Yuan, and P. Yu, "Temporal link prediction with motifs for social networks", *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 3, pp. 3145–3158, 2023.

[15] H. Nassar, A. R. Benson, and D. F. Gleich, "Neighborhood and pagerank methods for pairwise link prediction", *Social Network Analysis and Mining*, vol. 10, no. 1, p. 63, 2020.

[16] X.-H. Yang, X. Yang, F. Ling, H.-F. Zhang, D. Zhang, and J. Xiao, "Link prediction based on local major path degree", *Modern Physics Letters B*, vol. 32, no. 29, pp. 1 850 348–306, 2018.

[17] S. Kumar, A. Mallik, and B. Panda, "Link prediction in complex networks using node centrality and light gradient boosting machine", *World Wide Web*, vol. 25, no. 6, pp. 2487–2513, 2022.

[18] J. Cheriyan and G. Sajeev, "m-PageRank: A novel centrality measure for multilayer networks", *Advances in Complex Systems*, vol. 23, no. 5, p. 2 050 012, 2020.

[19] F. Feng, X. Liu, B. Yong, R. Zhou, and Q. Zhou, "Anomaly detection in ad-hoc networks based on deep learning model: A plug and play device", *Ad Hoc Networks*, vol. 84, pp. 82–89, 2019.

[20] R. Meddeb, F. Jemili, B. Triki, and O. Korbaa, "Anomaly-based behavioral detection in mobile ad-hoc networks", *Procedia Computer Science*, vol. 159, pp. 77–86, 2019.

[21] M. Zhang and Y. Chen, "Link prediction based on graph neural networks", in *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*, 2018, pp. 5165–5175.

[22] B. P. Chamberlain *et al.*, *Graph neural networks for link prediction with subgraph sketching*, 2023. arXiv: 2209.15486 [cs.LG].

[23] M. Niu, L. Liu, and J. Shu, "Link quality prediction for wireless networks: Current status and future directions", in *Proceedings of the 2023 8th International Conference on Intelligent Information Technology*, 2023, pp. 52–56.

[24] J. Chen, Y. Han, D. Li, and J. Nie, "Link prediction and route selection based on channel state detection in UASNs", *International Journal of Distributed Sensor Networks*, vol. 7, no. 1, p. 939 864, 2011.

[25] H. Shao, L. Wang, H. Liu, and R. Zhu, "A link prediction method for MANETs based on fast spatio-temporal feature extraction and LSGANs", *Scientific Reports*, vol. 12, no. 1, p. 16 896, 2022.

[26] F. Harary and G. Gupta, "Dynamic graph models", *Mathematical and Computer Modelling*, vol. 25, no. 7, pp. 79–87, 1997.

[27] L. K. Ketshabetswe, A. M. Zungeru, M. Mangwala, J. M. Chuma, and B. Sigweni, "Communication protocols for wireless sensor networks: A survey and comparison", *Heliyon*, vol. 5, no. 5, E01591, 2019.

[28] D. Kandris, C. Nakas, D. Vomvas, and G. Koulouras, "Applications of wireless sensor networks: An up-to-date survey", *Applied system innovation*, vol. 3, no. 1, p. 14, 2020.

[29] M. Kim, S. Park, and W. Lee, "Energy and distance-aware hopping sensor relocation for wireless sensor networks", *Sensors*, vol. 19, no. 7, p. 1567, 2019.

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 4, 2023
journal homepage: www.mais-journal.ru

THEORY OF SOFTWARE

LTL-specification for development and verification of control programs

M. V. Neyzov[1], E. V. Kuzmin[2]

DOI: 10.18255/1818-1015-2023-4-308-339

[1]Institute of Automation and Electrometry SB RAS, 1 Academician Koptyug ave., Novosibirsk 630090, Russia.
[2]P.G. Demidov Yaroslavl State University, 14 Sovetskaya str., Yaroslavl 150003, Russia.

MSC2020: 68N30
Research article
Full text in Russian

Received November 6, 2023
After revision November 20, 2023
Accepted November 22, 2023

This work continues the series of articles on development and verification of control programs based on the LTL-specification. The essence of the approach is to describe the behavior of programs using formulas of linear temporal logic LTL of a special form. The developed LTL-specification can be directly verified by using a model checking tool. Next, according to the LTL-specification, the program code in the imperative programming language is unambiguously built. The translation of the specification into the program is carried out using a template.

The novelty of the work consists in the proposal of two LTL-specifications of a new form — declarative and imperative, as well as in a more strict formal justification for this approach to program development and verification. A transition has been made to a more modern verification tool for finite and infinite systems — nuXmv. It is proposed to describe the behavior of control programs in a declarative style. For this purpose, a declarative LTL-specification is intended, which defines a labelled transition system as a formal model of program behavior. This method of describing behavior is quite expressive — the theorem on the Turing completeness of the declarative LTL-specification is proved. Next, to construct program code in an imperative language, the declarative LTL-specification is converted into an equivalent imperative LTL-specification. An equivalence theorem is proved, which guarantees that both specifications specify the same behavior. The imperative LTL-specification is translated into imperative program code according to the presented template. The declarative LTL-specification, which is subject to verification, and the control program built on it are guaranteed to specify the same behavior in the form of a corresponding transition system. Thus, during verification, a model is used that is adequate to the real behavior of the control program.

Keywords: control software; declarative LTL-specification; imperative LTL-specification; model checking; complete transition system; pseudo-complete transition system; Minsky counter machine; nuXmv verifier; PLC program; ST language

INFORMATION ABOUT THE AUTHORS

Maxim V. Neyzov
corresponding author
orcid.org/0009-0000-6893-6137. E-mail: neyzov.max@gmail.com
Researcher.

Egor V. Kuzmin
orcid.org/0000-0003-0500-306X. E-mail: kuzmin@uniyar.ac.ru
Head of the Chair of Theoretical Informatics, Doctor of Science.

Funding: State task IAaE SB RAS, project No. 122031600173-8; Yaroslavl State University (project VIP-016).

For citation: M. V. Neyzov and E. V. Kuzmin, “LTL-specification for development and verification of control programs”, *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 308-339, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 4, 2023
сайт журнала: www.mais-journal.ru

THEORY OF SOFTWARE

LTL-спецификация для разработки и верификации управляющих программ

М. В. Нейзов[1], Е. В. Кузьмин[2]

DOI: 10.18255/1818-1015-2023-4-308-339

[1]Институт автоматики и электрометрии СО РАН, просп. Академика Коптюга, д. 1, г. Новосибирск, 630090 Россия.
[2]Ярославский государственный университет им. П.Г. Демидова, ул. Советская, д. 14, г. Ярославль, 150003 Россия.

УДК 004.424+519.683.8
Научная статья
Полный текст на русском языке

Получена 6 ноября 2023 г.
После доработки 20 ноября 2023 г.
Принята к публикации 22 ноября 2023 г.

Настоящая работа продолжает цикл статей по разработке и верификации управляющих программ на основе LTL-спецификации. Суть подхода заключается в описании поведения программ с помощью формул линейной темпоральной логики LTL специального вида. Полученная LTL-спецификация может быть непосредственно верифицирована с помощью инструмента проверки модели. Далее по LTL-спецификации однозначно строится код программы на императивном языке программирования. Перевод спецификации в программу осуществляется по шаблону.

Новизна работы состоит в предложении двух LTL-спецификаций нового вида — декларативной и императивной, а также в более строгом формальном обосновании данного подхода к разработке и верификации программ. Выполнен переход на более современный инструмент верификации конечных и бесконечных систем — nuXmv. Предлагается описывать поведение управляющих программ в декларативном стиле. Для этого предназначена декларативная LTL-спецификация, которая задаёт размеченную систему переходов как формальную модель поведения программы. Данный способ описания поведения является достаточно выразительным — доказана теорема о Тьюринг-полноте декларативной LTL-спецификации. Далее для построения кода программы на императивном языке декларативная LTL-спецификация преобразуется в эквивалентную императивную LTL-спецификацию. Доказана теорема об эквивалентности, которая гарантирует, что обе спецификации задают одно и то же поведение. Императивная LTL-спецификация транслируется в императивный код программы по представленному шаблону. Декларативная LTL-спецификация, которая подвергается верификации, и построенная по ней управляющая программа гарантированно задают одно и то же поведение в виде соответствующей системы переходов. Таким образом, при верификации используется модель, адекватная реальному поведению управляющей программы.

Ключевые слова: управляющее программное обеспечение; декларативная LTL-спецификация; императивная LTL-спецификация; полная система переходов; псевдополная система переходов; счётчиковая машина Минского; проверка модели; верификатор nuXmv; ПЛК-программа; язык ST

ИНФОРМАЦИЯ ОБ АВТОРАХ

Максим Вячеславович Нейзов
автор для корреспонденции
orcid.org/0009-0000-6893-6137. E-mail: neyzov.max@gmail.com
исследователь.

Егор Владимирович Кузьмин
orcid.org/0000-0003-0500-306X. E-mail: kuzmin@uniyar.ac.ru
заведующий кафедрой теоретической информатики, доктор физ.-мат. наук.

Финансирование: Госзадание ИАиЭ СО РАН, проект № 122031600173-8; ЯрГУ (проект VIP-016).

Для цитирования: M. V. Neyzov and E. V. Kuzmin, “LTL-specification for development and verification of control programs”, *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 308-339, 2023.

Введение

При решении задач управления проектируемая система естественным образом разделяется на систему управления и объект управления, оказывающие влияние друг на друга в процессе работы. *Программное обеспечение* (ПО), реализующее процесс управления, является *управляющим* ПО (УПО) и представляет собой систему управления или её часть. Объект управления может иметь различную природу и назначение. Например, в *киберфизической системе* [1, 2] УПО формирует воздействие на физическую среду, взаимодействуя с ней через сенсоры и исполнительные устройства. Так программа *программируемого логического контроллера* (ПЛК) [3] управляет технологическим процессом на производстве. В этом случае УПО является частью аппаратно-программной системы управления. Другой пример — в системе управления базами данных УПО выполняет различные манипуляции с массивом данных, осуществляя управление их выборкой и хранением. Здесь объектом управления является массив данных, который имеет не физическую, а программную природу, как и система управления. При решении же вычислительных задач проектируемая система содержит только программные компоненты. Согласно [4] дискретный преобразователь информации может быть представлен в виде автоматной модели, состоящей из взаимодействующих компонентов — *операционного* и *управляющего* автоматов. Операционный автомат преобразует информацию, а управляющий автомат осуществляет управление действиями по её преобразованию, т. е. описывает процесс управления вычислениями. Выделение задачи управления при разработке ПО присуще направлению, которое получило название *программная кибернетика* [5]. В рамках этого подхода управляющий и операционный автоматы могут быть реализованы в виде программных компонентов на любом уровне абстракции, который определяется решаемой задачей [6]. Таким образом, УПО может разрабатываться как отдельная часть проектируемой системы при решении не только управляющих, но и вычислительных задач.

УПО представляет собой *реагирующую систему* [7, 8], которая постоянно циклически взаимодействует с объектом управления в темпе протекающих в нём процессов. Цикл работы реагирующей системы состоит в получении входных данных и их записи во входные переменные программы, работы программы, которая вычисляет и обновляет внутренние и выходные переменные, выдаче значений выходных переменных на объект управления.

При проектировании систем могут предъявляться высокие требования к их надёжности и безопасности [9], часть из которых неизбежно превращается в требования к *корректности* УПО. Под корректностью понимается соответствие заданному набору поведенческих свойств. Процедура проверки соответствия называется *верификацией* [10]. Высокие гарантии соответствия обеспечивают формальные методы верификации [11]. Одним из таких методов является *проверка модели* (model checking) [12—15], которая требует наличия модели, адекватной реальному поведению УПО.

Настоящая работа продолжает цикл статей по разработке и верификации управляющих программ ПЛК на основе LTL-спецификации [16—32]. Данный подход предлагает разрабатывать LTL-спецификацию поведения программы, которая может быть непосредственно верифицирована методом проверки модели. Более того, спецификация является *конструктивной*, то есть по ней однозначно может быть построена программа на некотором языке программирования (ST [16, 25], LD [28, 29], IL [30], CFC [19, 20]) стандарта МЭК 61131-3 [33], а также модель поведения на входном языке верификатора Cadence SMV (http://www.mcmil.net/smv.html). LTL-спецификация обладает достаточной *выразительностью* для описания любого вычислительного процесса [17, 18]. LTL-спецификации счётчиковых машин представлены в [21, 22]. Также с помощью LTL-спецификации предлагается описывать согласованное поведение датчиков [31, 32], необходимое для проверки ряда программных свойств.

Новизна. В настоящей работе предложена более компактная модификация LTL-спецификации. Дано строгое формальное обоснование её конструктивности и выразительности. Выполнен переход

на более современный инструмент верификации конечных и бесконечных систем — nuXmv (https://nuxmv.fbk.eu).

Содержание работы. В разделе 1 излагается концепция разработки управляющих программ — представлены архитектурные особенности проектируемого УПО. Раздел 2 содержит используемые модели поведения в виде размеченных систем переходов. Вводятся понятия *полной* и *псевдополной* систем переходов. В разделе 3 приводится синтаксис и семантика языка LTL. В разделе 4 вводятся *декларативная* и *императивная* LTL-спецификации. Декларативная предназначена для описания поведения программ, а императивная — для построения кода программ. Доказывается теорема об эквивалентности данных LTL-спецификаций. В разделе 5 проведена оценка выразительности декларативной LTL-спецификации в смысле Тьюринг-полноты. Представлена *счётчиковая машина Минского* как модель алгоритма, эквивалентная по своим вычислительным возможностям *машине Тьюринга*. Приведён пример счётчиковой машины возведения числа в квадрат. Доказывается теорема о полноте по Тьюрингу декларативной LTL-спецификации. В разделе 6 выполняется постановка задачи верификации. Накладывается ограничение на задачу с целью обеспечения её разрешимости. Раздел 7 содержит пример разработки и верификации с помощью инструмента nuXmv декларативной LTL-спецификация ПЛК-программы возведения числа в квадрат. В разделе 8 представлена схема построения ST-программы по императивной LTL-спецификации. Приведён код ST-программы возведения числа в квадрат. Далее делается заключение. Приложение демонстрирует процедуру построения LTL-спецификации, предложенную в теореме о Тьюринг-полноте.

1. Концепция разработки управляющих программ

Управляющая программа работает циклически. В каждом *цикле управления* значение переменной может либо измениться, либо остаться прежним. Мы сосредоточены только на изменениях значений переменных, так как в задачах управления именно они несут в себе ключевую смысловую нагрузку. При управлении всегда известно, почему состояние управляемого объекта нужно изменить, почему, например, то или иное устройство должно быть включено или выключено. Причины изменения значений переменных мы декларируем в виде LTL-спецификации. Также спецификация содержит информацию о том, каким образом происходят эти изменения.

В целом поведение программы — совокупное поведение всех её переменных $V = \{v_1, \ldots, v_n\}$. Данное множество содержит входные и внутренние/выходные переменные. Изначально поведение всех переменных из V не декларировано и является абсолютно недетерминированным. Далее декларированию подлежит поведение только внутренних/выходных переменных. Декларация накладывает некоторые ограничения на их поведение, после чего оно становится строго детерминированным. Оставшийся недетерминизм входных переменных необходим для того, чтобы сохранить всё многообразие в поведении детерминированной программы, так как именно от поведения входных переменных зависит поведение внутренних/выходных переменных. Чтобы не потерять ни один сценарий работы программы, необходимо подавать на её вход любые допустимые значения в любой последовательности, что реализуется абсолютным недетерминизмом поведения входов.

Для анализа и декларирования причин изменения значения переменной часто требуется знать её предыдущее значение. Например, для определения момента нажатия кнопки (переднего фронта сигнала) необходимо располагать информацией о том, была ли она нажата в предыдущем цикле работы. Для этой цели вводится набор вспомогательных переменных $\_V = \{\_v_1, \ldots, \_v_n\}$, которые хранят предыдущие значения соответствующих переменных из V. Поведение вспомогательных переменных не декларируется и всегда известно. При использовании переменных из $\_V$ можно считать, что знак лидирующего подчёркивания «_» является псевдооператором обращения к предыдущему значению соответствующей переменной из V. Сравнивая значения переменных v_i и $\_v_i$ (где $i = 1, \ldots, n$), можно судить о том, произошло ли изменение значения переменной v_i.

Далее декларацию в виде LTL-спецификации можно верифицировать и построить по ней программу. Построенная программа будет обладать двумя особенностями:

1. Значение каждой переменной изменяется не более одного раза за один цикл управления.
2. Значение каждой переменной изменяется только в одном месте программы в некотором простом операторном блоке.

Эти две особенности позволяют иметь прозрачное и наглядное представление о том, каким образом происходит изменение значения той или иной переменной при переходе программы из одного состояния в другое. Для каждой переменной устанавливается четкая зависимость нового её значения от прежних значений переменных, которые были получены при выполнении программы на предыдущем проходе рабочего цикла, и новых значений переменных, уже вычисленных при выполнении программы на текущем проходе рабочего цикла. Условие изменения значения переменной только в одном месте программы облегчает отладку, дает возможность простой оценки степени готовности и объема текста программы. Очевидно, что за один проход рабочего цикла значение любой переменной возрастает, убывает или остается без изменения по отношению к её значению, полученному на предыдущем проходе рабочего цикла. Если вообще не обращаться к переменной с целью присваивания какого-либо значения, то она сохранит своё прежнее значение.

2. Система переходов как модель поведения программы

Пусть программа имеет основные $v_1, \ldots, v_n$ и вспомогательные $\_v_1, \ldots, \_v_n$ переменные, которые принимают значения из соответствующих областей $D_1, \ldots, D_n$. Множество (в общем случае бесконечное) $S = (D_1 \times \ldots \times D_n)^2$ будет являться пространством всех возможных состояний программы. Множество $V = \{v_1, \ldots, v_n\}$ содержит входные и внутренние/выходные переменные. Вектор $(d_1, \ldots, d_n, \_d_1, \ldots, \_d_n) \in S$ значений переменных $(v_1, \ldots, v_n, \_v_1, \ldots, \_v_n)$ описывает конкретное состояние программы и содержит текущие значения входных переменных и вновь вычисленные на их основе значения внутренних/выходных переменных. В переменных $\_v_1, \ldots, \_v_n$ хранятся предыдущие значения переменных $v_1, \ldots, v_n$. Каждому циклу управления соответствует *переход* между состояниями. Если состояние не изменилось, то считается, что произошёл переход по петле в то же самое состояние.

Все возможные переходы формируют поведение программы. В качестве модели поведения программы примем *размеченную систему переходов* (labelled transition system) $LTS = \langle S, S_0, R, P, L \rangle$, где S — множество состояний (в общем случае бесконечное), $S_0 \subseteq S$ — множество начальных состояний, $R \subseteq S \times S$ — *тотальное* отношение переходов, $P = \{p_1, \ldots, p_m\}$ — множество атомарных утверждений относительно значений переменных $v_1, \ldots, v_n$ и $\_v_1, \ldots, \_v_n$, $L : S \to 2^P$ — функция разметки состояний атомарными утверждениями. Тотальность отношения переходов R означает, что из любого состояния $s \in S$ существует переход: $(\forall s \in S)\ (\exists s' \in S)\ (s, s') \in R$.

Граф системы переходов — граф, вершинами которого являются состояния из S, а дугами — переходы из R. *Путь* в системе переходов — бесконечная последовательность $\pi = s_0 s_1 s_2 \ldots$, где $(s_i, s_{i+1}) \in R$, $i \in \mathbb{N}_0 = \mathbb{N} \cup \{0\}$. Путь формирует бесконечный маршрут в графе системы переходов. Система переходов LTS задаёт множество путей Π_{LTS}, исходящих из начальных состояний:

$$\Pi_{LTS} = \{\, \pi \in S^\omega \mid \big(\pi(0) \in S_0\big) \wedge (\forall i \in \mathbb{N}_0)\ \big(\pi(i), \pi(i+1)\big) \in R \,\},$$

где S^ω — множество всех бесконечных слов из алфавита S, $\pi(i)$ — i-ое состояние пути π.

Моделью поведения программы с недетерминированным поведением всех её переменных $v_1, \ldots, v_n, \_v_1, \ldots, \_v_n$ будет *полная* система переходов $cLTS = \langle S, S_0, R_c, P, L \rangle$, где $S_0 = S$. В отличие от LTS система $cLTS$ имеет отношение переходов $R_c = S \times S$, которое задаёт *полный* граф переходов по состояниям. Действительно, при абсолютном недетерминизме всегда возможен переход из любого состояния в любое другое, включая само себя. Заметим, что программа с недетерминированным

поведением всех переменных (*абсолютно недетерминированная* программа) содержит в себе поведение любой другой программы с этим же набором переменных $v_1, \ldots, v_n, \_v_1, \ldots, \_v_n$. Здесь $S_0 = S$ для того, чтобы *cLTS* содержала любой путь, начинающийся с любого состояния.

Если на полную систему переходов *cLTS* наложить ограничение в виде ранее оговорённого поведения переменных $\_v_1, \ldots, \_v_n$, то получим *псевдополную* систему переходов $pLTS = \langle S, S_0, R'_c, P, L\rangle$, где $S_0 = S$, $R'_c = \{\big((a, \_a), (a', a)\big) \in R_c\}$. Псевдополная система переходов *pLTS* содержит в себе поведение любой другой программы, которая сохраняет предыдущие значения переменных $v_1, \ldots, v_n$ в переменных $\_v_1, \ldots, \_v_n$.

При декларировании поведения переменных $v_1, \ldots, v_n$ накладываются определённые ограничения на псевдополную систему переходов *pLTS*. В итоге, модель поведения программы *LTS* получается из псевдополной системы переходов *pLTS* путем устранения переходов, нарушающих заданную декларацию поведения.

3. Язык логики LTL

Линейная темпоральная логика LTL (Linear Temporal Logic), или темпоральная логика линейного времени (Linear-Time Temporal Logic), представляет собой расширение классической логики высказываний с помощью модальных операторов, позволяющих учитывать временной аспект в последовательностях событий [12—15]. Темпоральная логика LTL находит важное применение в области формальной верификации, где она используется для описания требований к аппаратным и программным системам [8, 34]. Мы будем использовать LTL для:

- описания требований к псевдополной системе переходов *pLTS*, чтобы сформировать из неё модель поведения программы *LTS*;
- описания требований к модели поведения программы *LTS*, чтобы провести их верификацию.

Формулы LTL имеют следующую грамматику при $P = \{p_1, \ldots, p_m\}$:

$$\varphi, \psi := \mathit{true} \mid \mathit{false} \mid p_1 \mid \ldots \mid p_m \mid \neg\varphi \mid \varphi \wedge \psi \mid \varphi \vee \psi \mid \varphi \Rightarrow \psi \mid \mathbf{X}\varphi \mid \psi\mathbf{U}\varphi \mid \mathbf{F}\varphi \mid \mathbf{G}\varphi.$$

Помимо классических булевых операторов в LTL-формуле могут присутствовать и темпоральные: $\mathbf{X}\varphi$ означает, что формула φ должна выполнятся в следующем состоянии, $\mathbf{F}\varphi$ требует, чтобы φ выполнялась в некотором будущем состоянии, $\mathbf{G}\varphi$ гарантирует, что в текущем и во всех будущих состояниях будет выполняться φ. Формула $\psi\mathbf{U}\varphi$ означает, что φ должна выполнятся в текущем или будущем состоянии, а во всех предшествующих состояниях должна выполняться ψ. Операторы **F** и **G** являются двойственными: $\mathbf{G}\varphi = \neg\mathbf{F}\neg\varphi$. При этом сам оператор **F** представляет собой специальный случай применения оператора **U**, а именно $\mathbf{F}\varphi = \mathit{true}\mathbf{U}\varphi$.

Индуктивно определим отношение выполнимости $\vDash$ формулы φ логики LTL для произвольного состояния s_i некоторого пути $\pi = s_0\, s_1\, s_2 \ldots$ системы переходов, где $i \in \mathbb{N}_0$, $p \in P$:

$$\begin{aligned}
&s_i \vDash \mathit{true}; && s_i \nvDash \mathit{false};\\
&s_i \vDash p && \Longleftrightarrow p \in L(s_i);\\
&s_i \vDash \neg\varphi && \Longleftrightarrow s_i \nvDash \varphi;\\
&s_i \vDash \varphi \wedge \psi && \Longleftrightarrow s_i \vDash \varphi \ \text{ и } \ s_i \vDash \psi;\\
&s_i \vDash \varphi \vee \psi && \Longleftrightarrow s_i \vDash \varphi \ \text{ или } \ s_i \vDash \psi;\\
&s_i \vDash \varphi \Rightarrow \psi && \Longleftrightarrow s_i \vDash \neg\varphi \ \text{ или } \ s_i \vDash \psi;\\
&s_i \vDash \mathbf{X}\varphi && \Longleftrightarrow s_{i+1} \vDash \varphi;\\
&s_i \vDash \psi\mathbf{U}\varphi && \Longleftrightarrow (\exists k \geq i)\ s_k \vDash \varphi \ \text{ и } \ (\forall j,\ i \leq j < k)\ s_j \vDash \psi;\\
&s_i \vDash \mathbf{F}\varphi && \Longleftrightarrow (\exists k \geq i)\ s_k \vDash \varphi;\\
&s_i \vDash \mathbf{G}\varphi && \Longleftrightarrow (\forall j \geq i)\ s_j \vDash \varphi.
\end{aligned}$$

Также определим семантику отношения $\vDash$ на путях и системах переходов:

$$\begin{aligned}
\pi \vDash \varphi &\iff \pi(0) \vDash \varphi;\\
\Pi \vDash \varphi &\iff (\forall \pi \in \Pi)\ \pi \vDash \varphi;\\
LTS, s \vDash \varphi &\iff (\forall \pi \in \Pi_{LTS})\ [\pi(0) = s] \Rightarrow [\pi \vDash \varphi];\\
LTS \vDash \varphi &\iff (\forall s \in S_0)\ LTS, s \vDash \varphi.
\end{aligned}$$

Формула φ выполняется на пути π, когда она выполняется в его начальном состоянии, а на множестве путей Π, когда она выполняется на всех его путях. Формула φ выполняется в состоянии $s \in S$ системы переходов LTS, когда φ выполняется для всех путей в LTS, начинающихся с состояния s. Формула φ выполняется на системе переходов LTS, когда φ выполняется для любого начального состояния $s \in S_0$. Обозначим $[\![\varphi]\!]_{LTS}$ множество всех путей в LTS, на которых истинна формула φ, т. е. $[\![\varphi]\!]_{LTS} = \{\, \pi \in \Pi_{LTS} \mid \pi \vDash \varphi \,\}$.

4. LTL-спецификация поведения программ

4.1. Декларативная и императивная LTL-спецификации

Для задания поведения программных переменных $V = \{v_1, \ldots, v_n\}$ используется *декларативная* LTL-спецификация. Для переменной $v \in V$ это пара формул специального вида (1), (2), которая описывает, каким образом происходит изменение значения переменной:

$$\mathbf{GX}\big(\neg(v = \_v) \Rightarrow cond_1 \wedge (v = expr_1) \vee \ldots \vee cond_k \wedge (v = expr_k)\big), \tag{1}$$

$$\mathbf{GX}\big((v = \_v) \Rightarrow \neg\,(cond_1 \vee \ldots \vee cond_k)\big). \tag{2}$$

Здесь $cond_i$ — условие (логическое выражение), выполнимость которого необходима для изменения значения переменной v в соответствии с выражением $expr_i$, $i = 1, \ldots, k$. Выражения $cond_i$ и $expr_i$ строятся над множеством переменных $V \cup \_V$, где $V = \{v_1, \ldots, v_n\}$ и $\_V = \{\_v_1, \ldots, \_v_n\}$, с использованием констант и применением логических и арифметических операторов, а также операторов сравнения. При этом предполагается, что при любых значениях переменных из $V \cup \_V$ для этих выражений выполняются 1) условие изменчивости: $cond_i \Rightarrow \neg\,(\_v = expr_i)$, $\forall i = 1, \ldots, k$, и 2) условие ортогональности: $cond_i \Rightarrow \neg cond_j$, $\forall\, i, j = 1, \ldots, k$ при $i \neq j$.

Первая LTL-формула (1) утверждает, что если значение переменной v изменилось (новое значение v отличается от старого $\_v$), то было верно одно из условий $cond_i$, а переменная v получила значение выражения $expr_i$. Вторая LTL-формула (2) имеет жёсткую конструкцию, составленную из элементов первой формулы (1). Она утверждает, что если значение переменной v не изменилось (осталось равным предыдущему значению $\_v$), то ни одно из условий $cond_i$ не является истинным. Приставка **GX** говорит о том, что утверждение в скобках действует всегда, начиная с первого цикла работы программы, т. е. не захватывает только начальное состояние $s \in S_0$.

Условие изменчивости предохраняет от противоречия в формуле (1), которое может возникнуть при равенстве значения выражения $expr_i$ прежнему значению переменой v, а условие ортогональности предназначено для соблюдения детерминизма в поведении этой переменной.

Требование *изменчивости* без учёта начального состояния программы $s_0 \in S$ может быть представлено в виде LTL-формулы следующим образом:

$$\mathbf{GX}\big(cond_1 \Rightarrow \neg\,(\_v = expr_1)\big) \wedge \ldots \wedge \mathbf{GX}\big(cond_k \Rightarrow \neg\,(\_v = expr_k)\big), \tag{3}$$

т. е. при истинном условии $cond_i$ выражение $expr_i$ должно возвращать значение, отличное от предыдущего значения переменной v.

Аналогично для условия *ортогональности* имеем набор LTL-формул ($\forall\, i, j = 1, \ldots, k$ при $i \neq j$):

$$\mathbf{GX}\big(cond_i \Rightarrow \neg cond_j\big), \tag{4}$$

т. е. истинным может быть не более одного условия $cond_i$.

Конъюнкция множества формул вида (1) и (2) для внутренних/выходных переменных из V при соблюдении условий (3) и (4) образует *декларативную* LTL-спецификацию поведения *неинициализированной* программы φ_{var}.

Для инициализации будем использовать LTL-формулу φ_0 специального вида без темпоральных операторов: $I(v_1, \ldots, v_n) \wedge (\_v_1 = v_1) \wedge \ldots \wedge (\_v_n = v_n)$, где $I(v_1, \ldots, v_n)$ — это предикат, задающий ограничения на основные переменные. Начальные значения вспомогательных переменных $\_v_1, \ldots, \_v_n$ всегда совпадают с начальными значениями соответствующих основных переменных.

Формулу вида $\varphi = \varphi_{var} \wedge \varphi_0$ будем называть *декларативной* LTL-спецификацией поведения некоторой программы. Эта формула на основе псевдополной системы переходов *pLTS* задаёт систему переходов *LTS*. Полученная система переходов *LTS* будет содержать те и только те пути, которые соответствуют возможным сценариям работы рассматриваемой программы.

Продемонстрируем идею спецификации на простом примере. Зададим поведение одной булевой переменной v так, чтобы её значение более не менялось после перехода от 0 к 1. Данное поведение изображено в виде графа (рис. 1). Вершинам графа соответствуют значения переменной v (состояния), а дугам — возможные переходы между состояниями. На графе видно, что запрещено сохранять значение 0 и изменять значение с 1 на 0 (соответствующие дуги перечёркнуты).

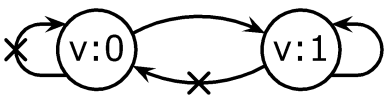


Fig. 1. Variable v behavior

Рис. 1. Поведение переменной v

Программа будет иметь две булевы переменные v и $\_v$. Потенциальное число состояний программы $2^2 = 4$, обозначим их $S = \{s_0, s_1, s_2, s_3\}$. Каждое состояние представляет собой уникальный вектор значений переменных v и $\_v$. В вершине графа значение переменной v указано над чертой, значение переменной $\_v$ — под чертой (рис. 2a).

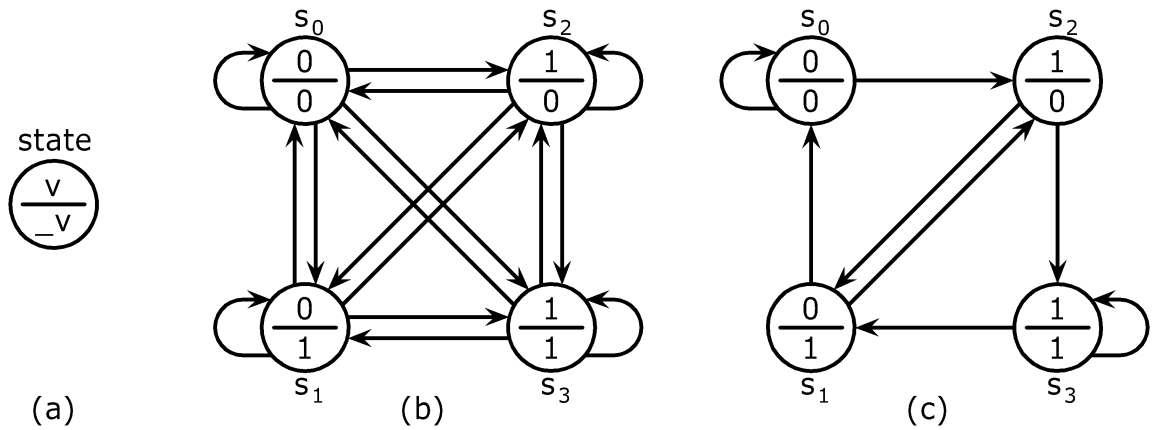


Fig. 2. State notation (a), complete transition system (b), pseudo-complete transition system (c)

Рис. 2. Обозначение состояния (a), полная система переходов (b), псевдополная система переходов (c)

Поведение абсолютно недетерминированной программы с переменными v и $\_v$ описывает полная система переходов *cLTS* (рис. 2b). Здесь возможен переход из любого состояния в любое другое,

включая само себя. Если переменная $\_v$ будет хранить предыдущее значение переменной v, то это накладывает некоторое ограничение на поведение программы. Любое возможное поведение такой программы описывает псевдополная система переходов *pLTS* (рис. 2c). Здесь отсутствуют переходы, которые нарушают условие равенства текущего значения переменной $\_v$ значению переменной v на предыдущем шаге (в предыдущем состоянии). Например, отсутствует переход из s_0 в s_1, так как в состоянии s_1 значение переменной $\_v = 1$, хотя в предыдущем состоянии s_0 значение переменной $v = 0$. То есть предыдущее значение переменной v не сохраняется в переменной $\_v$. И наоборот, присутствует переход из s_1 в s_0, так как предыдущее значение переменной v сохраняется в переменной $\_v$: $v = 0$ в состоянии s_1 и $\_v = 0$ в состоянии s_0.

Опишем поведение программы в виде декларативной LTL-спецификации, которая представляет собой в данном случае всего две формулы:

$$\varphi_1\colon\ \mathbf{GX}\big(\neg(v = \_v) \Rightarrow\ (\_v = 0) \wedge (v = \neg\_v)\big),$$
$$\varphi_2\colon\ \mathbf{GX}\big((v = \_v) \Rightarrow\ \neg\,(\_v = 0)\big).$$

Эти формулы задают именно те требования, которые схематично изображены на рис. 1:

1. Единственная причина изменения значения переменной v — это её нулевое значение (переход из 0 в 1 разрешён, а из 1 в 0 запрещён).
2. При наличии этой причины ($\_v = 0$) изменение значения переменной неизбежно (петля из 0 в 0 запрещена).

Формула φ_1 утверждает, что если изменилось значение переменной v, то её предыдущее значение было равно 0, и изменение заключалось в инверсии этого значения (формализация пункта 1). Формула φ_2 фиксирует, что если значение переменной v не изменилось, то её предыдущее значение не равно 0. Это эквивалентно утверждению, что если предыдущее значение переменной v равно 0, то значение переменной изменилось (формализация пункта 2).

Проследим, как каждая формула корректирует псевдополную систему переходов *pLTS* (рис. 2,c). Формула φ_1 нарушается в состоянии $s \in S$, если из него есть фрагмент пути не нулевой длины в состояние $s' \in S$, в котором истинна левая часть импликации и ложна правая. В нашем случае $s' = s_1$. Здесь v не равно $\_v$ (истинна левая часть импликации) и $\_v$ не равно 0 (ложна правая часть импликации). Поэтому любой фрагмент пути, приводящий в состояние s_1, является контрпримером, нарушающим истинность формулы φ_1. Примеры таких фрагментов: $s_0s_2s_1$, $s_2s_3s_1$. Таким образом, формула φ_1 нарушается на *pLTS* (рис. 2c), т. е. $pLTS \nvDash \varphi_1$. Чтобы этого не происходило, удалим все дуги, входящие в состояние s_1 (рис. 3a). Теперь контрпримеров, нарушающих формулу φ_1, нет.

Аналогично формула φ_2 нарушается в состоянии $s \in S$, если из него есть фрагмент пути не нулевой длины в состояние $s' \in S$, в котором истинна левая часть импликации и ложна правая. В данном случае $s' = s_0$. Здесь v равно $\_v$ (истинна левая часть импликации) и $\_v$ равно 0 (ложна правая часть импликации). Поэтому любой фрагмент пути, приводящий в состояние s_0, является контрпримером, нарушающим истинность формулы φ_2. Примеры таких фрагментов: s_0s_0, s_1s_0. Таким образом, формула φ_2 нарушается на *pLTS*, удовлетворяющей формуле φ_1 (рис. 3a). Чтобы этого не происходило, удалим все дуги, входящие в состояние s_0 (рис. 3b). Теперь контрпримеров, нарушающих формулы φ_1 или φ_2, нет. С помощью представленной LTL-спецификации поведения переменной v из псевдополной системы переходов *pLTS* были отобраны только те пути $[\![\varphi_1 \wedge \varphi_2]\!]_{pLTS}$, на которых выполняются формулы спецификации φ_1 и φ_2.

Зафиксируем одно начальное состояние программы: $S_0 = \{s_0\}$. Для этого добавим следующую LTL-формулу инициализации $\varphi_0 : (v = 0) \wedge (\_v = v)$, которая выполняется только в состоянии s_0 (на рис. 3c отмечено входной стрелкой). Из оставшихся состояний s_1, s_2, s_3 (рис. 3b) пути не могут начинаться, так как это приведёт к нарушению формулы φ_0. Состояния s_2, s_3 достижимы из начального состояния s_0, поэтому с их помощью могут образовываться пути. Состояние s_1 не является

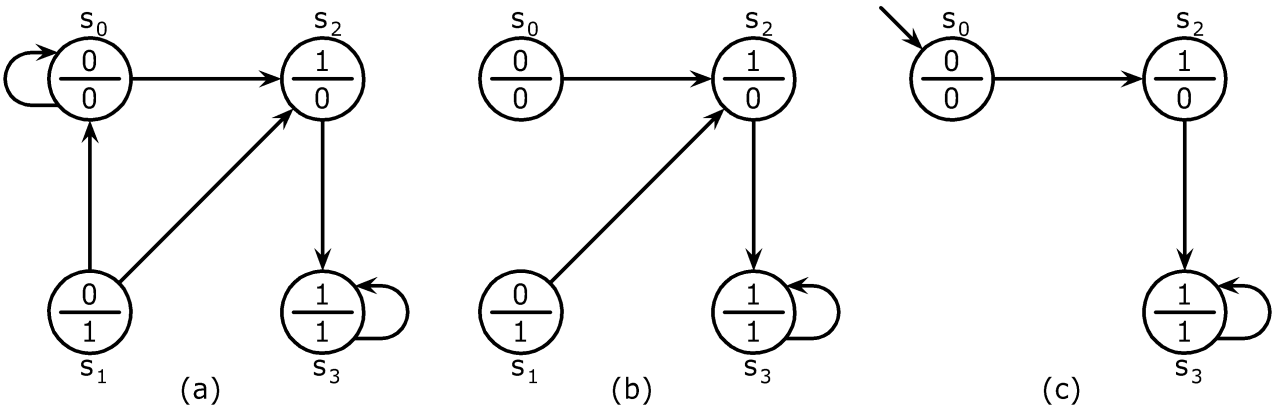


Fig. 3. Transition system: satisfies φ_1 (a), satisfies φ_1 and φ_2 (b), initialized program (c)

Рис. 3. Система переходов: удовлетворяющая φ_1 (a), удовлетворяющая φ_1 и φ_2 (b), инициализированной программы (c)

начальным и недостижимо из состояния s_0, оно не может формировать пути, удовлетворяющие формуле φ_0. Удалим недостижимое состояние s_1 и инцидентные ему дуги — получим систему переходов инициализированной программы (рис. 3c). Данная система переходов задаёт множество путей $[\![\varphi_0 \wedge \varphi_1 \wedge \varphi_2]\!]_{pLTS}$, на котором выполняются все формулы спецификации $\varphi_0, \varphi_1, \varphi_2$.

Проследим поведение переменной v. Изначально (в состоянии s_0) её значение равно 0. Далее значение обязательно меняется на 1 (неизбежно происходит переход в состояние s_2). После чего значение переменной v больше не изменяется (после перехода в состояние s_3, в котором $v = 1$, происходит зацикливание в нём). Таким образом, данная система переходов в точности задаёт поведение переменной v, представленное на рис. 1.

Задание системы переходов с помощью LTL-спецификации. На основе псевдополной системы переходов $pLTS = \langle S_{pLTS}, S_0, R_{pLTS}, P, L\rangle$ с помощью декларативной LTL-спецификации φ задаётся система переходов $LTS = \langle S, S'_0, R_\varphi, P, L'\rangle$, описывающая поведение программы. Отношение переходов R_φ имеет следующий вид:

$$R_\varphi = \{\, (s_1, s_2) \in R_{pLTS} \mid (\exists \pi \in \Pi_{pLTS})\ \pi = \sigma s_1 s_2 \ldots \vDash \varphi \,\},$$

где Π_{pLTS} — множество всех путей системы переходов $pLTS$, σ — конечный фрагмент пути, $|\sigma| \in \mathbb{N}_0$. Таким образом, из псевдополной системы переходов $pLTS$ отбираются только те переходы, которые образуют пути, удовлетворяющие формуле φ.

Множество состояний S можно описать так:

$$S = \{\, s \in S_{pLTS} \mid (\exists s' \in S_{pLTS})\ [(s, s') \in R_\varphi \vee (s', s) \in R_\varphi] \,\},$$

т. е. S содержит только те состояния, которые присутствуют в отобранных переходах. Множество начальных состояний $S'_0 = \{\, s \in S \mid s \vDash \varphi_0 \,\}$, где φ_0 — формула инициализации из спецификации φ. Функция разметки $L' \colon S \to 2^P$ совпадает с L на множестве S, т. е. $(\forall s \in S)\ L'(s) = L(s)$.

Декларативная LTL-спецификация φ отбирает множество путей $[\![\varphi]\!]_{pLTS}$ из псевдополной системы переходов $pLTS$. Формуле φ соответствует построенная выше система переходов LTS, которая задаёт отобранное множество путей $\Pi_{LTS} = [\![\varphi]\!]_{pLTS}$.

Задачи управления конвейером. Рассмотрим простой пример задания поведения программы управления конвейером (рис. 4). Конвейер транспортирует детали (подаются из лотка слева) в тару (находится справа). На конвейере установлено два датчика: датчик наличия и датчик массы детали. Датчик наличия детали устанавливает $d = 1$, когда обнаруживает её наличие, и $d = 0$ при её отсутствии. Показания датчика массы детали w принимают значения из $\mathbb{N}_0$. За работу конвейера отвечает переменная con, которая принимает значения из $\{0, 1\}$.

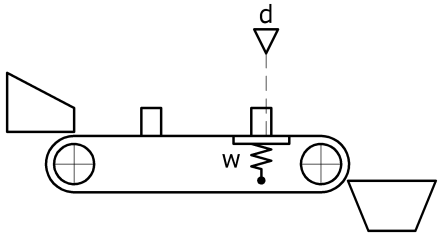


Fig. 4. Conveyor operation

Рис. 4. Работа конвейера

Пусть конвейер изначально включен и пуст: $(con = 1) \wedge (d = 0) \wedge (w = 0)$. Поведение входных переменных (d и w), отражающих значения датчиков, не специфицируем в виде LTL-формул, поэтому оно остаётся абсолютно недетерминированным.

Задача 1: требуется набрать заданное количество деталей $c_1 \in \mathbb{N}_0$ в тару и затем остановить конвейер. Для этого введём счётчик деталей n, который принимает значения из $\mathbb{N}_0$. Сначала опишем поведение программы, которая выполняет подсчёт деталей.

Формула инициализации программы подсчёта деталей имеет вид:

$$\varphi_0 : (d = 0) \wedge (n = 0) \wedge (\_d = d) \wedge (\_n = n).$$

LTL-формула, описывающая поведение счётчика деталей n, следующая:

$$\varphi_n : \mathbf{GX}\big(\neg(n = \_n) \Rightarrow\ (\_d = 0) \wedge (d = 1) \wedge (n = \_n + 1)\big) \wedge$$
$$\mathbf{GX}\big((n = \_n) \Rightarrow\ \neg\,[(\_d = 0) \wedge (d = 1)]\big).$$

Если значение счётчика n изменилось (новое значение отличается от предыдущего), то обнаружен передний фронт сигнала от датчика наличия детали (только новое значение d равно единице), и изменением является увеличение счётчика n на единицу. Вторая строка формулы φ_n утверждает, что если значение счётчика n не изменилось (новое значение равно предыдущему), то не было переднего фронта сигнала от датчика наличия детали.

Рассмотрим систему переходов программы подсчёта деталей (рис. 5b). Обозначение её состояний представлено на рис. 5a. Здесь над чертой указаны значения переменных n и d, под чертой — значения переменных $\_n$ и $\_d$. Значение переменной n выделяется рамкой.

На рис. 5b в начальном состоянии s_0 значения всех переменных равны нулю. Из любого состояния всегда исходят две дуги: одна для перехода при входном сигнале $d = 0$, другая — для перехода

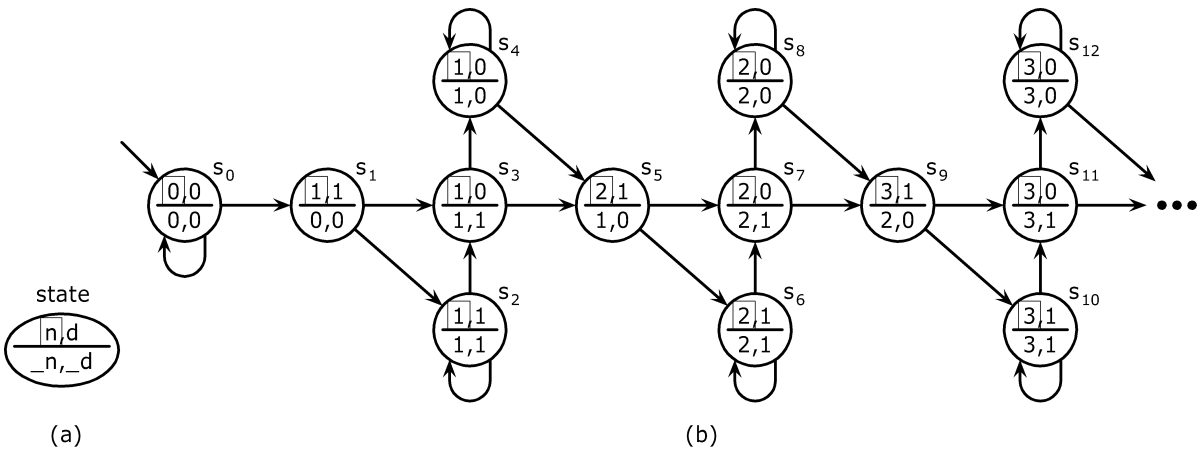


Fig. 5. State notation (a), transition system of the details counting program (b)

Рис. 5. Обозначение состояния (a), система переходов программы подсчёта деталей (b)

при входном сигнале $d = 1$. Видно, что переменные $\_n$ и $\_d$ в любом состоянии (кроме начального) принимают соответственно значения n и d из предыдущего состояния. Система переходов содержит только такие состояния, в которых отражена причина увеличения/сохранения значения переменной n — наличие/отсутствие переднего фронта сигнала от датчика. Других изменений значения переменной n не происходит, так как все прочие состояния удалены из системы переходов. При этом система переходов (рис. 5b) имеет бесконечное число состояний, в которых наблюдается монотонное увеличение значения переменной n при движении слева направо.

Теперь расширим программу, добавив в неё управление конвейером. Формула инициализации расширенной программы будет иметь вид:

$$\varphi_0 : (con = 1) \wedge (d = 0) \wedge (n = 0) \wedge (\_con = con) \wedge (\_d = d) \wedge (\_n = n).$$

После транспортировки заданного количества деталей конвейер должен остановиться. Запишем это с помощью следующей LTL-формулы спецификации поведения переменной *con*:

$$\begin{aligned}\varphi_{con1} : \mathbf{GX}\big(\neg(con = \_con) \implies (\_con = 1) \wedge (n = c_1) \wedge (con = 0)\big) \wedge \\ \mathbf{GX}\big((con = \_con) \implies \neg\,[(\_con = 1) \wedge (n = c_1)]\big).\end{aligned}$$

Если значение переменной *con* изменилось, то конвейер был включён, счётчик n достиг значения c_1, и изменение заключалось в выключении конвейера. Если значение переменной *con* не изменилось, то не возник нужный момент для отключения конвейера. Спецификация поведения всей программы — формула $\varphi = \varphi_0 \wedge \varphi_n \wedge \varphi_{con1}$.

Задача 2: требуется набрать в тару множество деталей с общей массой $c_2 \in \mathbb{N}_0$ и остановить конвейер. Для этого введём счётчик массы m, который принимает значения из $\mathbb{N}_0$. В этом случае формула инициализации программы имеет вид:

$$\varphi_0 : (con = 1) \wedge (d = 0) \wedge (m = 0) \wedge (\_con = con) \wedge (\_d = d) \wedge (\_m = m).$$

Поведение счётчика массы m зададим как

$$\begin{aligned}\varphi_m : \mathbf{GX}\big(\neg(m = \_m) \implies (\_d = 0) \wedge (d = 1) \wedge (m = \_m + w)\big) \wedge \\ \mathbf{GX}\big((m = \_m) \implies \neg\,[(\_d = 0) \wedge (d = 1)]\big).\end{aligned}$$

Если значение счётчика m изменилось, то обнаружен передний фронт сигнала от датчика наличия детали, а изменением является увеличение счётчика m на значение массы детали w, которая сейчас находится на измерительной подложке.

После транспортировки заданной массы деталей конвейер должен остановиться. LTL-формула, описывающая поведение переменной *con*, имеет следующий вид:

$$\begin{aligned}\varphi_{con2} : \mathbf{GX}\big(\neg(con = \_con) \implies (\_con = 1) \wedge (m \geq c_2) \wedge (con = 0)\big) \wedge \\ \mathbf{GX}\big((con = \_con) \implies \neg\,[(\_con = 1) \wedge (m \geq c_2)]\big).\end{aligned}$$

Если значение переменной *con* изменилось, то конвейер был включён, счётчик m достиг или превысил значение c_2, а изменение заключалось в выключении конвейера. Спецификация поведения всей программы — формула $\varphi = \varphi_0 \wedge \varphi_m \wedge \varphi_{con2}$.

Императивная LTL-спецификация. Для построения императивного кода программы по декларативной LTL-спецификации потребуется промежуточная формализация — императивная LTL-спецификация. Данная спецификация эквивалентна декларативной LTL-спецификации (выражает то же самое поведение), но при этом может быть непосредственно преобразована в императивный код программы, написанной, например, на языке ST.

Императивная LTL-спецификация для переменной v с учётом (3) и (4) имеет следующий вид:

$$\mathbf{GX}\big(cond_1 \Rightarrow (v = expr_1)\big) \wedge \ldots \wedge \mathbf{GX}\big(cond_k \Rightarrow (v = expr_k)\big), \tag{5}$$

$$\mathbf{GX}\big(\neg\, cond_1 \wedge \ldots \wedge \neg\, cond_k \Rightarrow (v = \_v)\big). \tag{6}$$

Эта спецификация описывает условия и соответствующие им правила изменения значения переменной v в стиле «если . . . , то . . . ».

4.2. Теорема об эквивалентности LTL-спецификаций

Лемма 1. *Из справедливости декларативной LTL-спецификации следует справедливость императивной LTL-спецификации, т. е. при условиях* (3) *и* (4) *имеем* (1), (2) $\vdash$ (5), (6).

Доказательство. Разобьём доказательство леммы на две части. В первой части покажем выполнимость вывода (4), (1), (2) $\vdash$ (5). Во второй части — справедливость логического вывода (1) $\vdash$ (6).

Докажем первую часть леммы:

Применим закон контрапозиции к формуле (2):

$$\mathbf{GX}\big((cond_1 \vee \ldots \vee cond_k) \Rightarrow \neg\,(v = \_v)\big). \tag{7}$$

Из (7) и (1) по транзитивности импликации получим:

$$\mathbf{GX}\big((cond_1 \vee \ldots \vee cond_k) \Rightarrow cond_1 \wedge (v = expr_1) \vee \ldots \vee cond_k \wedge (v = expr_k)\big) \Longleftrightarrow$$
$$\mathbf{GX}\Big(\neg cond_1 \wedge \ldots \wedge \neg cond_k \vee \big(cond_1 \wedge (v = expr_1) \vee \ldots \vee cond_k \wedge (v = expr_k)\big)\Big).$$

Отнесём выражение в скобках к каждому условию вида $cond_i$:

$$\mathbf{GX}\Big(\big(\neg cond_1 \vee cond_1 \wedge (v = expr_1) \vee \ldots \vee cond_k \wedge (v = expr_k)\big) \wedge \ldots \wedge$$
$$\big(\neg cond_k \vee cond_1 \wedge (v = expr_1) \vee \ldots \vee cond_k \wedge (v = expr_k)\big)\Big).$$

Отсюда следует первый конъюнкт:

$$\mathbf{GX}\big(\neg cond_1 \vee cond_1 \wedge (v = expr_1) \vee \ldots \vee cond_k \wedge (v = expr_k)\big) \Longleftrightarrow$$
$$\mathbf{GX}\big(cond_1 \Rightarrow cond_1 \wedge (v = expr_1) \vee \ldots \vee cond_k \wedge (v = expr_k)\big). \tag{8}$$

Из условия ортогональности (4) следует:

$$\mathbf{GX}\big(cond_1 \Rightarrow \neg cond_2 \wedge \ldots \wedge \neg cond_k\big). \tag{9}$$

Объединим формулы (8) и (9), получим:

$$\mathbf{GX}\Big(cond_1 \Rightarrow \big(cond_1 \wedge (v = expr_1) \vee \ldots \vee cond_k \wedge (v = expr_k)\big) \wedge \big(\neg cond_2 \wedge \ldots \wedge \neg cond_k\big)\Big).$$

Применим выражение в правой скобке к первому дизъюнкту левой скобки:

$$\mathbf{GX}\Big(cond_1 \Rightarrow cond_1 \wedge (v = expr_1) \wedge \neg cond_2 \wedge \ldots \wedge \neg cond_k \vee$$
$$\big(cond_2 \wedge (v = expr_2) \vee \ldots \vee cond_k \wedge (v = expr_k)\big) \wedge \big(\neg cond_2 \wedge \ldots \wedge \neg cond_k\big)\Big).$$

Вторая половина формулы является тождественно ложной. Сократим выражение:

$$\begin{aligned}&\mathbf{GX}\big(cond_1 \Rightarrow (v = expr_1) \wedge \neg cond_2 \wedge \ldots \wedge \neg cond_k\big) \Longleftrightarrow\\ &\mathbf{GX}\Big(\big(cond_1 \Rightarrow (v = expr_1)\big) \wedge \big(cond_1 \Rightarrow \neg cond_2 \wedge \ldots \wedge \neg cond_k\big)\Big) \vdash\\ &\mathbf{GX}\big(cond_1 \Rightarrow (v = expr_1)\big).\end{aligned} \tag{10}$$

Аналогичный вывод имеют формулы этого вида для остальных условий $cond_i$ при $i = 2, \ldots, k$. Взяв в конъюнкцию все эти формулы для всех условий $cond_i$, получим (5). Таким образом, мы доказали справедливость вывода (4), (1), (2) $\vdash$ (5).

Докажем вторую часть леммы: Применим закон контрапозиции к (1), получим:

$$\begin{aligned}&\mathbf{GX}\Big(\neg\big(cond_1 \wedge (v = expr_1) \vee \ldots \vee cond_k \wedge (v = expr_k)\big) \Rightarrow (v = \_v)\Big) \Longleftrightarrow\\ &\mathbf{GX}\Big(\neg\big(cond_1 \wedge (v = expr_1)\big) \wedge \ldots \wedge \neg\big(cond_k \wedge (v = expr_k)\big) \Rightarrow (v = \_v)\Big) \Longleftrightarrow\\ &\mathbf{GX}\Big(\big(\neg cond_1 \vee \neg(v = expr_1)\big) \wedge \ldots \wedge \big(\neg cond_k \vee \neg(v = expr_k)\big) \Rightarrow (v = \_v)\Big).\end{aligned}$$

После раскрытия скобок получится некоторая дизъюнктивная форма:

$$\mathbf{GX}\Big(\big(\neg cond_1 \wedge \ldots \wedge \neg cond_k \vee \ldots \vee \neg(v = expr_1) \wedge \ldots \wedge \neg(v = expr_k)\big) \Rightarrow (v = \_v)\Big).$$

Нас интересует первый дизъюнкт $\neg cond_1 \wedge \ldots \wedge \neg cond_k$, а всё остальное не имеет значения для доказательства, поэтому заменяется абстрактным выражением A. В итоге получим:

$$\begin{aligned}&\mathbf{GX}\Big(\big(\neg cond_1 \wedge \ldots \wedge \neg cond_k \vee A\big) \Rightarrow (v = \_v)\Big) \Longleftrightarrow\\ &\mathbf{GX}\Big(\big(\neg cond_1 \wedge \ldots \wedge \neg cond_k \Rightarrow (v = \_v)\big) \wedge \big(A \Rightarrow (v = \_v)\big)\Big).\end{aligned}$$

Отсюда следует (6). Таким образом, получили, что (1) $\vdash$ (6). Лемма 1 доказана. □

Лемма 2. *Из справедливости императивной LTL-спецификации следует справедливость декларативной LTL-спецификация, т. е. при условиях* (3) *и* (4) *имеем* (5), (6) $\vdash$ (1), (2).

Доказательство. Сначала докажем, что (5), (6) $\vdash$ (1). Преобразуем (5):

$$\mathbf{GX}\big(cond_1 \Rightarrow cond_1 \wedge (v = expr_1)\big) \wedge \ldots \wedge \mathbf{GX}\big(cond_k \Rightarrow cond_k \wedge (v = expr_k)\big). \tag{11}$$

Применим закон контрапозиции к (6), получим:

$$\begin{aligned}&\mathbf{GX}\big(\neg(v = \_v) \Rightarrow cond_1 \vee \ldots \vee cond_k\big) \Longleftrightarrow\\ &\mathbf{GX}\Big(\big(\neg(v = \_v) \Rightarrow cond_1\big) \vee \ldots \vee \big(\neg(v = \_v) \Rightarrow cond_k\big)\Big).\end{aligned} \tag{12}$$

Объединим (12) и (11):

$$\begin{aligned}&\mathbf{GX}\Big(\big[\big(\neg(v = \_v) \Rightarrow cond_1\big) \vee \ldots \vee \big(\neg(v = \_v) \Rightarrow cond_k\big)\big] \wedge\\ &\big[\big(cond_1 \Rightarrow cond_1 \wedge (v = expr_1)\big) \wedge \ldots \wedge \big(cond_k \Rightarrow cond_k \wedge (v = expr_k)\big)\big]\Big).\end{aligned}$$

Раскроем первые квадратные скобки:

$$\mathbf{GX}\Big(\big(\neg(v = \_v) \Rightarrow cond_1\big) \wedge \big[\big(cond_1 \Rightarrow cond_1 \wedge (v = expr_1)\big) \wedge \ldots \wedge \big(cond_k \Rightarrow cond_k \wedge (v = expr_k)\big)\big] \vee \ldots \vee$$
$$\big(\neg(v = \_v) \Rightarrow cond_k\big) \wedge \big[\big(cond_1 \Rightarrow cond_1 \wedge (v = expr_1)\big) \wedge \ldots \wedge \big(cond_k \Rightarrow cond_k \wedge (v = expr_k)\big)\big]\Big) \vdash$$
$$\mathbf{GX}\Big(\big(\neg(v = \_v) \Rightarrow cond_1\big) \wedge \big(cond_1 \Rightarrow cond_1 \wedge (v = expr_1)\big) \vee \ldots \vee$$
$$\big(\neg(v = \_v) \Rightarrow cond_k\big) \wedge \big(cond_k \Rightarrow cond_k \wedge (v = expr_k)\big)\Big) \vdash$$
$$\mathbf{GX}\big(\neg(v = \_v) \Rightarrow cond_1 \wedge (v = expr_1) \vee \ldots \vee \neg(v = \_v) \Rightarrow cond_k \wedge (v = expr_k)\big).$$

Из последней формулы получаем (1). Справедливость вывода (5), (6) $\vdash$ (1) доказана.
Теперь докажем, что (3), (5) $\vdash$ (2). Объединим (5) и (3), получим:

$$\mathbf{GX}\Big(\big(cond_1 \Rightarrow (v = expr_1)\big) \wedge \ldots \wedge \big(cond_k \Rightarrow (v = expr_k)\big) \wedge \ldots \wedge$$
$$\big(cond_1 \Rightarrow \neg(\_v = expr_1)\big) \wedge \ldots \wedge \big(cond_k \Rightarrow \neg(\_v = expr_k)\big)\Big) \Longleftrightarrow$$
$$\mathbf{GX}\Big(\big(cond_1 \Rightarrow (v = expr_1) \wedge \neg(\_v = expr_1)\big) \wedge \ldots \wedge \big(cond_k \Rightarrow (v = expr_k) \wedge \neg(\_v = expr_k)\big)\Big) \vdash$$
$$\mathbf{GX}\Big(\big(cond_1 \Rightarrow \neg(v = \_v)\big) \wedge \ldots \wedge \big(cond_k \Rightarrow \neg(v = \_v)\big)\Big).$$

Применим закон контрапозиции к последней формуле:

$$\mathbf{GX}\Big(\big((v = \_v) \Rightarrow \neg cond_1\big) \wedge \ldots \wedge \big((v = \_v) \Rightarrow \neg cond_k\big)\Big) \Longleftrightarrow$$
$$\mathbf{GX}\big((v = \_v) \Rightarrow \neg cond_1 \wedge \ldots \wedge \neg cond_k\big) \Longleftrightarrow \mathbf{GX}\big((v = \_v) \Rightarrow \neg(cond_1 \vee \ldots \vee cond_k)\big).$$

Получили формулу (2). Следовательно, вывод (3), (5) $\vdash$ (2) является справедливым.
Лемма 2 доказана. □

Теорема 1 (Об эквивалентности спецификаций)**.** *Декларативная* (1), (2) *и императивная* (5), (6) *LTL-спецификации эквивалентны при соблюдении условий изменчивости* (3) *и ортогональности* (4)*.*

Доказательство. Следует из Лемм 1 и 2. □

5. Выразительность декларативной LTL-спецификации

Оценим выразительные возможности декларативной LTL-спецификации в смысле Тьюринг-полноты. Для доказательства теоремы о Тьюринг-полноте декларативной LTL-спецификации в качестве формальной модели алгоритма выберем счётчиковую машину Минского [35—37]. Она эквивалентна по вычислительным возможностям машине Тьюринга, но представляется более удобной для описания поведения программ, так как имеет более наглядный программный вид, т. е. вид компьютерной программы, написанной на языке высокого уровня.

Счётчиковая машина Минского M представляет собой набор $\langle Q, q_1, q_n, Y, \Delta\rangle$, где $Q = \{q_1, \ldots, q_n\}$ — конечное непустое множество управляющих состояний машины; $q_1 \in Q$ — начальное управляющее состояние; $q_n \in Q$ — заключительное, или финальное, управляющее состояние; $Y = \{y_1, \ldots, y_m\}$ — конечное непустое множество счётчиков, которые могут принимать значения из $\mathbb{N}_0$; $\Delta = \{\delta_1, \ldots, \delta_{n-1}\}$ — набор правил переходов по управляющим состояниям машины; δ_i — правило переходов для управляющего состояния q_i, где $i = 1, \ldots, n-1$.

Состояния q_i, $1 \leqslant i \leqslant n-1$, подразделяются на два типа. Управляющие состояния первого типа имеют правила переходов вида:

$$(\delta_i)\ q_i\colon\ y := y + 1;\ \textbf{goto}\ q_k, \tag{13}$$

где $y \in Y$, $q_k \in Q$. Для управляющих состояний второго типа определяются правила переходов вида:

$$(\delta_i)\ q_i\colon \textbf{if}\ y > 0\ \textbf{then}\ (y := y - 1;\ \textbf{goto}\ q_k)\ \textbf{else goto}\ q_l, \tag{14}$$

где $q_k, q_l \in Q$. Для финального состояния q_n правил переходов не предусмотрено. При попадании в финальное состояние q_n машина завершает свою работу.

Состояние (*конфигурация*) счётчиковой машины — это набор $(q, c_1, \ldots, c_m)$, где $q \in Q$, $c_i \in \mathbb{N}_0$, $i = 1, \ldots, m$; здесь $c_1, \ldots, c_m$ являются значениями соответствующих счётчиков $y_1, \ldots, y_m$.

Исполнением машины Минского называется последовательность состояний $s_0\, s_1\, s_2\, \ldots$, индуктивно определяемая в соответствии с правилами переходов. Счётчиковая машина имеет одно исполнение из начального состояния s_0, так как для каждого управляющего состояния предусмотрено не более одного правила переходов.

Машина, получив на вход некоторый набор значений счетчиков, стартует из управляющего состояния q_1 и либо останавливается в управляющем состоянии q_n с выходным набором значений счетчиков, либо зацикливается, реализуя тем самым частичную числовую функцию.

Рассмотрим в качестве примера трехсчётчиковую машину Минского *3cM*, реализующую функцию возведения числа n в квадрат, где $n \in \mathbb{N}_0$. Правила переходов по управляющим состояниям машины *3cM* представлены ниже [36]:

$$\begin{aligned}
&(\delta_1)\ q_1\colon \textbf{if}\ a > 0\ \textbf{then}\ (a := a - 1;\ \textbf{goto}\ q_2)\ \textbf{else goto}\ q_8;\\
&(\delta_2)\ q_2\colon\ c := c + 1;\ \textbf{goto}\ q_3;\\
&(\delta_3)\ q_3\colon \textbf{if}\ a > 0\ \textbf{then}\ (a := a - 1;\ \textbf{goto}\ q_4)\ \textbf{else goto}\ q_6;\\
&(\delta_4)\ q_4\colon\ b := b + 1;\ \textbf{goto}\ q_5;\\
&(\delta_5)\ q_5\colon\ c := c + 1;\ \textbf{goto}\ q_2;\\
&(\delta_6)\ q_6\colon \textbf{if}\ b > 0\ \textbf{then}\ (b := b - 1;\ \textbf{goto}\ q_7)\ \textbf{else goto}\ q_1;\\
&(\delta_7)\ q_7\colon\ a := a + 1;\ \textbf{goto}\ q_6.
\end{aligned} \tag{15}$$

Счётчиковая машина *3cM* имеет восемь управляющих состояний $q_1, \ldots, q_8$, где q_1 является начальным управляющим состоянием, а q_8 — финальным. Множество счетчиков $Y = \{a, b, c\}$. Предполагается, что в начальном состоянии счетчик a получает значение n, а начальные значения двух других счетчиков b и c равны нулю. В финальном состоянии результат вычисления будет содержаться в счетчике c при нулевых значениях счетчиков a и b.

Правила переходов по управляющим состояниям машины *3cM* представлены в графическом виде на рис. 6, где для счётчика $y \in Y$ обозначение «$y+$» соответствует его увеличению на единицу, а «$y-$» используется для обозначения условного вычитания единицы с переходом в другое управляющее состояние по правосторонней стрелке в случае нулевого значения счётчика y.

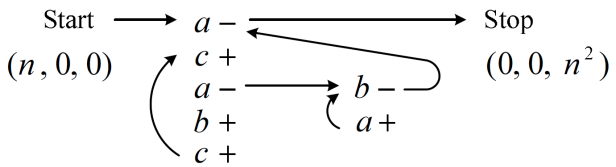


Fig. 6. Graphical representation of the counter machine *3cM* of squaring a number

Рис. 6. Графическое представление счётчиковой машины *3cM* возведения числа в квадрат

Теорема 2 (О Тьюринг-полноте LTL-спецификации)**.** *Для любой счётчиковой машины Минского её поведение, т. е. множество всех возможных исполнений, может быть задано с помощью декларативной LTL-спецификации.*

Доказательство. В качестве доказательства приведём общую процедуру построения декларативной LTL-спецификации поведения программы, которая моделирует работу произвольной счётчиковой машины Минского.

Пусть имеется некоторая m-счётчиковая машина Минского, представленная в виде набора правил переходов. Программа будет содержать основные $q, y_1, \ldots, y_m$ и вспомогательные $\_q, \_y_1, \ldots, \_y_m$ переменные. Переменная q предназначена для хранения номера текущего управляющего состояния, переменные $y_1, \ldots, y_m$ — для хранения значений счётчиков машины. Таким образом, вектор $(q, y_1, \ldots, y_m, \_q, \_y_1, \ldots, \_y_m)$ описывает текущее состояние машины. Все переменные программы принимают значения из $\mathbb{N}_0$. Любому переходу (согласно правилам переходов) машины Минского соответствует переход между состояниями программы. Если для текущего состояния машины ни одно из правил переходов не выполняется, то машина остаётся в прежнем состоянии. При этом в программе происходит переход по петле в то же самое программное состояние.

Формула инициализации программы, описывающая множество начальных состояний, будет иметь следующий вид:

$$q = 1 \wedge y_1 \geqslant 0 \wedge \cdots \wedge y_m \geqslant 0 \wedge \_q = q \wedge \_y_1 = y_1 \wedge \cdots \wedge \_y_m = y_m.$$

Далее, выполним LTL-формализацию правил переходов для управляющих состояний $q_i \in Q$ первого типа следующим образом:

$$\mathbf{GX}\big((\_q = i) \Rightarrow (y = \_y + 1) \wedge (q = k)\big). \tag{16}$$

LTL-формализация правил переходов для состояний $q_i \in Q$ второго типа будет иметь вид:

$$\mathbf{GX}\Big(\big[(\_q = i) \wedge (\_y > 0) \Rightarrow (y = \_y - 1) \wedge (q = k)\big] \wedge \big[(\_q = i) \wedge \neg(\_y > 0) \Rightarrow (q = l)\big]\Big). \tag{17}$$

Каждую формулу вида (16) разобьём на две LTL-формулы:

$$\begin{aligned} &\mathbf{GX}\big((\_q = i) \Rightarrow (y = \_y + 1)\big), \\ &\mathbf{GX}\big((\_q = i) \Rightarrow (q = k)\big). \end{aligned} \tag{18}$$

Фрагментация формул (17) будет такой:

$$\begin{aligned} &\mathbf{GX}\big((\_q = i) \wedge (\_y > 0) \Rightarrow (y = \_y - 1)\big), \\ &\mathbf{GX}\big((\_q = i) \wedge (\_y > 0) \Rightarrow (q = k)\big), \\ &\mathbf{GX}\big((\_q = i) \wedge \neg(\_y > 0) \Rightarrow (q = l)\big). \end{aligned} \tag{19}$$

Все счётчики имеют одну и ту же процедуру построения спецификации. Покажем, как построить спецификацию для произвольного счётчика $y \in Y$. Сгруппируем все формулы, в которых фигурирует счётчик y, получим:

$$\begin{aligned} &\mathbf{GX}\big((\_q = e_{11}) \Rightarrow (y = \_y + 1)\big), \ldots, \\ &\mathbf{GX}\big((\_q = e_{1s}) \Rightarrow (y = \_y + 1)\big), \\ &\mathbf{GX}\big((\_q = e_{21}) \wedge (\_y > 0) \Rightarrow (y = \_y - 1)\big), \ldots, \\ &\mathbf{GX}\big((\_q = e_{2p}) \wedge (\_y > 0) \Rightarrow (y = \_y - 1)\big). \end{aligned} \tag{20}$$

Конъюнктивно объединим формулы в группе (20), а также добавим формулу, описывающую отсутствие изменений значения переменной. Получим императивную LTL-спецификацию для y:

$$\begin{aligned}
&\mathbf{GX}\big((_q = e_{11}) \Rightarrow (y = _y + 1)\big) \wedge \ldots \wedge \\
&\mathbf{GX}\big((_q = e_{1s}) \Rightarrow (y = _y + 1)\big) \wedge \\
&\mathbf{GX}\big((_q = e_{21}) \wedge (_y > 0) \Rightarrow (y = _y - 1)\big) \wedge \ldots \wedge \\
&\mathbf{GX}\big((_q = e_{2p}) \wedge (_y > 0) \Rightarrow (y = _y - 1)\big), \\
&\mathbf{GX}\Big(\neg\big(_q = e_{11}\big) \wedge \ldots \wedge \neg\big(_q = e_{1s}\big) \wedge \\
&\qquad \neg\big((_q = e_{21}) \wedge (_y > 0)\big) \wedge \ldots \wedge \neg\big((_q = e_{2p}) \wedge (_y > 0)\big) \Rightarrow (y = _y)\Big).
\end{aligned} \tag{21}$$

Также покажем, как построить спецификацию для переменной q. Сгруппируем все формулы, в которых она фигурирует, получим:

$$\begin{aligned}
&\mathbf{GX}\big((_q = d_{11}) \Rightarrow (q = g_{11})\big), \ldots, \\
&\mathbf{GX}\big((_q = d_{1h}) \Rightarrow (q = g_{1h})\big), \\
&\mathbf{GX}\big((_q = d_{21}) \wedge (_y > 0) \Rightarrow (q = g_{21})\big), \ldots, \\
&\mathbf{GX}\big((_q = d_{2t}) \wedge (_y > 0) \Rightarrow (q = g_{2t})\big), \\
&\mathbf{GX}\big((_q = d_{31}) \wedge \neg(_y > 0) \Rightarrow (q = g_{31})\big), \ldots, \\
&\mathbf{GX}\big((_q = d_{3z}) \wedge \neg(_y > 0) \Rightarrow (q = g_{3z})\big).
\end{aligned} \tag{22}$$

Удалим из группы (22) формулы, в которых $d_i = g_i$, где $i = 11\ldots1h, 21\ldots2t, 31\ldots3z$. Оставшиеся формулы конъюнктивно объединим, а также добавим формулу, описывающую отсутствие изменений значения переменной. Получим императивную LTL-спецификацию для q:

$$\begin{aligned}
&\mathbf{GX}\big((_q = d_{11}) \Rightarrow (q = g_{11})\big) \wedge \ldots \wedge \\
&\mathbf{GX}\big((_q = d_{1h}) \Rightarrow (q = g_{1h})\big) \wedge \\
&\mathbf{GX}\big((_q = d_{21}) \wedge (_y > 0) \Rightarrow (q = g_{21})\big) \wedge \ldots \wedge \\
&\mathbf{GX}\big((_q = d_{2t}) \wedge (_y > 0) \Rightarrow (q = g_{2t})\big) \wedge \\
&\mathbf{GX}\big((_q = d_{31}) \wedge \neg(_y > 0) \Rightarrow (q = g_{31})\big) \wedge \ldots \wedge \\
&\mathbf{GX}\big((_q = d_{3z}) \wedge \neg(_y > 0) \Rightarrow (q = g_{3z})\big), \\
&\mathbf{GX}\Big(\neg\big(_q = d_{11}\big) \wedge \ldots \wedge \neg\big(_q = d_{1h}\big) \wedge \\
&\qquad \neg\big((_q = d_{21}) \wedge (_y > 0)\big) \wedge \ldots \wedge \neg\big((_q = d_{2t}) \wedge (_y > 0)\big) \wedge \\
&\qquad \neg\big((_q = d_{31}) \wedge \neg(_y > 0)\big) \wedge \ldots \wedge \neg\big((_q = d_{3z}) \wedge \neg(_y > 0)\big) \Rightarrow (q = _q)\Big).
\end{aligned} \tag{23}$$

На основании императивных LTL-спецификаций (21) и (23) построим эквивалентные декларативные LTL-спецификации для y и q соответственно:

$$\begin{aligned}
\mathbf{GX}\big(&\neg(y = _y) \Rightarrow (_q = e_{11}) \wedge (y = _y + 1) \vee \ldots \vee (_q = e_{1s}) \wedge (y = _y + 1) \vee \\
&(_q = e_{21}) \wedge (_y > 0) \wedge (y = _y - 1) \vee \ldots \vee (_q = e_{2p}) \wedge (_y > 0) \wedge (y = _y - 1)\big), \\
\mathbf{GX}\big(&(y = _y) \Rightarrow \neg\big((_q = e_{11}) \vee \ldots \vee (_q = e_{1s}) \vee \\
&(_q = e_{21}) \wedge (_y > 0) \vee \ldots \vee (_q = e_{2p}) \wedge (_y > 0)\big)\big).
\end{aligned} \tag{24}$$

$$\begin{aligned}
&\mathbf{GX}\big(\neg(q = _q) \Rightarrow (_q = d_{11}) \wedge (q = g_{11}) \vee \ldots \vee (_q = d_{1h}) \wedge (q = g_{1h}) \vee \\
&\qquad (_q = d_{21}) \wedge (_y > 0) \wedge (q = g_{21}) \vee \ldots \vee (_q = d_{2t}) \wedge (_y > 0) \wedge (q = g_{2t}) \vee \\
&\qquad (_q = d_{31}) \wedge \neg(_y > 0) \wedge (q = g_{31}) \vee \ldots \vee (_q = d_{3z}) \wedge \neg(_y > 0) \wedge (q = g_{3z})\big), \\
&\mathbf{GX}\big((q = _q) \Rightarrow \neg\big((_q = d_{11}) \vee \ldots \vee (_q = d_{1h}) \vee \\
&\qquad (_q = d_{21}) \wedge (_y > 0) \vee \ldots \vee (_q = d_{2t}) \wedge (_y > 0) \vee \\
&\qquad (_q = d_{31}) \wedge \neg(_y > 0) \vee \ldots \vee (_q = d_{3z}) \wedge \neg(_y > 0)\big)\big).
\end{aligned} \tag{25}$$

Декларативная LTL-спецификация поведения программы, моделирующей работу счётчиковой машины, — формула инициализации и набор формул, описывающих поведение переменной q (25), а также всех переменных $y_1, \ldots, y_m$ в виде (24). Формулы (25) и (24) удовлетворяют условиям изменчивости (3) и ортогональности (4). □

Процедура построения декларативной LTL-спецификации φ_{3cM} счётчиковой машины *3cM* представлена в приложении. Сама спецификация φ_{3cM} имеет следующий вид:

$$\mathrm{S0}: \quad (q = 1) \wedge (a \geqslant 0) \wedge (b \geqslant 0) \wedge (c \geqslant 0) \wedge (\_q = q) \wedge (\_a = a) \wedge (\_b = b) \wedge (\_c = c);$$

$$\begin{aligned}
\mathrm{S}a: \quad &\mathbf{GX}\big(\neg(a = _a) \Rightarrow (_q = 1) \wedge (_a > 0) \wedge (a = _a - 1) \vee \\
&\qquad (_q = 3) \wedge (_a > 0) \wedge (a = _a - 1) \vee \\
&\qquad (_q = 7) \qquad\quad \wedge (a = _a + 1) \quad \big) \wedge \\
&\mathbf{GX}\big((a = _a) \Rightarrow \neg\big((_q = 1) \wedge (_a > 0) \vee (_q = 3) \wedge (_a > 0) \vee (_q = 7)\big)\big);
\end{aligned}$$

$$\begin{aligned}
\mathrm{S}b: \quad &\mathbf{GX}\big(\neg(b = _b) \Rightarrow (_q = 4) \qquad\quad \wedge (b = _b + 1) \vee \\
&\qquad (_q = 6) \wedge (_b > 0) \wedge (b = _b - 1) \quad \big) \wedge \\
&\mathbf{GX}\big((b = _b) \Rightarrow \neg\big((_q = 4) \vee (_q = 6) \wedge (_b > 0)\big)\big);
\end{aligned}$$

$$\begin{aligned}
\mathrm{S}c: \quad &\mathbf{GX}\big(\neg(c = _c) \Rightarrow (_q = 2) \wedge (c = _c + 1) \vee \\
&\qquad (_q = 5) \wedge (c = _c + 1) \quad \big) \wedge \\
&\mathbf{GX}\big((c = _c) \Rightarrow \neg\big((_q = 2) \vee (_q = 5)\big)\big);
\end{aligned}$$

$$\begin{aligned}
\mathrm{S}q: \quad &\mathbf{GX}\big(\neg(q = _q) \Rightarrow (_q = 1) \wedge \quad (_a > 0) \wedge (q = 2) \vee \\
&\qquad (_q = 1) \wedge \neg(_a > 0) \wedge (q = 8) \vee \\
&\qquad (_q = 2) \qquad\quad \wedge (q = 3) \vee \\
&\qquad (_q = 3) \wedge \quad (_a > 0) \wedge (q = 4) \vee \\
&\qquad (_q = 3) \wedge \neg(_a > 0) \wedge (q = 6) \vee \\
&\qquad (_q = 4) \qquad\quad \wedge (q = 5) \vee \\
&\qquad (_q = 5) \qquad\quad \wedge (q = 2) \vee \\
&\qquad (_q = 6) \wedge \quad (_b > 0) \wedge (q = 7) \vee \\
&\qquad (_q = 6) \wedge \neg(_b > 0) \wedge (q = 1) \vee \\
&\qquad (_q = 7) \qquad\quad \wedge (q = 6) \quad \big) \wedge \\
&\mathbf{GX}\big((q = _q) \Rightarrow \neg\big((_q = 1) \wedge (_a > 0) \vee (_q = 1) \wedge \neg(_a > 0) \vee (_q = 2) \vee \\
&\qquad (_q = 3) \wedge (_a > 0) \vee (_q = 3) \wedge \neg(_a > 0) \vee (_q = 4) \vee (_q = 5) \vee \\
&\qquad (_q = 6) \wedge (_b > 0) \vee (_q = 6) \wedge \neg(_b > 0) \vee (_q = 7)\big)\big).
\end{aligned}$$

Заметим, что последнюю формулу LTL-спецификации можно упростить следующим образом:

$$\mathbf{GX}\big((q = \_q) \Rightarrow \neg\big((\_q = 1) \vee (\_q = 2) \vee (\_q = 3) \vee (\_q = 4) \vee (\_q = 5) \vee (\_q = 6) \vee (\_q = 7)\big)\big).$$

Счётчиковая машина *3cM* имеет всего восемь управляющих состояний $q_1, \ldots, q_8$, поэтому переменная $q = 1, \ldots, 8$. Из упрощённой формулы видно, что если $\_q = 1, \ldots, 7$, то значение q изменяется.

Это говорит о том, что машина *3cM* всегда переходит в другое управляющее состояние. Исключением является только финальное управляющее состояние q_8, в котором машина останавливается.

6. Верификация декларативной LTL-спецификации

6.1. Постановка задачи

Исходная задача верификации декларативной LTL-спецификации

$$[[\varphi]]_{pLTS} \vDash \psi \tag{26}$$

является задачей проверки модели, т. е. проверки того, является ли интерпретация $[[\varphi]]_{pLTS}$ моделью формулы ψ, где φ — декларативная LTL-спецификация, задающая множество путей в псевдополной системе переходов *pLTS*, а ψ — проверяемое LTL-свойство.

Мы намерены решать данную задачу с помощью программного средства nuXmv — инструмента символьной *проверки модели* (model checking) [12—15], который позволяет работать с моделями с конечным и бесконечным числом состояний.

Важно отметить, что задача (26) в общем случае неразрешима, т. е. не существует алгоритма проверки выполнимости формулы ψ на интерпретации $[[\varphi]]_{pLTS}$. Если бы такая процедура существовала, то можно было бы выяснить для произвольной счётчиковой машины *cM* справедливость $[[\varphi_{cM}]]_{pLTS} \vDash \mathbf{F}\,\mathbf{G}\,(q = q_n)$, где φ_{cM} — декларативная LTL-спецификация поведения этой счётчиковой машины, а q_n — её финальное состояние. И мы бы имели решение проблемы *тотальности* для счётчиковых машин Минского, т. е. имели бы алгоритм, который для любой счётчиковой машины определял, будет ли она останавливаться при всех возможных начальных значениях счётчиков. Но известно, что проблема тотальности не является даже частично разрешимой уже для двухсчётиковых машин Минского [37].

Для разрешимости задачи (26) потребуем, чтобы множество состояний $S = (D_1 \times \ldots \times D_n)^2$ псевдополной системы переходов $pLTS = \langle S, S_0, R'_c, P, L\rangle$ было конечным. Для этого необходимо ограничить область значений D_i, где $i = 1, \ldots, n$, каждой переменной из $V = \{v_1, \ldots, v_n\}$. В этом случае конечная система переходов *pLTS* становится *структурой Крипке* [12—15].

Для корректного описания поведения конечных программ декларативная LTL-спецификация должна удовлетворять дополнительному условию *ограниченности* для каждой переменной $v_i \in V$:

$$\mathbf{GX}\big(cond_1 \Rightarrow (val(expr_1) \in D_i)\big) \wedge \ldots \wedge \mathbf{GX}\big(cond_k \Rightarrow (val(expr_k) \in D_i)\big), \tag{27}$$

т. е. если условие $cond_j$ истинно ($j = 1, \ldots, k$), то выражение $expr_j$ возвращает значение $val(expr_j)$, которое принадлежит области значений данной переменной. А также потребуем, чтобы в формуле инициализации содержались выражения, обеспечивающие попадание начального значения каждой переменной в допустимую область значений.

Декларативная LTL-спецификация с ограничением (27) позволяет, например, задавать поведение *конечной* счётчиковой машины Минского, которая представляет собой счётчиковую машину с ограничениями на значения счётчиков и модифицированным правилом переходов для управляющих состояний первого типа. Ограничения вводятся через отображение $\mathrm{bnd} : Y \to \mathbb{N}_0$, устанавливающее верхнее предельное значение $\mathrm{bnd}y \in \mathbb{N}_0$ для каждого счётчика $y \in Y$. А модифицированное правило переходов для управляющих состояний q_i первого типа имеет вид:

$$(\delta_i)\ q_i\text{: } \mathbf{if}\ y < \mathrm{bnd}y\ \mathbf{then}\ (y := y + 1;\ \mathbf{goto}\ q_k)\ \mathbf{else\ goto}\ q_l.$$

В качестве примера в следующем параграфе данного раздела будет рассмотрена конечная счётчиковая машина *3cM′* возведения числа в квадрат.

Для решения задачи проверки модели (26) с помощью инструмента nuXmv необходимо задать множество путей $[[\varphi]]_{pLTS}$ в некотором приемлемом для этого инструмента виде. Так как φ и ψ

уже являются LTL-формулами, которые поддерживает nuXmv, то самым простым решением будет не преобразовывать их в другую форму, а оставить как есть. При этом в соответствии с определением логики LTL задача (26) может быть сведена к следующей задаче:

$$pLTS \vDash (\varphi \Rightarrow \psi). \tag{28}$$

Здесь проверяется выполнимость импликации $\varphi \Rightarrow \psi$ на псевдополной системе переходов $pLTS$. Верификатор в данном случае будет обходить множество всех возможных путей в $pLTS$ и проверять истинность свойства ψ только на тех путях, на которых истинна спецификация поведения φ.

В рамках такой постановки задачи верификации потребуется задать псевдополную систему переходов $pLTS$ на входном языке верификатора nuXmv. Для этого нужно объявить два набора переменных $V = \{v_1, \ldots, v_n\}$ и $\_V = \{\_v_1, \ldots, \_v_n\}$, а затем описать поведение каждой вспомогательной переменной $\_v \in \_V$ с помощью конструкции **next** $(\_v) := v$, обеспечивающей сохранение в $\_v$ предыдущего значения соответствующей переменной $v \in V$. По умолчанию в nuXmv поведение переменных из V является абсолютно недетерминированным. После задания $pLTS$ можно запускать инструмент nuXmv для проверки выполнимости формулы $\varphi \Rightarrow \psi$. Если эта формула является истинной, значит, спецификация поведения φ удовлетворяет свойству ψ, в противном случае спецификация φ не соответствует свойству ψ.

6.2. Конечная счётчиковая машина возведения числа в квадрат

Построим конечную счётчиковую машину $3cM'$ возведения числа в квадрат при ограничениях $\mathrm{bnd}a$, $\mathrm{bnd}b$ и $\mathrm{bnd}c$ на счётчики a, b и c соответственно:

(δ_1) q_1: **if** $a > 0$ **then** $(a := a - 1;$ **goto** $q_2)$ **else goto** q_8;
(δ_2) q_2: **if** $c < \mathrm{bnd}c$ **then** $(c := c + 1;$ **goto** $q_3)$ **else goto** q_3;
(δ_3) q_3: **if** $a > 0$ **then** $(a := a - 1;$ **goto** $q_4)$ **else goto** q_6;
(δ_4) q_4: **if** $b < \mathrm{bnd}b$ **then** $(b := b + 1;$ **goto** $q_5)$ **else goto** q_5;
(δ_5) q_5: **if** $c < \mathrm{bnd}c$ **then** $(c := c + 1;$ **goto** $q_2)$ **else goto** q_2;
(δ_6) q_6: **if** $b > 0$ **then** $(b := b - 1;$ **goto** $q_7)$ **else goto** q_1;
(δ_7) q_7: **if** $a < \mathrm{bnd}a$ **then** $(a := a + 1;$ **goto** $q_6)$ **else goto** q_6.

Приведём декларативную LTL-спецификацию поведения конечной счётчиковой машины $3cM'$ в той её части, которая отличается от декларативной спецификации исходной машины $3cM$ возведения числа в квадрат. Изменятся только формулы S0, Sa, Sb и Sc:

$$\mathrm{S0}: \quad (q = 1) \wedge (a \geqslant 0) \wedge (a \leqslant \mathrm{bnd}a) \wedge (b \geqslant 0) \wedge (b \leqslant \mathrm{bnd}b) \wedge (c \geqslant 0) \wedge (c \leqslant \mathrm{bnd}c) \wedge (\_q = q) \wedge (\_a = a) \wedge (\_b = b) \wedge (\_c = c);$$

$$\begin{aligned}\mathrm{S}a: \quad & \mathbf{GX}\big(\neg(a = \_a) \Rightarrow (\_q = 1) \wedge (\_a > 0) \quad \wedge (a = \_a - 1) \vee \\ & \qquad (\_q = 3) \wedge (\_a > 0) \quad \wedge (a = \_a - 1) \vee \\ & \qquad (\_q = 7) \wedge (\_a < \mathrm{bnd}a) \wedge (a = \_a + 1) \quad \big) \wedge \\ & \mathbf{GX}\big((a = \_a) \Rightarrow \neg\big((\_q = 1) \wedge (\_a > 0) \vee (\_q = 3) \wedge (\_a > 0) \vee (\_q = 7) \wedge (\_a < \mathrm{bnd}a)\big)\big);\end{aligned}$$

$$\begin{aligned}\mathrm{S}b: \quad & \mathbf{GX}\big(\neg(b = \_b) \Rightarrow (\_q = 4) \wedge (\_b < \mathrm{bnd}b) \wedge (b = \_b + 1) \vee \\ & \qquad (\_q = 6) \wedge (\_b > 0) \quad \wedge (b = \_b - 1) \quad \big) \wedge \\ & \mathbf{GX}\big((b = \_b) \Rightarrow \neg\big((\_q = 4) \wedge (\_b < \mathrm{bnd}b) \vee (\_q = 6) \wedge (\_b > 0)\big)\big);\end{aligned}$$

$$\begin{aligned}\mathrm{S}c: \quad & \mathbf{GX}\big(\neg(c = \_c) \Rightarrow (\_q = 2) \wedge (\_c < \mathrm{bnd}c) \wedge (c = \_c + 1) \vee \\ & \qquad (\_q = 5) \wedge (\_c < \mathrm{bnd}c) \wedge (c = \_c + 1) \quad \big) \wedge \\ & \mathbf{GX}\big((c = \_c) \Rightarrow \neg\big((\_q = 2) \wedge (\_c < \mathrm{bnd}c) \vee (\_q = 5) \wedge (\_c < \mathrm{bnd}c)\big)\big).\end{aligned}$$

Формула Sq останется неизменной, так как в каждом правиле переходов первого типа для *3cM′* обе ветки условия ведут в одно и то же управляющее состояние. Поэтому для переменной q формула описания её поведения упрощается до прежнего варианта Sq.

Рассмотрим некоторые LTL-свойства счётчиковой машины *3cM′*, которые обязательно должны выполняться для любого её исполнения.

$$(a = n \wedge b = 0 \wedge c = 0 \wedge n \geqslant 0 \wedge \mathbf{G}(n = \_n)) \Rightarrow \mathbf{G}(\ q = 8 \ \Rightarrow \ c = n^2 \wedge a = 0 \wedge b = 0\ ).$$

Это свойство означает, что если счётчиковая машина *3cM′*, стартуя при $a = n$, $b = 0$ и $c = 0$, оказывается в финальном управляющем состоянии q_8, то значения счётчиков a и b будут равны 0, а счётчик c будет содержать искомый результат n^2.

$$(a = n \wedge b = 0 \wedge c = 0 \wedge n \geqslant 0 \wedge \mathbf{G}(n = \_n)) \Rightarrow \mathbf{G}(\ a + b \leqslant n\ ).$$

Это свойство требует, чтобы суммарное значение счётчиков a и b было ограничено константой n на протяжении всего исполнения счётчиковой машины *3cM′*.

$$(a = n \wedge b = 0 \wedge c = 0 \wedge n \geqslant 0 \wedge \mathbf{G}(n = \_n)) \Rightarrow \mathbf{G}(\ c \leqslant n^2\ ).$$

Требуется, чтобы значение счётчика c на протяжении всего исполнения машины *3cM′* не выходило за пределы n^2.

$$\mathbf{G}\,\mathbf{F}\,(q = 8).$$

Машина *3cM′* из любого управляющего состояния (в том числе и из начального q_1) всегда рано или поздно переходит в финальное управляющее состояние q_8.

$$\mathbf{G}\,(\ q = 8 \ \Rightarrow \ \mathbf{X}(q = 8)\ ).$$

Если счётчиковая машина *3cM′* попадает в финальное управляющее состояние q_8, то она навсегда остаётся в этом состоянии.

$$\mathbf{G}\,(\ q \geqslant 1 \wedge q \leqslant 8\ ).$$

Управляющее состояние q всегда принимает значения из диапазона от 1 до 8.

Теперь рассмотрим LTL-свойство, которое нарушается на некотором исполнении машины *3cM′*.

$$(a = n \wedge b = 0 \wedge c = 0 \wedge n > 0 \wedge \mathbf{G}(n = \_n)) \Rightarrow \mathbf{G}(\ q = 8 \ \Rightarrow \ \neg(c = 2n)\ ).$$

Это свойство утверждает, что в каждом возможном исполнении машины *3cM′* из определённых начальных состояний итоговый результат её работы в управляющем состоянии q_8 отличен от $c = 2n$ при $n > 0$. Контрпример соответствует исполнению машины при $n = 2$.

Заметим, что при спецификации арифметических свойств машины *3cM′*, вычисляющей значение n^2, потребовалось ввести входную переменную n и соответствующую ей вспомогательную переменную $\_n$, которых не было в правилах переходов машины *3cM′*. Для проверки справедливости свойств такого рода необходимо расширить псевдополную систему переходов *pLTS* с помощью введения в её состояния значения этих переменных n и $\_n$. Имитируя входное число, подлежащее возведению в квадрат, переменная n должна сохранять своё значение на протяжении всего процесса вычисления. Это достигается с помощью специальной подформулы $\mathbf{G}(n = \_n)$, включённой в сами свойства. Выражение $\mathbf{G}(n = \_n)$ требует, чтобы текущее значение переменной n всегда было равно своему предыдущему значению, т. е. не изменялось.

7. Разработка и верификация LTL-спецификации программы ПЛК

Реализуем реагирующую систему в виде программы ПЛК на базе счётчиковой машины возведения числа в квадрат *3cM′*. Мотивация данного примера — продемонстрировать некоторое абстрактное поведение автомата управления без привязки к реальному оборудованию и технологическому процессу. Технологическим процессом здесь можно считать процесс вычисления квадрата числа: входное число n выступает в роли исходной заготовки, а n^2 является готовым изделием. Схема системы представлена на рис. 7. ПЛК реагирует на входные сигналы от кнопок (*Buttons*), выдавая выходные сигналы на индикаторы (*Indicators*). Управляющий автомат (*Control automaton*) с состоянием q является системой управления процессом вычисления квадрата числа. Объектом управления является набор из четырёх переменных n, a, b, c. Автомат управления оказывает управляющее воздействие на данный набор переменных путем увеличения (*inc*) и уменьшения (*dec*) их значений. Также управляющий автомат может считывать их значения в любом цикле работы ПЛК.

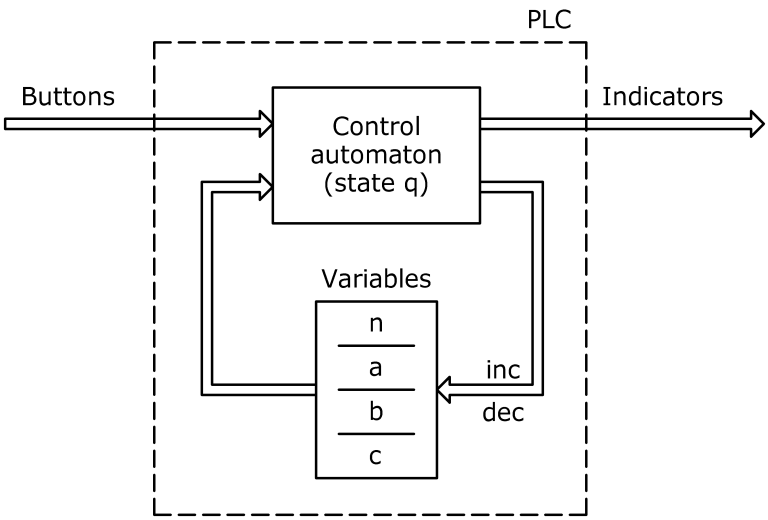


Fig. 7. Reactive system in the form of a PLC program

Рис. 7. Реагирующая система в виде программы ПЛК

Пусть имеется панель управления (см. рис. 8), на которой расположены четыре кнопки «Start», «Reset», «+1» и «-1», девять ламп состояний «q0», . . . , «q8», а также четыре индикатора для отображения значений счётчиков a, b, c и входного числа n. С помощью кнопок «+1» и «-1» происходит выставление числа n, которое будет возводиться в квадрат. Нажатие на кнопку «+1» приводит к увеличению значения n на 1, а нажатие на кнопку «-1» — к уменьшению на 1. Обе кнопки будут срабатывать только тогда, когда управляющий автомат будет находиться в начальном состоянии q_0. Остальные состояния автомата $q_1, \ldots, q_8$ унаследованы от счётчиковой машины *3cM′*. Кнопка «Start» запускает вычислительный процесс для текущего n при условии, что автомат находится в состоянии q_0. При запуске значение переменной n помещается в переменную a, значения b и c равны нулю. Кнопка «Reset» переводит автомат из финального состояния q_8, в которое он попадает после завершения вычисления числа n^2, в стартовое состояние q_0. При этом счётчик c сбрасывается в 0. С помощью ламп «q0», . . . , «q8» отображается текущее состояния управляющего автомата (номер активного состояния хранится в переменной q).

Интерфейс ПЛК представлен на рис. 9. Здесь кнопкам «Start», «Reset», «+1» и «-1» сопоставлены булевы переменные PBStart, PBReset, PBPls и PBMns соответственно. Программные переменные q, a, b, c, n играют роль выходов ПЛК на соответствующие лампы и индикаторы.

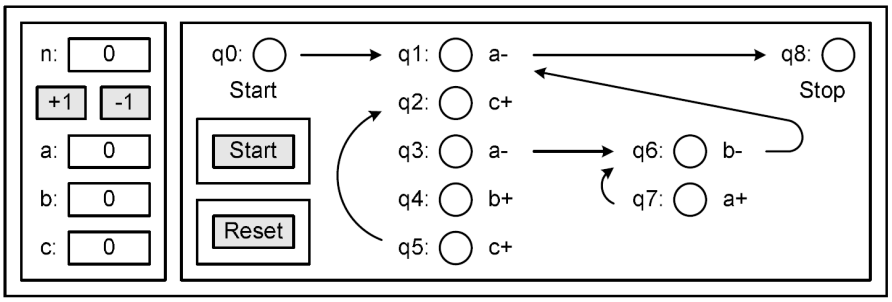


Fig. 8. PLC control panel

Рис. 8. Панель управления ПЛК

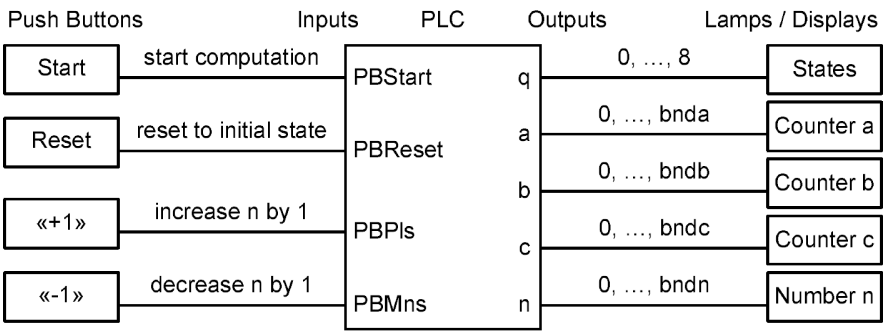


Fig. 9. PLC interface

Рис. 9. Интерфейс ПЛК

Так как верификация выполняется с помощью nuXmv, то спецификацию и её свойства будем записывать на языке этого верификатора. Декларативная LTL-спецификация ПЛК-программы возведения числа в квадрат в синтаксисе nuXmv имеет следующий вид:

```
-- S0:
   q=0 & n=0 & a=0 & b=0 & c=0 & _q=q & _n=n & _a=a & _b=b & _c=c &
   !PBStart & !PBReset & !PBPls & !PBMns & !_PBStart & !_PBReset & !_PBPls & !_PBMns &
-- Sn:
   G X( !(n = _n) ->   _q=0 & _n<bndn  &
                       !_PBPls & PBPls & !PBMns & !PBStart & (n = _n + 1) |
                       _q=0 & _n>0     &
                       !_PBMns & PBMns & !PBPls & !PBStart & (n = _n - 1)  ) &
   G X(  (n = _n) -> !(_q=0 & _n<bndn & !_PBPls & PBPls & !PBMns & !PBStart |
                       _q=0 & _n>0    & !_PBMns & PBMns & !PBPls & !PBStart) ) &
-- Sa:
   G X( !(a = _a) ->   _q=0 & PBStart & !(_a=n) & (a = n)      |
                       _q=7 & _a<bnda           & (a = _a + 1) |
                       _q=1 & _a>0              & (a = _a - 1) |
                       _q=3 & _a>0              & (a = _a - 1) ) &
   G X(  (a = _a) -> !(_q=0 & PBStart & !(_a=n) | _q=7 & _a<bnda |
                       _q=1 & _a>0              | _q=3 & _a>0     ) ) &
-- Sb:
   G X( !(b = _b) ->   _q=4 & _b<bndb & (b = _b + 1) |
                       _q=6 & _b>0    & (b = _b - 1)  ) &
   G X(  (b = _b) -> !(_q=4 & _b<bndb | _q=6 & _b>0)  ) &
-- Sc:
   G X( !(c = _c) ->   _q=2 & _c<bndc        & (c = _c + 1) |
                       _q=5 & _c<bndc        & (c = _c + 1) |
                       _q=8 & _c>0 & PBReset & (c = 0)       ) &
```

```
   G X(  (c = _c) -> !(_q=2 & _c<bndc | _q=5 & _c<bndc | _q=8 & _c>0 & PBReset) ) &
-- Sq:
   G X( !(q = _q) ->   _q=1 &  (_a>0) & (q=2) |
                       _q=1 & !(_a>0) & (q=8) |
                       _q=2           & (q=3) |
                       _q=3 &  (_a>0) & (q=4) |
                       _q=3 & !(_a>0) & (q=6) |
                       _q=4           & (q=5) |
                       _q=5           & (q=2) |
                       _q=6 &  (_b>0) & (q=7) |
                       _q=6 & !(_b>0) & (q=1) |
                       _q=7           & (q=6) |
                       _q=8 & PBReset & (q=0) |
                       _q=0 & PBStart & (q=1)  ) &
   G X(  (q = _q) -> !(_q=1 &  (_a>0) | _q=1 & !(_a>0) | _q=2 | _q=3 & (_a>0) |
                       _q=3 & !(_a>0) | _q=4 | _q=5 | _q=6 & (_b>0) |
                       _q=6 & !(_b>0) | _q=7 | _q=8 & PBReset | _q=0 & PBStart) )
```

На языке верификатора nuXmv символы «&», «|», «!» и «->» означают логические операторы «∧», «∨», «¬» и «⇒» соответственно.

Отметим, что спецификация поведения входных переменных PBStart, PBReset, PBPls и PBMns не производится, так как значения этих переменных могут быть любыми на каждом новом шаге рабочего цикла ПЛК. Ограничения для переменных a, b, c, n задаются с помощью соответствующих констант bnd$a$, bnd$b$, bnd$c$, bnd$n$.

Как можно видеть, спецификация поведения программы записана в виде одной большой LTL-формулы. Обозначим её буквой $\varphi = \mathrm{S0} \wedge \mathrm{S}n \wedge \mathrm{S}a \wedge \mathrm{S}b \wedge \mathrm{S}c \wedge \mathrm{S}q$.

Зададим в синтаксисе nuXmv спецификацию свойств ПЛК-программы возведения числа в квадрат в виде набора LTL-формул P1, . . . , P7:

```
G( q=8 -> c=n*n & a=0 & b=0 )                                    -- P1;
G( a+b <= n )                                                    -- P2;
G( c <= n*n )                                                    -- P3;
G( !(q=0) -> F(q=8) )                                            -- P4;
G( q=0 & X(PBStart) -> F(q=8) )                                  -- P5;
G( q=8 -> X(q=8 | PBReset & q=0 & a=0 & b=0 & c=0) )             -- P6;
G( ((q=2 | q=5) -> c<bndc) & (q=4 -> b<bndb) & (q=7 -> a<bnda) )  -- P7.
```

Свойство P1 утверждает, что всегда в заключительном состоянии $q = 8$ результат вычисления n^2 находится в переменной c, а переменные a и b равны нулю. Свойство P2 — всегда сумма a и b не превышает n. Свойство P3 — значение переменной c в процессе вычисления никогда не превышает n^2. Свойство P4 — если процесс вычисления начался (автомат не находится в начальном состоянии $q = 0$), то он обязательно закончится (в будущем автомат перейдёт в финальное состояние $q = 8$). Свойство P5 — если в начальном состоянии $q = 0$ нажата кнопка «Start», то процесс вычисления будет запущен и завершится в своём финальном состоянии $q = 8$. Свойство P6 — если процесс вычисления достиг финального состояния $q = 8$, то он либо продолжает оставаться в данном состоянии, либо по нажатию кнопки «Reset» переходит в начальное состояние $q = 0$ при $a = 0, b = 0, c = 0$. Свойство P7 проверяет ограничение переменных: в состоянии $q = 2$ и $q = 5$ значение переменной c меньше bnd$c$, в состоянии $q = 4$ значение переменной b меньше bnd$b$, в состоянии $q = 7$ значение переменной a меньше bnd$a$.

Каждое свойство $p \in \{\mathrm{P1}, \ldots, \mathrm{P7}\}$ проверяется по отдельности — выполняется проверка выполнимости $pLTS \vDash (\varphi \Rightarrow p)$, где φ — спецификация поведения программы возведения числа в квадрат.

Опишем псевдополную систему переходов *pLTS* и проверяемую формулу в синтаксисе nuXmv:

```
MODULE main
VAR
    q : 0..8;    _q : 0..8;                  -- State q
    n : 0..15;   _n : 0..15;                 -- Number n
    a : 0..15;   _a : 0..15;                 -- Counter a
    b : 0..15;   _b : 0..15;                 -- Counter b
    c : 0..255;  _c : 0..255;                -- Counter c
    PBStart : boolean;  _PBStart : boolean;  -- Push Button "Start"
    PBReset : boolean;  _PBReset : boolean;  -- Push Button "Reset"
    PBPls   : boolean;  _PBPls   : boolean;  -- Push Button "+1"
    PBMns   : boolean;  _PBMns   : boolean;  -- Push Button "-1"
DEFINE
    bndn := 15; bnda := 15; bndb := 15; bndc := 255;  -- Bounds
ASSIGN -- pLTS
    next(_q) := q;  next(_n) := n;  next(_a) := a;  next(_b) := b;  next(_c) := c;
    next(_PBStart) := PBStart;  next(_PBPls) := PBPls;
    next(_PBReset) := PBReset;  next(_PBMns) := PBMns;
LTLSPEC (Spec -> Prop)
```

В последней строке после ключевого слова LTLSPEC приводится формула, для которой необходимо провести проверку на выполнимость относительно псевдополной системы переходов *pLTS*. В этой строке Spec — это LTL-спецификация φ программы возведения числа в квадрат, а Prop — любое свойство из P1, . . . , P7.

Проверка выполнимости свойств P1, . . . , P7 проводилась на персональном компьютере с процессором Intel Core i5-3570 3.4 ГГц и 8 ГБ оперативной памяти. Свойства P4 и P5 проверялись около двух минут, остальные — несколько секунд.

8. Построение программы ПЛК по LTL-спецификации

По декларативной LTL-спецификации φ из раздела 7 построим код программы возведения числа в квадрат на языке ST. Для этого потребуется перевести декларативную LTL-спецификацию φ в императивную. В свою очередь императивная LTL-спецификация (5), (6) для переменной v из множества $\{v_1, \ldots, v_n\}$ имеет следующую схему построения ST-кода:

IF $cond_1$ **THEN** $v := expr_1$;
ELSIF $cond_2$ **THEN** $v := expr_2$;
. . .
ELSIF $cond_k$ **THEN** $v := expr_k$;
END_IF;

ST-программа, построенная в среде CoDeSys (https://codesys.com) на основе этой схемы, имеет вид:

```
PROGRAM PLC_PRG
VAR_INPUT
    PBStart, PBReset, PBPls, PBMns : BOOL := FALSE;
END_VAR
VAR_OUTPUT
    q, n, a, b, c : BYTE := 0;
END_VAR
```

```
VAR
    _q, _n, _a, _b, _c : BYTE := 0;
    _PBStart, _PBReset, _PBPls, _PBMns : BOOL := FALSE;
END_VAR
VAR CONSTANT
    bndn, bnda, bndb : BYTE := 15;
    bndc : BYTE := 255;
END_VAR
IF    _q=0 AND _n<bndn AND PBPls AND NOT _PBPls AND NOT PBMns AND NOT PBStart THEN n:=_n+1;
ELSIF _q=0 AND _n>0    AND PBMns AND NOT _PBMns AND NOT PBPls AND NOT PBStart THEN n:=_n-1;
END_IF;
IF    _q=0 AND PBStart AND NOT(_a=n) THEN a:=n;
ELSIF _q=7 AND _a<bnda               THEN a:=_a+1;
ELSIF _q=1 AND _a>0                  THEN a:=_a-1;
ELSIF _q=3 AND _a>0                  THEN a:=_a-1;
END_IF;
IF    _q=4 AND _b<bndb          THEN b:=_b+1;
ELSIF _q=6 AND _b>0             THEN b:=_b-1;
END_IF;
IF    _q=2 AND _c<bndc          THEN c:=_c+1;
ELSIF _q=5 AND _c<bndc          THEN c:=_c+1;
ELSIF _q=8 AND _c>0 AND PBReset THEN c:=0;
END_IF;
IF    _q=1 AND     _a>0  THEN q:=2;
ELSIF _q=1 AND NOT(_a>0) THEN q:=8;
ELSIF _q=2               THEN q:=3;
ELSIF _q=3 AND     _a>0  THEN q:=4;
ELSIF _q=3 AND NOT(_a>0) THEN q:=6;
ELSIF _q=4               THEN q:=5;
ELSIF _q=5               THEN q:=2;
ELSIF _q=6 AND     _b>0  THEN q:=7;
ELSIF _q=6 AND NOT(_b>0) THEN q:=1;
ELSIF _q=7               THEN q:=6;
ELSIF _q=8 AND PBReset   THEN q:=0;
ELSIF _q=0 AND PBStart   THEN q:=1;
END_IF;
(* Pseudo operator section: saving previous values of variables *)
_q:=q; _a:=a; _b:=b; _c:=c; _n:=n;
_PBStart:=PBStart; _PBReset:=PBReset; _PBPls:=PBPls; _PBMns:=PBMns;
```

При генерации программного кода по LTL-спецификации порядок размещения блоков IF-ELSIF должен подчиняться следующему правилу: некоторая переменная без псевдооператора «_» может быть задействована в IF-ELSIF-блоке другой переменной только в том случае, если её IF-ELSIF-блок уже находится выше по тексту.

Например, в представленной программе IF-ELSIF-блок переменной *n* должен идти раньше IF-ELSIF-блока переменной *a*, поскольку в правилах формирования нового значения *a* переменная *n* участвует без применения псевдооператора «_», т. е. новое значение *a* формируется на основе нового значения *n*.

При построении программы ПЛК каждая переменная из LTL-спецификации должна быть определена в подходящем разделе описания переменных («входы», «выходы», «локальные переменные») и проинициализирована в соответствии со спецификацией, а конкретнее — в соответствии с LTL-формулой инициализации вида S0. Входы ПЛК описываются в разделе VAR_INPUT, выходы —

в разделе VAR_OUTPUT, остальные переменные — в разделе VAR. Инициализация осуществляется в разделах объявления переменных.

Кроме того, для псевдополной системы переходов необходимо реализовать идею псевдооператора лидирующего подчеркивания «_». Для этого в самом конце ПЛК-программы выделяется место для псевдооператорного раздела, куда добавляются присваивания $\_v_i := v_i$, где $i = 1, \ldots, n$. При этом $\_v_i$ также необходимо определить в разделе описания переменных с таким же начальным значением, что и для переменной v_i.

В рамках модели поведения *LTS* переход из одного состояния в другое будет соответствовать выполнению программы ПЛК вплоть до псевдооператорного раздела, т. е. состояние будет представлять собой вектор значений всех программных переменных, который был получен до выполнения присваиваний, реализующих псевдооператор «_». Начальное состояние — состояние программы ПЛК после инициализации. Построение нового состояния ПЛК-программы происходит в следующем порядке: 1) выполняется псевдооператорный раздел (кроме первого раза), 2) входные переменные (входы ПЛК) получают новые значения, 3) программа ПЛК выполняется с самого начала до псевдооператорного раздела. При этом поведение ST-программы гарантированно соответствует исходной декларативной LTL-спецификации.

Заключение

В работе представлена декларативная LTL-спецификация, которую предлагается использовать для описания поведения управляющих программ. Данная спецификация позволяет описывать причины и правила изменения значений программных переменных. Предложенный способ декларирования поведения представляется естественным и удобным для управляющих систем.

В качестве модели программы используется размеченная система переходов *LTS*. Псевдополная система переходов *pLTS* содержит в себе поведение всех программ с заданным набором переменных. Декларативная LTL-спецификация формирует модель поведения программы *LTS* путем отбора из псевдополной системы переходов *pLTS* только необходимых путей.

Декларативная LTL-спецификация программы может быть непосредственно верифицирована методом проверки модели с помощью инструмента nuXmv. Для этого необходимо задать nuXmv-модель псевдополной системы переходов. Задача верификации для бесконечных моделей в общем случае неразрешима. Для разрешимости предлагается ограничить модель — сделать множество состояний конечным. В этом случае система переходов становится структурой Крипке.

Декларативная LTL-спецификация является конструктивной в том смысле, что по ней может быть построена управляющая программа. С этой целью в работе представлена вспомогательная императивная LTL-спецификация, для которой затем доказывается её эквивалентность декларативной LTL-спецификации. Императивная LTL-спецификация описывает поведение программы в императивном стиле, соответствующем, например, стилю языка программирования ST. Благодаря этому, становится возможной непосредственная трансляция императивной LTL-спецификации в ST-программу. Подход к трансляции описывается схематично в виде шаблона. Приведённая схема трансляции обеспечивает полное соответствие поведения полученной ST-программы исходной декларативной LTL-спецификации.

Декларативная LTL-спецификация является полной по Тьюрингу. Для доказательства этого факта применяется приём, позволяющий построить декларативную LTL-спецификацию для произвольной счётчиковой машины Минского, представляющей собой формализм, равномощный машине Тьюринга. Полнота по Тьюрингу даёт гарантию возможности описания абсолютно любого вычислительного алгоритма с помощью декларативной LTL-спецификации. Способ построения LTL-спецификации демонстрируется на примере счётчиковой машины возведения числа в квадрат.

Полученные в работе результаты подтверждают теоретическую состоятельность подхода к разработке и верификации управляющих программ на основе LTL-спецификации.

References

[1] S. Oks, M. Jalowski, M. Lechner, *et al.*, "Cyber-Physical Systems in the Context of Industry 4.0: A Review, Categorization and Outlook", *Inf. Systems Frontiers*, 2022. DOI: 10.1007/s10796-022-10252-x.

[2] E. Lee and S. Seshia, *Introduction to Embedded Systems – A Cyber-Physical Systems Approach*, 2nd ed. MIT Press, 2017, 561 pp.

[3] E. A. Parr, *Programmable Controllers. An Engineer's Guide*, 3rd ed. Newnes, 2003, 448 pp.

[4] V. Lifshic and L. Sadovskii, "Algebraic Models of Computing Machines", *UMN*, vol. 27, no. 3(165), pp. 79–125, 1972, in Russian.

[5] K.-Y. Cai, T. Chen, and T. Tse, "Towards Research on Software Cybernetics", in *Proceedings of 7th IEEE International on High-assurance Systems Engineering (HASE 2002)*, IEEE Computer Society Press, 2002, pp. 240–241.

[6] N. Polikarpova and A. Shalyto, *Automata-Based Programming*, 2nd ed. Spb.: Piter, 2011, 176 pp., in Russian.

[7] D. Harel and A. Pnueli, "On the Development of Reactive Systems", in *Logics and Models of Concurrent Systems*, vol. 13, 1985, pp. 477–498.

[8] A. Pnueli, "Applications of Temporal Logic to the Specification and Verification of Reactive Systems: A Survey of Current Trends", in *Current Trends in Concurrency*, vol. 224, Springer, 1986, pp. 510–584.

[9] A. Maurya and D. Kumar, "Reliability of safety-critical systems: A state-of-the-art review", *Quality and Reliability Engineering International*, vol. 36, no. 7, pp. 2547–2568, 2020.

[10] N. Rajabli, F. Flammini, R. Nardone, and V. Vittorini, "Software Verification and Validation of Safe Autonomous Cars: A Systematic Literature Review", in *IEEE Access*, vol. 9, 2021, pp. 4797–4819.

[11] V. D'Silva, D. Kroening, and G. Weissenbacher, "A Survey of Automated Techniques for Formal Software Verification", in *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 27, 2008, pp. 1165–1178.

[12] E. M. Clarke, T. A. Henzinger, H. Veith, and R. Bloem, *Handbook of Model Checking*, 1st ed. Springer Publishing Company, Incorporated, 2018, 1212 pp.

[13] Y. G. Karpov, *Model checking. Verification of Parallel and Distributed Program Systems*. BHV-Peterburg, 2010, 560 pp., in Russian.

[14] S. Velder, M. Lukin, A. Shalyto, and B. Yaminov, *Verification of Automata-Based Programs*. Spb Science, 2011, 244 pp., in Russian.

[15] E. M. Clarke, O. Grumberg, and D. Peled, *Verification of Program Models: Model Checking*. MCNMO, 2002, 416 pp., Transl. from English to Russian.

[16] E. Kuzmin and V. Sokolov, "Modeling, Specification and Construction of PLC-programs", *Modeling and Analysis of Information Systems*, vol. 20, no. 2, pp. 104–120, 2013, in Russian.

[17] E. Kuzmin, D. Ryabukhin, and V. Sokolov, "On the Expressiveness of the Approach to Constructing PLC-programs by LTL-Specification", *Modeling and Analysis of Information Systems*, vol. 22, no. 4, pp. 507–520, 2015, in Russian.

[18] E. Kuzmin, D. Ryabukhin, and V. Sokolov, "On the Expressiveness of the Approach to Constructing PLC-programs by LTL-Specification", *Automatic Control and Computer Sciences*, vol. 50, pp. 510–519, 2016.

[19] D. Ryabukhin, E. Kuzmin, and V. Sokolov, "Construction of CFC-programs by LTL-specification", *Modeling and Analysis of Information Systems*, vol. 23, no. 2, pp. 173–184, 2016, in Russian.

[20] D. Ryabukhin, E. Kuzmin, and V. Sokolov, “Construction of CFC-Programs by LTL-Specification”, *Automatic Control and Computer Sciences*, vol. 51, pp. 567–575, 2017.

[21] E. Kuzmin, “LTL-Specification of Counter Machines”, *Modeling and Analysis of Information Systems*, vol. 28, no. 1, pp. 104–119, 2021, in Russian.

[22] E. Kuzmin, “LTL-Specification of Bounded Counter Machines”, *Modeling and Analysis of Information Systems*, vol. 29, no. 1, pp. 44–59, 2022, in Russian.

[23] E. Kuzmin and V. Sokolov, “On Construction and Verification of PLC-Programs”, *Modeling and Analysis of Information Systems*, vol. 19, no. 4, pp. 25–36, 2012, in Russian.

[24] E. Kuzmin and V. Sokolov, “On Construction and Verification of PLC Programs”, *Automatic Control and Computer Sciences*, vol. 47, no. 7, pp. 443–451, 2013.

[25] E. Kuzmin and V. Sokolov, “Modeling, Specification and Construction of PLC-programs”, *Automatic Control and Computer Sciences*, vol. 48, no. 7, pp. 554–563, 2014.

[26] E. Kuzmin, V. Sokolov, and D. Ryabukhin, “Construction and Verification of PLC-programs by LTL-specification”, *Modeling and Analysis of Information Systems*, vol. 20, no. 4, pp. 5–22, 2013, in Russian.

[27] E. Kuzmin, V. Sokolov, and D. Ryabukhin, “Construction and Verification of PLC-Programs by LTL-Specification”, *Automatic Control and Computer Sciences*, vol. 49, no. 7, pp. 453–465, 2015.

[28] E. Kuzmin, V. Sokolov, and D. Ryabukhin, “Construction and Verification of PLC LD-programs by LTL-specification”, *Modeling and Analysis of Information Systems*, vol. 20, no. 6, pp. 78–94, 2013, in Russian.

[29] E. Kuzmin, V. Sokolov, and D. Ryabukhin, “Construction and Verification of PLC LD Programs by the LTL Specification”, *Automatic Control and Computer Sciences*, vol. 48, no. 7, pp. 424–436, 2014.

[30] D. Ryabukhin, E. Kuzmin, and V. Sokolov, “Construction of PLC IL-programs by LTL-specification”, *Modeling and Analysis of Information Systems*, vol. 21, no. 2, pp. 26–38, 2014, in Russian.

[31] E. Kuzmin, D. Ryabukhin, and V. Sokolov, “Modeling a Consistent Behavior of PLC-Sensors”, *Modeling and Analysis of Information Systems*, vol. 21, no. 4, pp. 75–90, 2014, in Russian.

[32] E. Kuzmin, D. Ryabukhin, and V. Sokolov, “Modeling a Consistent Behavior of PLC-Sensors”, *Automatic Control and Computer Sciences*, vol. 48, no. 7, pp. 602–614, 2014.

[33] *IEC 61131-3:2013 Programmable controllers – Part 3: Programming languages*, International Standard. [Online]. Available: https://webstore.iec.ch/publication/4552.

[34] A. Pnueli, “The Temporal Logic of Programs”, in *18th Annual Symposium on Foundations of Computer Science (SFCS 1977)*, IEEE Computer Society Press, 1977, pp. 46–57.

[35] M. Minsky, *Computation: Finite and Infinite Machines*. Prentice-Hall, 1967, 317 pp.

[36] R. Schroeppel, “A Two Counter Machine Cannot Calculate 2^N”, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, Artificial Intelligence Memo #257, 1972, 32 pp.

[37] E. V. Kuzmin, *Counter Machines*. Yaroslavl: Yaroslavl State University, 2010, 127 pp., in Russian.

Приложение. Построение LTL-спецификации счётчиковой машины 3cM

Продемонстрируем процедуру построения LTL-спецификации из теоремы 2 на примере счётчиковой машины *3cM* возведения числа в квадрат (раздел 5).

Выполним LTL-формализацию правил (15):

$$
\begin{aligned}
&(\delta_1)\ \mathbf{GX}\Big(\big[(\_q = 1) \wedge (\_a > 0) \Rightarrow (a = \_a - 1) \wedge (q = 2)\big] \wedge \big[(\_q = 1) \wedge \neg(\_a > 0) \Rightarrow (q = 8)\big]\Big),\\
&(\delta_2)\ \mathbf{GX}\big((\_q = 2) \Rightarrow (c = \_c + 1) \wedge (q = 3)\big),\\
&(\delta_3)\ \mathbf{GX}\Big(\big[(\_q = 3) \wedge (\_a > 0) \Rightarrow (a = \_a - 1) \wedge (q = 4)\big] \wedge \big[(\_q = 3) \wedge \neg(\_a > 0) \Rightarrow (q = 6)\big]\Big),\\
&(\delta_4)\ \mathbf{GX}\big((\_q = 4) \Rightarrow (b = \_b + 1) \wedge (q = 5)\big),\\
&(\delta_5)\ \mathbf{GX}\big((\_q = 5) \Rightarrow (c = \_c + 1) \wedge (q = 2)\big),\\
&(\delta_6)\ \mathbf{GX}\Big(\big[(\_q = 6) \wedge (\_b > 0) \Rightarrow (b = \_b - 1) \wedge (q = 7)\big] \wedge \big[(\_q = 6) \wedge \neg(\_b > 0) \Rightarrow (q = 1)\big]\Big),\\
&(\delta_7)\ \mathbf{GX}\big((\_q = 7) \Rightarrow (a = \_a + 1) \wedge (q = 6)\big).
\end{aligned}
\tag{29}
$$

Осуществим фрагментацию формул (29) (результат представлен в левой колонке таблицы 1), и сгруппируем эти формулы по переменным (результат приведён в правой колонке таблицы 1).

Table 1. Fragmented (left) and grouped by variables (right) LTL-formulas

(δ_1)	$\mathbf{GX}\big((\_q = 1) \wedge (\_a > 0) \Rightarrow (a = \_a - 1)\big)$,
	$\mathbf{GX}\big((\_q = 1) \wedge (\_a > 0) \Rightarrow (q = 2)\big)$,
	$\mathbf{GX}\big((\_q = 1) \wedge \neg(\_a > 0) \Rightarrow (q = 8)\big)$,
(δ_2)	$\mathbf{GX}\big((\_q = 2) \Rightarrow (c = \_c + 1)\big)$,
	$\mathbf{GX}\big((\_q = 2) \Rightarrow (q = 3)\big)$,
(δ_3)	$\mathbf{GX}\big((\_q = 3) \wedge (\_a > 0) \Rightarrow (a = \_a - 1)\big)$,
	$\mathbf{GX}\big((\_q = 3) \wedge (\_a > 0) \Rightarrow (q = 4)\big)$,
	$\mathbf{GX}\big((\_q = 3) \wedge \neg(\_a > 0) \Rightarrow (q = 6)\big)$,
(δ_4)	$\mathbf{GX}\big((\_q = 4) \Rightarrow (b = \_b + 1)\big)$,
	$\mathbf{GX}\big((\_q = 4) \Rightarrow (q = 5)\big)$,
(δ_5)	$\mathbf{GX}\big((\_q = 5) \Rightarrow (c = \_c + 1)\big)$,
	$\mathbf{GX}\big((\_q = 5) \Rightarrow (q = 2)\big)$,
(δ_6)	$\mathbf{GX}\big((\_q = 6) \wedge (\_b > 0) \Rightarrow (b = \_b - 1)\big)$,
	$\mathbf{GX}\big((\_q = 6) \wedge (\_b > 0) \Rightarrow (q = 7)\big)$,
	$\mathbf{GX}\big((\_q = 6) \wedge \neg(\_b > 0) \Rightarrow (q = 1)\big)$,
(δ_7)	$\mathbf{GX}\big((\_q = 7) \Rightarrow (a = \_a + 1)\big)$,
	$\mathbf{GX}\big((\_q = 7) \Rightarrow (q = 6)\big)$.

Таблица 1. Фрагментированные (слева) и сгруппированные по переменным (справа) LTL-формулы

(a)	$\mathbf{GX}\big((\_q = 1) \wedge (\_a > 0) \Rightarrow (a = \_a - 1)\big)$,
	$\mathbf{GX}\big((\_q = 3) \wedge (\_a > 0) \Rightarrow (a = \_a - 1)\big)$,
	$\mathbf{GX}\big((\_q = 7) \Rightarrow (a = \_a + 1)\big)$,
(b)	$\mathbf{GX}\big((\_q = 4) \Rightarrow (b = \_b + 1)\big)$,
	$\mathbf{GX}\big((\_q = 6) \wedge (\_b > 0) \Rightarrow (b = \_b - 1)\big)$,
(c)	$\mathbf{GX}\big((\_q = 2) \Rightarrow (c = \_c + 1)\big)$,
	$\mathbf{GX}\big((\_q = 5) \Rightarrow (c = \_c + 1)\big)$,
(q)	$\mathbf{GX}\big((\_q = 1) \wedge (\_a > 0) \Rightarrow (q = 2)\big)$,
	$\mathbf{GX}\big((\_q = 1) \wedge \neg(\_a > 0) \Rightarrow (q = 8)\big)$,
	$\mathbf{GX}\big((\_q = 2) \Rightarrow (q = 3)\big)$,
	$\mathbf{GX}\big((\_q = 3) \wedge (\_a > 0) \Rightarrow (q = 4)\big)$,
	$\mathbf{GX}\big((\_q = 3) \wedge \neg(\_a > 0) \Rightarrow (q = 6)\big)$,
	$\mathbf{GX}\big((\_q = 4) \Rightarrow (q = 5)\big)$,
	$\mathbf{GX}\big((\_q = 5) \Rightarrow (q = 2)\big)$,
	$\mathbf{GX}\big((\_q = 6) \wedge (\_b > 0) \Rightarrow (q = 7)\big)$,
	$\mathbf{GX}\big((\_q = 6) \wedge \neg(\_b > 0) \Rightarrow (q = 1)\big)$,
	$\mathbf{GX}\big((\_q = 7) \Rightarrow (q = 6)\big)$.

Конъюнктивно объединим формулы в каждой группе из правой колонки таблицы 1, а также добавим формулы, описывающие отсутствие изменений значения переменной. Получим императивную LTL-спецификацию:

$$
\begin{aligned}
(a)\ & \mathbf{GX}\big((\_q = 1) \wedge (\_a > 0) \Rightarrow (a = \_a - 1)\big) \wedge \\
& \mathbf{GX}\big((\_q = 3) \wedge (\_a > 0) \Rightarrow (a = \_a - 1)\big) \wedge \\
& \mathbf{GX}\big((\_q = 7) \Rightarrow (a = \_a + 1)\big), \\
& \mathbf{GX}\Big(\neg\big((\_q = 1) \wedge (\_a > 0)\big) \wedge \neg\big((\_q = 3) \wedge (\_a > 0)\big) \wedge \neg\big((\_q = 7)\big) \Rightarrow (a = \_a)\Big), \\
(b)\ & \mathbf{GX}\big((\_q = 4) \Rightarrow (b = \_b + 1)\big) \wedge \\
& \mathbf{GX}\big((\_q = 6) \wedge (\_b > 0) \Rightarrow (b = \_b - 1)\big), \\
& \mathbf{GX}\Big(\neg\big((\_q = 4)\big) \wedge \neg\big((\_q = 6) \wedge (\_b > 0)\big) \Rightarrow (b = \_b)\Big), \\
(c)\ & \mathbf{GX}\big((\_q = 2) \Rightarrow (c = \_c + 1)\big) \wedge \\
& \mathbf{GX}\big((\_q = 5) \Rightarrow (c = \_c + 1)\big), \\
& \mathbf{GX}\Big(\neg\big((\_q = 2)\big) \wedge \neg\big((\_q = 5)\big) \Rightarrow (c = \_c)\Big), \\
(q)\ & \mathbf{GX}\big((\_q = 1) \wedge (\_a > 0) \Rightarrow (q = 2)\big) \wedge \\
& \mathbf{GX}\big((\_q = 1) \wedge \neg(\_a > 0) \Rightarrow (q = 8)\big) \wedge \\
& \mathbf{GX}\big((\_q = 2) \Rightarrow (q = 3)\big) \wedge \\
& \mathbf{GX}\big((\_q = 3) \wedge (\_a > 0) \Rightarrow (q = 4)\big) \wedge \\
& \mathbf{GX}\big((\_q = 3) \wedge \neg(\_a > 0) \Rightarrow (q = 6)\big) \wedge \\
& \mathbf{GX}\big((\_q = 4) \Rightarrow (q = 5)\big) \wedge \\
& \mathbf{GX}\big((\_q = 5) \Rightarrow (q = 2)\big) \wedge \\
& \mathbf{GX}\big((\_q = 6) \wedge (\_b > 0) \Rightarrow (q = 7)\big) \wedge \\
& \mathbf{GX}\big((\_q = 6) \wedge \neg(\_b > 0) \Rightarrow (q = 1)\big) \wedge \\
& \mathbf{GX}\big((\_q = 7) \Rightarrow (q = 6)\big), \\
& \mathbf{GX}\Big(\neg\big((\_q = 1) \wedge (\_a > 0)\big) \wedge \neg\big((\_q = 1) \wedge \neg(\_a > 0)\big) \wedge \neg\big((\_q = 2)\big) \wedge \\
& \qquad \neg\big((\_q = 3) \wedge (\_a > 0)\big) \wedge \neg\big((\_q = 3) \wedge \neg(\_a > 0)\big) \wedge \neg\big((\_q = 4)\big) \wedge \\
& \qquad \neg\big((\_q = 5)\big) \wedge \neg\big((\_q = 6) \wedge (\_b > 0)\big) \wedge \neg\big((\_q = 6) \wedge \neg(\_b > 0)\big) \wedge \\
& \qquad \neg\big((\_q = 7)\big) \Rightarrow (q = \_q)\Big).
\end{aligned}
$$

Далее по этой императивной LTL-спецификации строится декларативная LTL-спецификация счётчиковой машины *3cM* из раздела 5.

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 4, 2023
journal homepage: www.mais-journal.ru

ALGORITHMS IN COMPUTER SCIENCE

Joint simplification of various types spatial objects while preserving topological relationships

O. P. Yakimova[1], D. M. Murin[1], V. G. Gorshkov[1]

DOI: 10.18255/1818-1015-2023-4-340-353

[1]P.G. Demidov Yaroslavl State University, 14 Sovetskaya str., Yaroslavl 150003, Russia.

MSC2020: 68W99
Research article
Full text in Russian

Received November 13, 2023
After revision November 21, 2023
Accepted November 22, 2023

Cartographic generalization includes the process of graphically reducing information from reality or larger scaled maps to display only the information that is necessary at a specific scale. After generalization, maps can show the main things and essential characteristics. The scale, use and theme of maps, geographical features of cartographic regions and graphic dimensions of symbols are the main factors affecting cartographic generalization. Geometric simplification is one of the core components of cartographic generalization. The topological relations of spatial features also play an important role in spatial data organization, queries, updates, and quality control. Various map transformations can change the relationships between features, especially since it is common practice to simplify each type of spatial feature independently (first administrative boundaries, then road network, settlements, hydrographic network, etc.). In order to detect the spatial conflicts a refined description of topological relationships is needed. Considering coverings and mesh structures allows us to reduce the more general problem of topological conflict correction to the problem of resolving topological conflicts within a single mesh cell. In this paper, a new simplification algorithm is proposed. Its peculiarity is the joint simplification of a set of spatial objects of different types while preserving their topological relations. The proposed algorithm has a single parameter — the minimum map detail size (usually it is equal to one millimeter in the target map scale). The first step of the algorithm is the construction of a special mesh data structure. On its basis for each spatial object a sequence of cells is formed, to which points of this object belong. If a cell contains points of only one object, its geometric simplification is performed within the bounding cell using the sleeve-fitting algorithm. If a cell contains points of several objects, geometric simplification is performed using a special topology-preserving procedure.

Keywords: simplification algorithm; topological relationships; mesh data structure; spatial data; consistent cartographic generalization

INFORMATION ABOUT THE AUTHORS

Olga P. Yakimova corresponding author	orcid.org/0000-0001-8816-2802 . E-mail: olga_pavl02@mail.ru Associate professor, Ph.D. in Mathematics.
Dmitriy M. Murin	orcid.org/0000-0002-8068-0784. E-mail: d.murin@uniyar.ac.ru Associate professor, Ph.D. in Mathematics.
Vladislav G. Gorshkov	orcid.org/0000-0002-2424-3942. E-mail: gorshkov.vladik@mail.ru Programmer.

Funding: P.G. Demidov Yaroslavl State University, project No. GM-2023-03.

For citation: O. P. Yakimova, D. M. Murin, and V. G. Gorshkov, “Joint simplification of various types spatial objects while preserving topological relationships”, *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 340-353, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 4, 2023
сайт журнала: www.mais-journal.ru

ALGORITHMS IN COMPUTER SCIENCE

Совместное упрощение пространственных объектов различного типа с сохранением топологических отношений

О. П. Якимова[1], Д. М. Мурин[1], В. Г. Горшков[1]

DOI: 10.18255/1818-1015-2023-4-340-353

[1]Ярославский государственный университет им П.Г. Демидова, ул. Советская, д. 14, г. Ярославль, 150003 Россия.

УДК 004.67+528.91
Научная статья
Полный текст на русском языке

Получена 13 ноября 2023 г.
После доработки 21 ноября 2023 г.
Принята к публикации 22 ноября 2023 г.

Картографическая генерализация включает выбор отображаемых на карте объектов и явлений и их упрощение (обобщение) с сохранением основных типичных черт и характерных особенностей, а также взаимосвязей в соответствии с критериями, задаваемыми в запросе пользователем, в том числе решаемой задачей и масштабом отображаемой карты. Различные преобразования карт могут изменить отношения между объектами, тем более что общепринятой является практика упрощения каждого типа пространственных объектов независимо (сначала административные границы, потом дорожная сеть, населенные пункты, гидрографическая сеть и т. д.). Разрешение топологических конфликтов — одна из важнейших задач цифровой генерализации карт, решению которой уделяется особое внимание с начала исследований в этой области. Рассмотрение покрытий и сеточных структур позволяет свести более общую проблему коррекции топологических конфликтов к задаче разрешения топологических конфликтов внутри одной ячейки сетки.

В настоящей работе предлагается новый алгоритм геометрического упрощения. Его особенностью является совместное упрощение множества пространственных объектов различного типа с сохранением их топологических отношений. Предлагаемый алгоритм имеет единственный параметр минимальный размер отображаемой на карте детали (обычно он равен одному миллиметру в целевом масштабе карты). Первым шагом алгоритма является построение специальной сеточной структуры данных. На ее основе для каждого пространственного объекта формируется последовательность ячеек, которым принадлежат точки данного объекта. Если в ячейке находятся точки только одного объекта, то его геометрическое упрощение происходит в рамках ограничивающей ячейки по алгоритму sleeve-fitting. Если в ячейке содержатся точки нескольких объектов, то геометрическое упрощение осуществляется с помощью специальной, сохраняющей топологию, процедуры.

Ключевые слова: алгоритм упрощения; топологические отношения; сеточная структура данных; пространственные данные; согласованная картографическая генерализация

ИНФОРМАЦИЯ ОБ АВТОРАХ

Ольга Павловна Якимова автор для корреспонденции	orcid.org/0000-0001-8816-2802 . E-mail: olga_pavl02@mail.ru доцент, канд. физ.-мат. наук.
Дмитрий Михайлович Мурин	orcid.org/0000-0002-8068-0784. E-mail: d.murin@uniyar.ac.ru доцент, канд. физ.-мат. наук.
Владислав Геннадьевич Горшков	orcid.org/0000-0002-2424-3942. E-mail: gorshkov.vladik@mail.ru программист.

Финансирование: Ярославский государственный университет им. П.Г. Демидова, проект № GM-2023-03.

Для цитирования: O. P. Yakimova, D. M. Murin, and V. G. Gorshkov, “Joint simplification of various types spatial objects while preserving topological relationships”, *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 340-353, 2023.

Введение

Геоинформационные системы (ГИС) представляют собой совокупность содержащейся в базах данных картографической информации и обеспечивающих ее обработку информационных технологий и технических средств. Такие системы обеспечивают сбор, актуализацию, хранение, анализ и графическое отображение (представление) картографических и иных, связанных с ними, данных. Конечными пользователями геоинформационных систем могут выступать как физические лица, так и иные информационные системы и процессы. Технические средства и информационные технологии, входящие в состав ГИС можно разделить на серверные, реализующие основной функционал ГИС, и пользовательские. Используемые физическими лицами технические средства и информационные технологии в первую очередь рассчитаны на отображение (визуализацию) данных, получаемых от серверной части ГИС.

Число (в том числе прикладных) задач, решаемых с помощью ГИС, непрерывно растет и охватывает различные сферы экономики и жизнедеятельности, например, государственное и муниципальное управление, землеустройство, налогообложение, логистика, транспорт, сельское хозяйство, туризм, метеорология, геологоразведка и т. д. Для эффективного решения задач в каждой области представляют интерес свои данные. Например, для решения задач землеустройства наибольший интерес представляют границы земельных участков, а для управления транспортом — дорожные сети. При этом одни и те же данные могут иметь отличную ценность для различных областей. Например, для определения маршрута достаточно знать карту дорожной сети и возможные варианты путей, а для расчета стоимости ремонта дорожного полотна требуются детальные сведения о протяженности участка, текущем состоянии, типе дорожного покрытия, прогнозируемом транспортном потоке и т. д. Возможность оперативного решения задач с помощью картографических данных может быть обеспечена только в том случае, если выдаваемые серверной частью ГИС данные могут успешно отображаться (обрабатываться) на пользовательских устройствах. Большой объем информации, который может быть выдан серверной частью ГИС, не всегда с успехом может быть обработан клиентским устройством.

Цифровая (компьютерная) картографическая генерализация (от лат. generalis — общий, главный) — это процесс обработки картографических данных, в целях предоставления по запросу пользователя информации, которая может быть воспроизведена (обработана) на пользовательском устройстве, достаточной для обеспечения эффективного решения стоящих перед пользователем задач. Картографическая генерализация включает выбор отображаемых на карте объектов и явлений и их упрощение (обобщение) с сохранением основных типичных черт и характерных особенностей, а также взаимосвязей в соответствии с критериями, задаваемыми в запросе пользователем, в том числе решаемой задачей и масштабом отображаемой карты.

Одни из первых работ [1—3], связанные с цифровой генерализацией карт были опубликованы в открытой печати в конце 1950-х, начале 1960-х годов. Параллельно с этим в начале 1960-х годов начинают появляться первые геоинформационные системы, которые на начальном этапе являлись по сути структурами данных, содержащих информацию о географических объектах и их координатах. В таких системах использовалось растровое (ячеистое) представление информации. В конце 1960-х годов Бюро переписи США был разработан формат Geographic Base File, Dual Independent Map Encding (GBF-DIME), который позволил учитывать топологические отношения между полилинейными и полигональными картографическими объектами. В этом формате впервые были проиндексированы узловые точки, а также присвоены идентификаторы некоторым объектам.

В 1970-х годах основные усилия исследователей были сосредоточены на разработке алгоритмов генерализации линейных и полилинейных объектов. Один из подходов к упрощению полилиний, до сих пор не потерявший актуальность, был независимо предложен Рамером [4], Дугласом и Пекером [5] в 1972 и 1973 годах соответственно. Идея данного алгоритма заключается в том,

чтобы по заданной полилинии построить другую, содержащую меньшее число вершин, однако сохраняющую некоторые характеристики исходной. Близость исходной и упрощенной полилиний определяется по максимальному расстоянию между ними.

Единого мнения о том существует ли «оптимальная» упрощенная линия для каждой заданной нет. Так в работе [6] приводятся аргументы в пользу того, что некоторые упрощенные вершины более точно передают форму исходной полилинии. Однако в работе [7] утверждается, что ни одно конкретное множество вершин не может быть единственно верным для представления полилинии. С нашей точки зрения, различные множества вершин могут быть более репрезентативны при решении различных задач, что, в том числе, должно стимулировать исследования в сфере адаптирующихся алгоритмов [8, 9].

В 1974 году в работе Р. Финкеля и Дж. Бетли [10] была предложена структура данных, с помощью которой можно эффективно решать задачу хранения и извлечения информации по ключам, подходящая для хранения картографических данных (более точно, данных на плоскости) — квадродерево (или дерево квадрантов). Кроме самой структуры квадродерева в работе также рассматривался ее оптимизированный вариант. Квадродеревья нашли широкое применение в пространственных базах данных.

В начале 1980-х годов были исследованы алгоритмы генерализации для полигональных объектов. При этом уже в это время было осознано, что представление картографических данных в различных масштабах является одной из важнейших проблем вычислительной картографии. В 1982 году появляется первая коммерческая ГИС — ARC/INFO компании ERSI.

В 1984 году А. Гуттман в работе [11] предложил структуру данных, представляющую многомерное пространство в виде множества иерархически вложенных и, возможно, пересекающихся, прямоугольников (параллелепипедов), обладающую свойством сбалансированности, — *R*-дерево. При хранении картографических данных в такой структуре близко расположенные объекты должны попадать в один лист дерева. Каждый лист *R*-дерева при этом должен хранить данные, описывающие картографический объект и ограничивающий прямоугольник объекта (объектов).

Развитие ГИС породило вопрос об унификации форматов представления информации для хранения и обмена картографическими данными. В связи с чем в середине-конце 1980-х годов был разработан и принят стандарт стран Варшавского договора «Единая система классификации и кодирования картографической информации» (ЕСККИ), а в 1991 году Digital Geographic Information Exchange Standard (DIGEST) — стандарт стран НАТО. При этом единого, всеми признаваемого стандарта представления геопространственных данных не существует до сих пор [12].

Различные преобразования карт, в том числе изменение масштаба или генерализация, могут изменить отношения между объектами. Например, могут измениться топологические отношения, отношения порядка или соотношения размеров [13]. Разрешение топологических конфликтов — одна из важнейших задач цифровой генерализации карт, решению которой уделяется особое внимание с начала исследований этой области [14, 15]. С 1990-х годов исследуются подходы к генерализации линейных картографических объектов на основе различных разбиений (покрытий) карты, например, триангуляции Делоне [16], квадратных [17] и гексагональных [18]. Рассмотрение покрытий и сеточных структур позволяет свести более общую проблему коррекции топологических конфликтов к задаче разрешения топологических конфликтов внутри одной ячейки сетки.

Следует отметить, что алгоритмы генерализации часто рассматривают объекты исключительно одной геометрии: либо полилинии, либо полигоны. Можно сказать, что общепринятой является классификация алгоритмов по этому признаку [19]. У такого подхода есть некоторые основания, поскольку в ГИС информация часто хранится «по слоям», причем в одном слое хранятся данные одной геометрической природы, например, дорожные сети или речные сети, представленные полилиниями, водоемы или здания, преставленные полигонами. С одним слоем карты существенно

проще работать в силу относительно небольшого (по сравнению со всей картой) объема данных. Тем не менее, нередко могут возникать ситуации, при которых один географический объект при таком подходе оказывается сегментирован, а его элементы хранятся в разных слоях карты. Так происходит, например, в случае, если в течении реки находится водоем: озеро, болото и т. п. или на отдельном участке русло имеет существенно большую ширину по сравнению с соседними участками. В связи с вышеизложенным представляет интерес алгоритм, способный решать задачи по разрешению топологических конфликтов для объектов различного вида: и точек, и полилиний, и полигонов.

В настоящей работе предлагается новый алгоритм геометрического упрощения. Его особенностью является совместное упрощение множества пространственных объектов различного типа с сохранением их топологических отношений.

1. Алгоритм согласованной генерализации

Исходными данными для предлагаемого алгоритма является набор слоев картографических данных. Каждый слой содержит информацию о некотором виде пространственных объектов (границах, реках, озерах, дорогах, лесных массивах и т. д.) и имеет один тип геометрии, то есть содержит только полилинии, только полигоны или только точки. На рисунке 1 показан пример набора исходных данных.

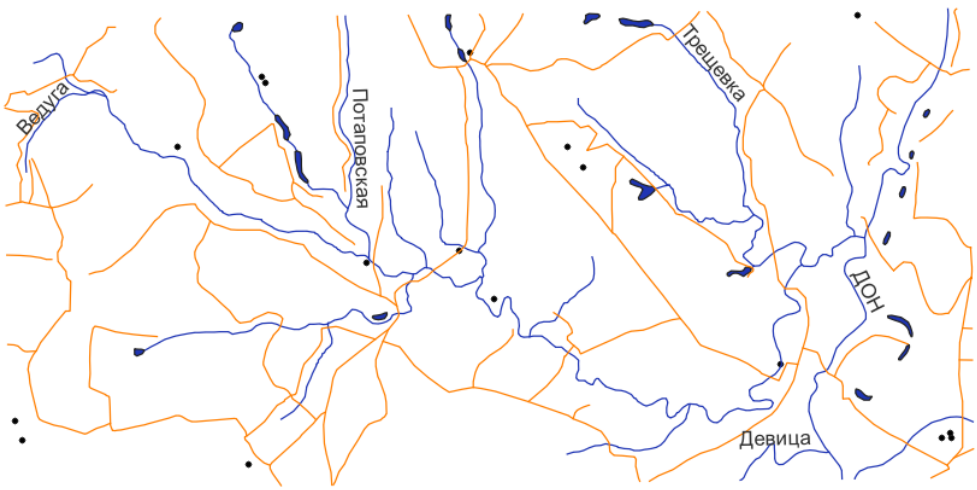


Fig. 1. Input data example. Different line layers are shown by different types of polylines

Рис. 1. Пример исходных данных. Различные линейные слои показаны разными полилиниями

Предлагаемый алгоритм имеет единственный параметр — минимальный размер отображаемой на карте детали — *Minsize* (обычно он равен одному миллиметру в целевом масштабе карты). Основными шагами алгоритма являются:

1. *Построение сеточной структуры данных.* Все слои данных последовательно просматриваются и объекты каждого слоя заносятся в сеточную структуру данных.
2. *Упрощение.* Для каждого пространственного объекта формируется последовательность ячеек сеточной структуры данных, которым принадлежат точки данного объекта. Если в ячейке находится точки только одного объекта, то его геометрическое упрощение происходит в рамках ограничивающей ячейки по алгоритму «облегающего/обтягивающего рукава», предложенному в работе [20]. Если в ячейке содержатся точки нескольких объектов, то геометрическое упрощение осуществляется с помощью специальной, сохраняющей топологию, процедуры.

Далее каждый шаг алгоритма рассматривается более подробно.

1.1. Построение сеточной структуры данных

Предлагаемая структура данных на основе квадродерева может содержать следующие типы объектов:

1. Точечные объекты.
2. Полилинейные объекты.
3. Полигональные объекты (тип полилинейных объектов, у которых начало и конец совпадают).

Каждый тип объекта представляет собой структуру данных, включающую:

- *ID* — идентификатор объекта.
- *Geometry* — геометрия объекта (Point (точка), Polyline (полилиния), Polygon (полигональный объект)).
- *Points* — список точек, которые составляют объект.
- *Path* — представляет собой маршрут по ячейкам сеточной структуры данных (проинициализирован * — корневой ячейкой).

Вход алгоритма:

1. *Minsize* — минимальный размер различимой детали на карте.
2. Список всех картографических объектов.

Стадия предобработки:

Находятся минимальные и максимальные значения по осям X и Y от охвата карты.

$$y_{max}, y_{min}, x_{max}, x_{min}.$$

Создается корневая ограничивающая квадратная ячейка, куда помещаются все картографические объекты. Сторона этой ячейки равна:

$$\max(y_{max} - y_{min}, x_{max} - x_{min}).$$

Левый нижний угол ячейки помещается в точку (x_{min}, y_{min}).

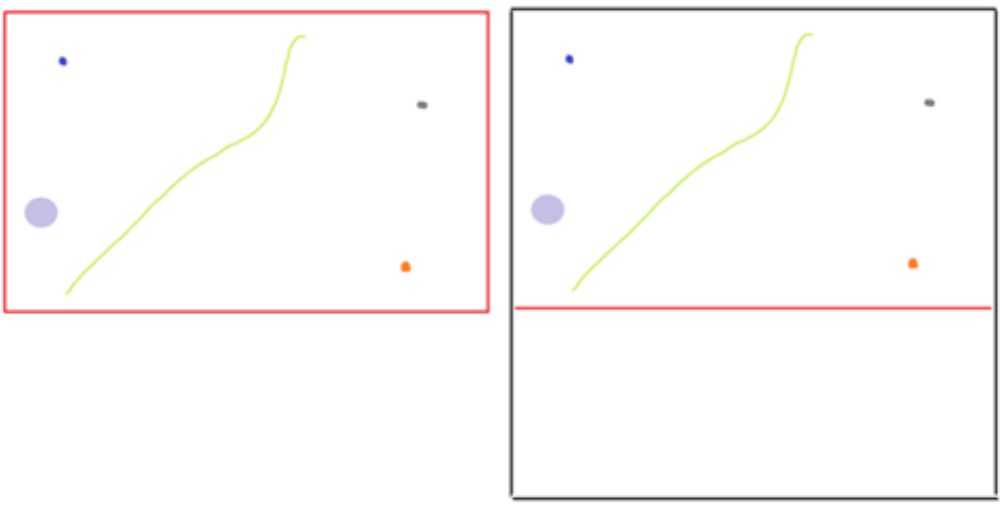


Fig. 2. Map coverage (left — before preprocessing, right — after)

Рис. 2. Охват карты (слева до предобработки, справа — после)

Таким образом, получаем охват карты, который представляет собой ограничивающий квадрат, в котором находятся все рассматриваемые объекты, с координатами левого нижнего, левого верхнего, правого верхнего, правого нижнего углов.

Тело алгоритма:

1. Создаем очередь Q на обработку ячеек.
2. Помещаем в Q корневую ячейку.
3. Пока очередь не пуста:
   (a) Извлекаем из очереди Q ячейку на обработку, помещаем указатель на нее в переменную *currentNode*.
   (b) Если в ячейке больше одного уникального объекта (то есть ячейке принадлежат два или более объекта с различными *ID*) и сторона ячейки — $side > Minsize$, то:
      i. Делим ячейку *currentNode* на четыре одинаковых квадрата с длиной стороны $side/2$ (полученные ячейки обозначим $node_1$, $node_2$, $node_3$, $node_4$.
      ii. Для каждого объекта из *currentNode* определяем, в каких дочерних ячейках присутствуют его элементы, и корректируем его поле *Path*, заменяя *currentNode* на последовательность дочерних ячеек, по которым проходит данный объект.
      iii. Помещаем ячейки $node_1$, $node_2$, $node_3$, $node_4$ в Q.
   (c) Иначе продолжаем алгоритм.

Каждый точечный объект лежит в какой-то определенной единственной ячейке сетки. Полилинейный объект сетка разбивает на сегменты, каждый из которых принадлежит определенной ячейке. Сегменты полигонального объекта также будут расположены в нескольких ячейках, в силу малого размера которых, сегмент полигонального объекта внутри ячейки можно рассматривать как полилинейный (считаем ячейку замкнутым множеством). Пример построенной сеточной структуры данных изображен на рисунке 3.

Каждая ячейка сетки может быть пустой, т. е. не содержать ни одного пространственного объекта, содержать один или несколько пространственных объектов.

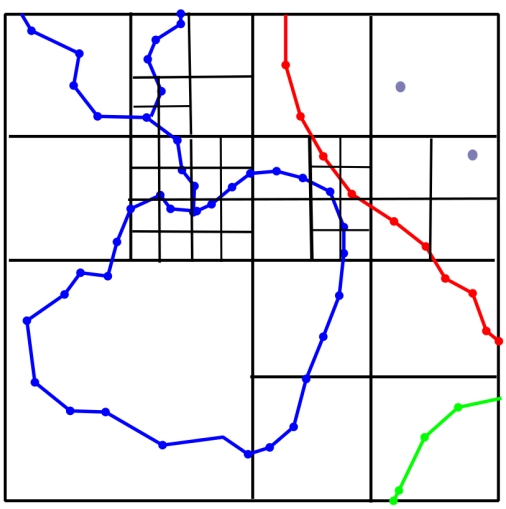


Fig. 3. Example of building a grid data structure

Рис. 3. Пример построения сеточной структуры данных

Рассмотрим подробнее шаг ii алгоритма. Дочерние ячейки имеют номера:

- 0 — левая нижняя ячейка,
- 1 — левая верхняя ячейка,
- 2 — правая верхняя ячейка,
- 3 — правая нижняя ячейка.

Будем рассматривать точки линейного или полигонального объекта в порядке их следования. Допустим, что первая точка S объекта, принадлежащая данной ячейке, находится в левом нижнем

углу, то есть в дочерней ячейке с номером 0. Если такая точка находится на границе сетки, то рассматривается следующая за ней, до ситуации, при которой дочерняя ячейка может быть однозначно определена. Обозначим через F следующую точку объекта.

На рисунке 4, показаны возможные варианты расположения вершин S и F при условии что S лежит в нулевой дочерней ячейке. Очевидно, что для дочерних ячеек с номерами $1-3$ рассуждения будут аналогичными.

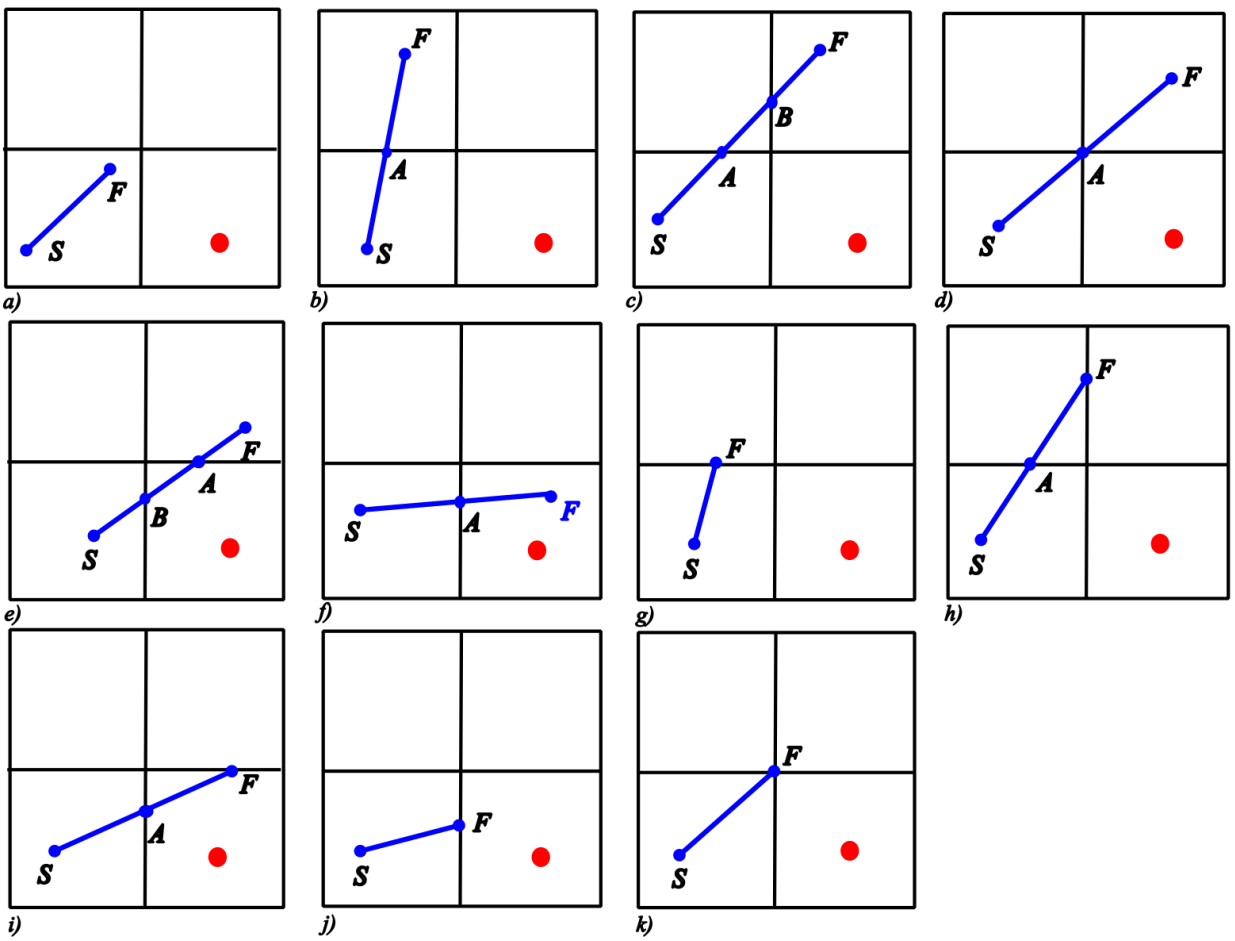


Fig. 4. Options for the location of polyline vertices

Рис. 4. Варианты расположения вершин полилинии

Допустим, что два объекта находятся в одной ячейке сетки (на рисунке 4 — линейный и точечный), и тогда необходимо разбить текущую ячейку на четыре дочерних.

Для случаев a), g), j), к) нам не нужно искать точки пересечения с ячейками сетки, так как вершина F (как и вершина S) лежит в левой нижней ячейке.

Для остальных ситуаций нужно искать точки пересечения с сеткой для того, чтобы в дальнейшем учитывать их топологические отношения, если потребуется делить дочернюю ячейку на еще более мелкие. Обозначим через A и B точки пересечения с дочерними ячейками сетки.

Для случаев b) и h) точка пересечения A находится пересечением отрезка SF с горизонтальным «терминатором», а для случаев f), i) — с вертикальным «терминатором». Каждая из этих ситуаций однозначно идентифицируется положением точки F.

Ситуации c), d) и e) несколько сложнее. Для них требуется определить каким образом отрезок *SF* проходит по ячейкам сетки. Во всех этих случаях точка *F* лежит в правом верхнем углу, следовательно, у нас есть следующие варианты прохождения отрезка *SF*:

– Отрезок *SF* проходит через левую верхнюю ячейку — c).
– Отрезок *SF* проходит через точку пересечения терминаторов — d).
– Отрезок *SF* проходит через правую нижнюю ячейку — e).

Для идентификации одной из этих ситуаций (c, d, e) используем следующий, разработанный нами, алгоритм:

1. Находим точку пересечения отрезка *SF* с горизонтальным терминатором. Обозначим эту точку через *A*.
2. Если точка *A* лежит левее, чем точка пересечения терминаторов, то это ситуация c). Находим точку пересечения отрезка *SF* с вертикальным терминатором. Обозначим эту точку через *B*. В соответствующие ячейки помещаем точки пересечения полилинии с терминаторами (точку *A* помещаем в левую нижнюю ячейку, *A* и *B* — в левую верхнюю ячейку, *B* — в правую верхнюю ячейку).
3. Иначе, если точка *A* совпадает с точкой пересечения терминаторов, то это ситуация d). В соответствующие ячейки (левую нижнюю, правую верхнюю) помещаем точку *A* (пересечения терминаторов).
4. Иначе (точка *A* находится правее, чем точка пересечения терминаторов) — это ситуация e). Находим точку пересечения отрезка *SF* с вертикальным терминатором. Обозначим эту точку через *B*. В соответствующие ячейки помещаем точки пересечения полилинии с терминаторами (точку *A* помещаем в правую верхнюю, *A* и *B* — в правую нижнюю, *B* — в левую нижнюю ячейку).

Результат работы алгоритма:

1. Построенное квадродерево.
2. Для каждого объекта создан свой маршрут *Path* по ячейкам сетки.

С помощью разработанной структуры данных можно ускорить работу ГИС при выполнении запросов вида: «найти ближайший объект по отношению к данному» (наивный алгоритм — перебор всех картографических объектов, в то время как достаточно перебрать все объекты в ячейке, в которой находится данный объект и все ближайшие ячейки по отношению к той, в которой находится данный объект), «получить информацию о топологических отношениях объектов» и др.

1.2. Упрощение

Как результат работы первого этапа алгоритма для каждого полилинейного или полигонального объекта получаем последовательность ячеек сеточной структуры данных, по которым он проходит. Согласно построению описанной выше структуры только в ячейках минимального размера может находиться более одного объекта. Внутри любых других ячеек производится геометрическое упрощение полилинейного объекта или границы полигонального объекта с помощью алгоритма «обтягивающего рукава» (sleeve-fitting). Идея его работы заключается в методе постепенного «обтягивания» участков упрощаемой полилинии «рукавами» заданной ширины *d*. Алгоритм итеративно проходит по точкам и выполняет построение рукава. Если какая-то точка не попадает в заданные рамки, то в ней рукав изламывается. Часть полилинии, попавшей внутрь рукава, будет упрощена до отрезка, соединяющего первую и последнюю точки, попавшие в рукав (см. рис. 5).

Параметр алгоритма sleeve-fitting *d* задается равным половине минимального размера различимой детали *Minsize*, который установлен при старте работы предлагаемого алгоритма. Такое соотношение позволяет равномерно сокращать число точек в больших и маленьких ячейках.

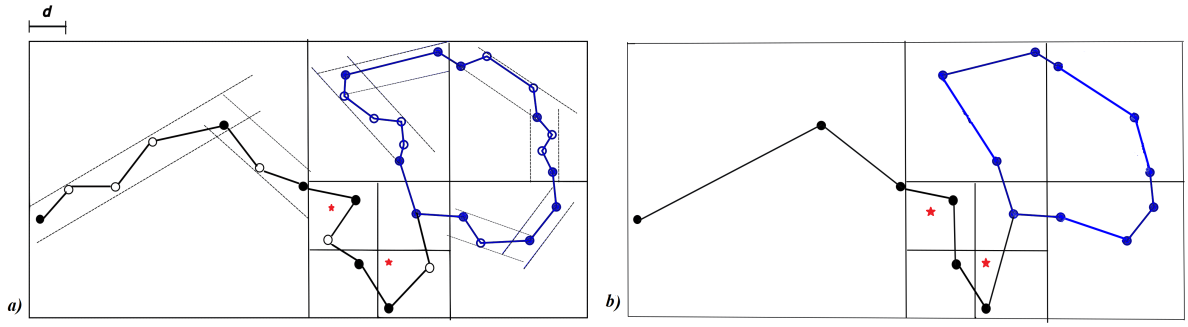


Fig. 5. Simplification process

Рис. 5. Процедура упрощения

В ячейках минимального размера может находиться несколько картографических объектов, причем разной геометрии. Следует отметить, что полигональный объект целиком в маленькой ячейке находиться не может, так как первым этапом картографической генерализации является отбор, где объекты слишком мелкие для целевого масштаба удаляются. Отсюда можно считать, что в ячейке минимального размера могут быть либо точечные, либо полилинейные объекты, представляющие собой как границы полигонов, так и собственно полилинии.

Если в ячейке минимального размера находится один объект, то для его представления выбирается одна точка, что оправданно малостью ячейки в целевом масштабе карты. В случае наличия нескольких объектов для каждой пары строится матрица девяти пересечений [21, 22], позволяющая решить вопрос об их взаимном расположении.

Если полилинейные объекты касаются или пересекаются, то в упрощенном виде сохраняется их точка касания (пересечения). Если совпадают, то для обоих объектов сохраняется линия, соединяющая точки на границах ячейки, остальные точки удаляются. Для непересекающихся точечных и полилинейных объектов отслеживается их взаимное расположение, чтобы точка оставалась в той же полуплоскости относительно полилинии.

Пример упрощения показан на рисунке 5.

2. Эксперименты и результаты

Тестирование разработанного алгоритма производилось на фрагментах карты Российской Федерации масштабов 1:500000 и 1:1000000. Все фрагменты карты проходили предобработку. К сожалению, качество цифровых пространственных данных, доступных в открытых источниках, оставляет желать лучшего.

Например, при оцифровке реки полилиния разбивается на отрезки от впадения одного притока до впадения другого. Каждый отрезок имеет свой числовой идентификатор, причем его значения для последовательных участков реки существенно отличаются. Объединить реку в единый объект (с одним и тем же идентификатором) можно по ее имени, а если имя не указано, то только вручную, средствами ГИС, непосредственно выделяя попарно те части объекта, которые надо соединить с одним и тем же идентификатором. Для построения структуры данных, на основе которой работает предлагаемый алгоритм согласованного упрощения, необходимо, чтобы один объект имел один и тот же идентификатор. Для оценки точности пространственного позиционирования будет использоваться модифицированное расстояние Хаусдорфа (MHD), введенное и исследованное в работе [23]. Оно определяется следующим образом. Пусть $A = \{a_1, a_2, \ldots, a_n\}$ и $B = \{b_1, b_2, \ldots, b_m\}$ — два множества точек, а $d(x, y)$ — евклидово расстояние между двумя точками. Среднее расстояние между A и B может быть вычислено по формуле:

$$\overline{d}(A, B) = \frac{1}{n} \sum_{a_i \in A} d(a_i, B),$$

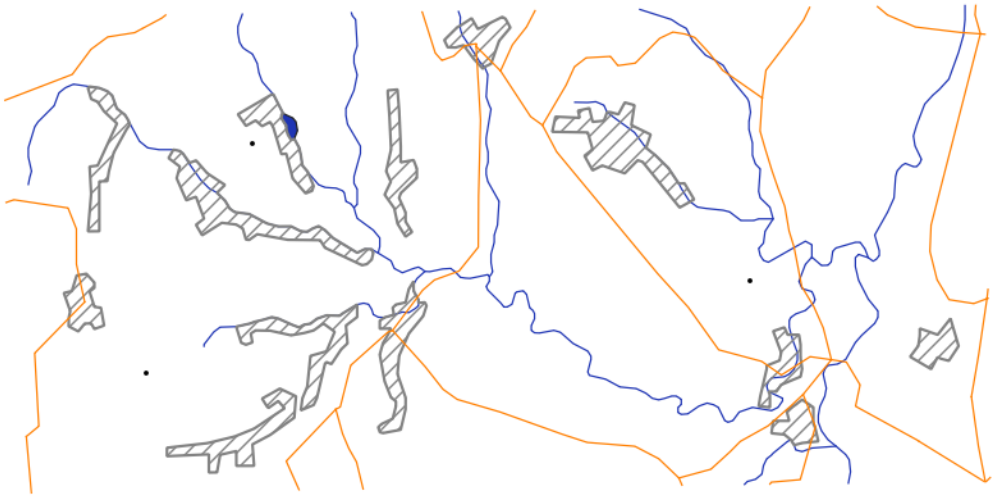


Fig. 6. A fragment of a map of the Voronezh region at a scale of 1:1000000

Рис. 6. Фрагмент карты Воронежской области масштаба 1:1000000

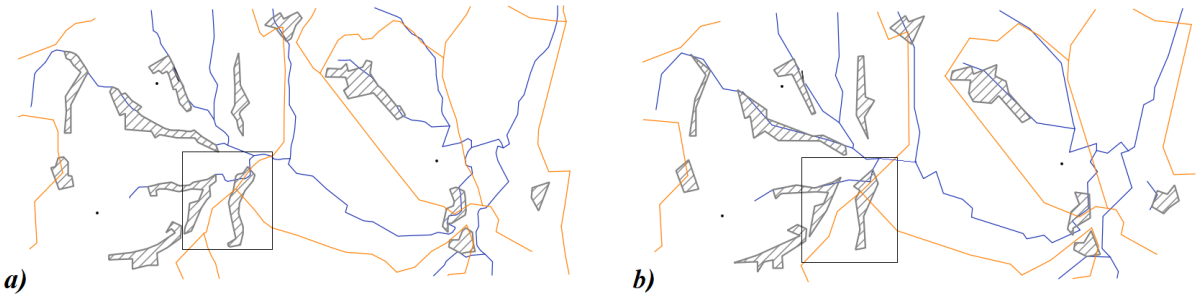


Fig. 7. The result of simplification by the proposed algorithm a) and the sleeve-fitting algorithm b)

Рис. 7. Результат упрощения предлагаемым алгоритмом а) и алгоритмом sleeve-fitting b)

где $d(a_i, B) = \min\limits_{b_j \in B} d(a_i, b_j)$. С помощью симметричной формулы вычисляется среднее расстояние между B и A:

$$\overline{d}(B, A) = \frac{1}{m} \sum_{b_j \in B} d(b_j, A),$$

где $d(b_j, A) = \min\limits_{a_i \in A} d(b_j, a_i)$. Тогда модифицированное расстояние Хаусдорфа будет вычисляться так:

$$MHD(A, B) = \max(\overline{d}(A, B), \overline{d}(B, A)).$$

Чем меньше MHD, тем ближе исходная полилиния к упрощенной. Использование среднего значения расстояния позволяет снизить влияние выбросов — единичных точек, сильно удаленных от исходного положения.

На рисунке 6 представлен фрагмент карты Воронежской области, включающий в себя реки, озера, дороги, границы населенных пунктов и уникальные растительные объекты, которые отображаются черными точками. Этот фрагмент был упрощен алгоритмом sleeve-fitting [20] (каждый слой отдельно) и предлагаемым алгоритмом согласованной генерализации.

Результаты упрощения приведены на рисунке 7.

Выделенный фрагмент был вынесен на рисунок 8. Очевидно, что при упрощении предлагаемым алгоритмом сохраняются правильные топологические отношения картографических объектов.

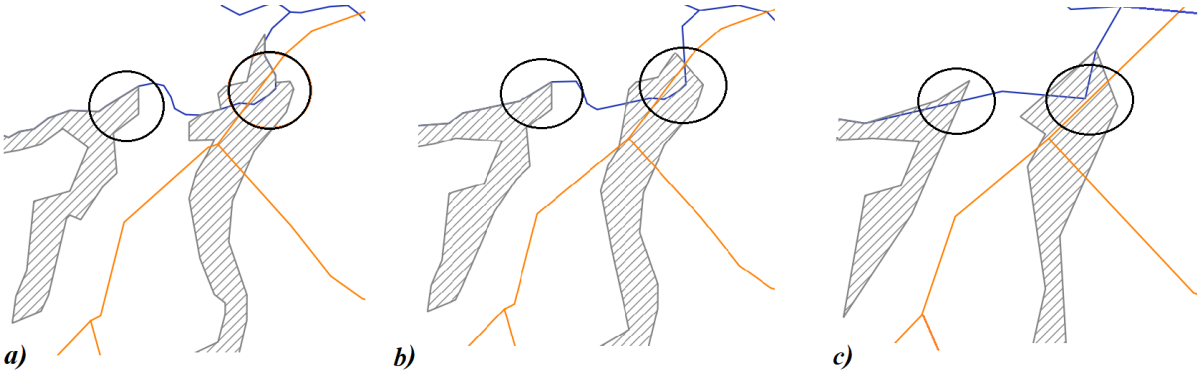


Fig. 8. Input data a), after simplification by the proposed algorithm b) and by the sleeve-fitting algorithm c)

Рис. 8. Исходные данные a), после упрощения предлагаемым алгоритмом b) и алгоритмом sleeve-fitting c)

Table 1. Modified Hausdorff distance between original and simplified objects

Таблица 1. Модифицированное расстояние Хаусдорфа между исходными и упрощенными объектами

Объекты	Исходное количество точек	Алгоритм согласованной генерализации		Алгоритм sleeve-fitting	
		Количество точек	MHD	Количество точек	MHD
Фрагмент карты масштаба 1: 1000000					
Реки	397	219	5	98	17
Водоемы	14	6	37	3	34
Границы населенных пунктов	335	231	14	129	16
Дороги	165	150	7	96	29
Фрагмент карты масштаба 1:500000					
Реки	1758	1018	4	382	8
Водоемы	614	324	15	176	21
Трубопроводы	120	112	1	55	5
Железные дороги	392	277	3	77	8

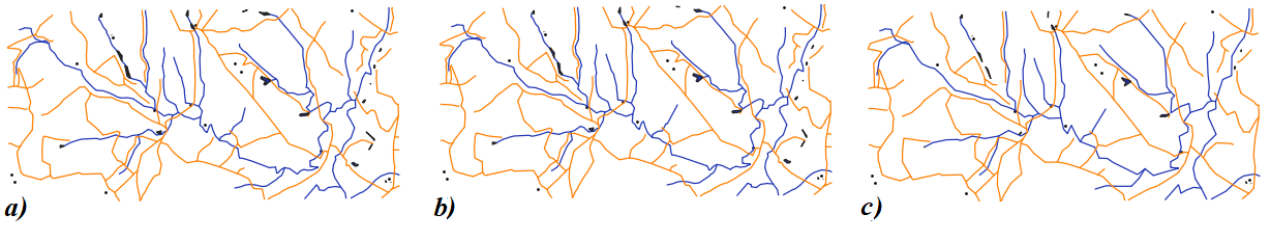


Fig. 9. Generalization results with the algorithm parameter a) 1000 m b) 1500 m c) 2000 m

Рис. 9. Результаты генерализации с параметром алгоритма a) 1000 м b) 1500 м c) 2000 м

В таблице (см. таблицу 1) приведены значения модифицированного расстояния Хаусдорфа между исходными и упрощенными объектами. Объекты, генерализованные с помощью предлагаемого алгоритма, ближе к исходным данным, но при этом разница количества точек между исходным

данными и результатом работы нашего алгоритма существенно меньше, чем аналогичная разница у алгоритма sleeve-fitting.

На рисунке 9 показано последовательное упрощение исходных данных формата 1:500000, представленных на рисунке 1, с параметром алгоритма равным 1 000 м, 1 500 м и 2 000 м.

Выводы

В работе предложен оригинальный подход к генерализации всей совокупности картографических объектов, сохраняющий их топологические отношения. В качестве перспектив дальнейших исследований следует назвать применение разработанной структуры данных для операций отбора и обобщения при мультимасштабном картографировании, а также всестороннее тестирование и, возможно, модификация предлагаемого алгоритма для упрощения карт крупных масштабов.

References

[1] W. Tobler, *Numerical map generalization*. Department of Geography, University of Michigan Ann Arbour, MI, USA, 1966, 78 pp.

[2] F. Töpfer and W. Pillewizer, “The principles of selection”, *Cartographic Journal*, vol. 3, no. 1, pp. 10–16, 1966.

[3] J. D. Perkal, “Proba obiektywnej generalizacji”, *Geodezia I Kartografia*, vol. 7, no. 2, pp. 130–142, 1958, in Polish.

[4] U. Ramer, “An iterative procedure for the polygonal approximation of plane curves”, *Computer Graphics and Image Processing*, vol. 1, no. 3, pp. 244–256, 1972.

[5] D. H. Douglas and T. K. Peucker, “Algorithms for the reduction of the number of points required to represent a digitized line or its caricature”, *Cartographica: the international journal for geographic information and geovisualization*, vol. 10, no. 2, pp. 112–122, 1973.

[6] J. S. Marino, “Identification of characteristic points along naturally occurring lines. an empirical study”, *The Canadian cartographer Toronto*, vol. 16, no. 1, 1979.

[7] G. Dutton, “Scale, sinuosity, and point selection in digital line generalization”, *Cartography and Geographic Information Science*, vol. 26, no. 1, pp. 33–54, 1999.

[8] A. Chehreghan and R. Ali Abbaspour, “Estimation of empirical parameters in matching of linear vector datasets: An optimization approach”, *Model. Earth Syst. Environ*, pp. 1029–1043, 3 2017.

[9] A. Chehreghan and R. Ali Abbaspour, “A geometric-based approach for road matching on multi-scale datasets using a genetic algorithm”, *Cartography and Geographic Information Science*, vol. 45, pp. 255–269, 3 2018.

[10] R. A. Finkel and J. L. Bentley, “Quad trees a data structure for retrieval on composite keys”, *Acta informatica*, vol. 4, pp. 1–9, 1974.

[11] A. Guttman, “R-trees: A dynamic index structure for spatial searching”, in *Proceedings of the 1984 ACM SIGMOD international conference on Management of data*, 1984, pp. 47–57.

[12] U. A. Kravchenko, *Information geomodeling: The problem of data and knowledge representation*, Abstract of the dissertation for the degree of Doctor of Technical Sciences. In Russian, 2013.

[13] G. Dettori and E. Puppo, “How generalization interacts with the topological and metric structure of maps”, in *Proceedings of the 7th International Symposium on Spatial Data Handling*, 1996, pp. 559–570.

[14] D. Rhind, “Generalization and realism with automated cartographic system”, *Canadian Cartographer*, vol. 10, no. 1, pp. 51–62, 1973.

[15] M. Monmonier, “Displacement in vector-and raster-mode graphics”, *Cartographica: The International Journal for Geographic Information and Geovisualization*, vol. 24, no. 4, pp. 25–36, 1987.

[16] P. M. Van Der Poorten and C. B. Jones, “Characterisation and generalisation of cartographic lines using delaunay triangulation”, *International Journal of Geographical Information Science*, vol. 16, no. 8, pp. 773–794, 2002.

[17] Z. Li and S. Openshaw, “Algorithms for automated line generalization1 based on a natural principle of objective generalization”, *International journal of geographical information systems*, vol. 6, no. 5, pp. 373–389, 1992.

[18] P. Raposo, “Scale-specific automated line simplification by vertex clustering on a hexagonal tessellation”, *Cartography and Geographic Information Science*, vol. 40, no. 5, pp. 427–443, 2013.

[19] L. Zhilin, *Algorithmic Foundation of Multi-Scale Spatial Representation*. Taylor & Francis Group, LLC, 2007, 282 pp.

[20] Z. Zhao and A. Saalfeld, “Linear-time sleeve-fitting polyline simplification algorithms”, in *Proceedings of AutoCarto*, vol. 13, 1997, pp. 214–223.

[21] M. Egenhofer and J. Herring, “Categorizing binary topological relations between regions, lines and points in geographic databases, the 9-intersection: Formalism and its use for naturallanguage spatial predicates”, *Santa Barbara CA National Center for Geographic Information and Analysis Technical Report*, vol. 94, pp. 1–28, 1990.

[22] V. G. Gorshkov, D. M. Murin, and O. P. Yakimova, “Research of models of topological relations of spatial objects”, *Modelirovanie i Analiz Informatsionnykh Sistem*, vol. 29, no. 3, pp. 154–165, 2022, in Russian.

[23] M.-P. Dubuisson and A. K. Jain, “A modified Hausdorff distance for object matching”, in *Proceedings of 12th international conference on pattern recognition*, vol. 1, 1994, pp. 566–568.

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 4, 2023
journal homepage: www.mais-journal.ru

THEORY OF DATA

Fast computation of cyclic convolutions and their applications in code-based asymmetric encryption schemes

A. N. Sushko[1], B. Y. Steinberg[1], K. V. Vedenev[1], A. A. Glukhikh[1], Y. V. Kosolapov[1]

DOI: 10.18255/1818-1015-2023-4-354-365

[1]Southern Federal University, 105/42 Bolshaya Sadovaya str., Rostov-on-Don, 344006, Russia.

MSC2020: 68P30; 68W99
Research article
Full text in English

Received November 6, 2023
After revision November 22, 2023
Accepted November 29, 2023

The development of fast algorithms for key generation, encryption and decryption not only increases the efficiency of related operations. Such fast algorithms, for example, for asymmetric cryptosystems on quasi-cyclic codes, make it possible to experimentally study the dependence of decoding failure rate on code parameters for small security levels and to extrapolate these results to large values of security levels. In this article, we explore efficient cyclic convolution algorithms, specifically designed, among other things, for use in encoding and decoding algorithms for quasi-cyclic LDPC and MDPC codes. Corresponding convolutions operate on binary vectors, which can be either sparse or dense. The proposed algorithms achieve high speed by compactly storing sparse vectors, using hardware-supported XOR instructions, and replacing modulo operations with specialized loop transformations. These fast algorithms have potential applications not only in cryptography, but also in other areas where convolutions are used.

Keywords: cyclic convolutions; fast algorithms; encryption schemes

INFORMATION ABOUT THE AUTHORS

Andrey N. Sushko	orcid.org/0009-0009-7528-8532. E-mail: andrew-sush@mail.ru Undergraduate Student.
Boris Y. Steinberg	orcid.org/0000-0001-8146-0479. E-mail: borsteinb@mail.ru Head of the Chair.
Kirill V. Vedenev	orcid.org/0000-0002-7893-655X. E-mail: vedenevk@gmail.com PhD Student.
Anton A. Glukhikh	orcid.org/0009-0005-0160-3609. E-mail: Antong.21072003@gmail.com Undergraduate Student.
Yury V. Kosolapov corresponding author	orcid.org/0000-0002-1491-524X. E-mail: itaim@mail.ru Associate professor, Ph.D.

For citation: A. N. Sushko, B. Y. Steinberg, K. V. Vedenev, A. A. Glukhikh, and Y. V. Kosolapov, "Fast computation of cyclic convolutions and their applications in code-based asymmetric encryption schemes", *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 354-365, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 4, 2023
сайт журнала: www.mais-journal.ru

THEORY OF DATA

Быстрое вычисление циклических сверток и их приложения в кодовых схемах асимметричного шифрования

А. Н. Сушко[1], Б. Я. Штейнберг[1], К. В. Веденев[1], А. А. Глухих[1], Ю. В. Косолапов [1]

DOI: 10.18255/1818-1015-2023-4-354-365

[1]Южный федеральный университет, ул. Большая Садовая, 105/42, Ростов-на-Дону, 344006, Россия.

УДК 004.421.4+004.051
Научная статья
Полный текст на английском языке

Получена 6 ноября 2023 г.
После доработки 22 ноября 2023 г.
Принята к публикации 29 ноября 2023 г.

Разработка быстрых алгоритмов генерации ключей, шифрования и дешифрования не только повышает эффективность соответствующих операций. Такие быстрые алгоритмы, например, для асимметричных криптосистем на квазициклических кодах, позволяют экспериментально исследовать зависимость вероятности ошибочного расшифрования от параметров кода для малых параметров безопасности и экстраполировать эти результаты на большие значения параметров безопасности. В этой статье мы исследуем эффективные алгоритмы циклической свертки, специально разработанные, в том числе, для использования в алгоритмах кодирования и декодирования квазициклических LDPC и MDPC кодов. Соответствующие свертки работают с двоичными векторами, которые могут быть как разреженными, так и плотными. Предлагаемые алгоритмы достигают высокой скорости за счет компактного хранения разреженных векторов, использования аппаратно поддерживаемых инструкций XOR и замены операций по модулю специализированными преобразованиями цикла. Эти быстрые алгоритмы имеют потенциальное применение не только в криптографии, но и в других областях, где используются свертки.

Ключевые слова: циклические свертки; быстрые алгоритмы; схемы шифрования

ИНФОРМАЦИЯ ОБ АВТОРАХ

Андрей Николаевич Сушко	orcid.org/0009-0009-7528-8532. E-mail: andrew-sush@mail.ru студент.
Борис Яковлевич Штейнберг	orcid.org/0000-0001-8146-0479. E-mail: borsteinb@mail.ru заведующий кафедрой.
Кирилл Владимирович Веденев	orcid.org/0000-0002-7893-655X. E-mail: vedenevk@gmail.com аспирант.
Антон Анатольевич Глухих	orcid.org/0009-0005-0160-3609. E-mail: Antong.21072003@gmail.com студент.
Юрий Владимирович Косолапов автор для корреспонденции	orcid.org/0000-0002-1491-524X. E-mail: itaim@mail.ru доцент, канд. техн. наук.

Для цитирования: A. N. Sushko, B. Y. Steinberg, K. V. Vedenev, A. A. Glukhikh, and Y. V. Kosolapov, “Fast computation of cyclic convolutions and their applications in code-based asymmetric encryption schemes”, *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 354-365, 2023.

Introduction

Convolution computation finds wide applications in digital signal processing [1], image processing [2], steganography [3], deep neural network training and inference [4], and other areas of applied mathematics. The Fourier transform can be utilized to compute convolutions: classical Fast Fourier Transform (FFT) algorithms such as Cooley-Tukey, Good-Thomas, Rader, and Winograd have a computational complexity of $O(n\log(n))$, where n is a length of vectors. However, these algorithms are limited to specific lengths n, such as powers of 2 or products of coprime numbers. In addition, these algorithms use complex numbers arithmetic that may require more memory than the original vectors. Moreover, when using the Fourier transform for convolution computation, the inverse Fourier transform must also be used. As memory access is the performance bottleneck for modern computers [5], the use of Fourier transform could only be beneficial for large-scale instances. For small vector lengths, and especially if the vector coordinates are Boolean, optimized direct convolution computations can be faster. Note that Boolean convolutions over cyclic groups find widespread applications across various computer science domains. One prominent area is error-correction codes, where many practical codes exhibit cyclic or quasi-cyclic properties. By leveraging convolutions over Galois fields, message encoding can be efficiently implemented for codes with cyclic or quasi-cyclic properties. Additionally, for specific codes such as Quasi-Cyclic Low-Density Parity-Check (QC-LDPC) and Quasi-Cyclic Moderate Density Parity Check Codes (QC-MDPC), convolutions play a crucial role in implementation of decoding algorithms. Moreover, convolutions are extensively utilized in recently proposed cryptographic protocols for public-key encryption and digital signatures. One notable example is the Bit-flipping Key Encapsulation (BIKE) [6], a post-quantum public-key encryption algorithm that utilizes random QC-MDPC codes. The convolutions involved in BIKE operate on both sparse and dense vectors over both two-element Galois field $\mathrm{GF}(2)$ and ring of integers $\mathbb{Z}$.

In the proposed paper, we delve into the optimization techniques for software implementations of convolutions used in BIKE and in the decoding of QC-MDPC codes. Specifically, we consider dense-by-sparse convolutions over $\mathrm{GF}(2)$ and $\mathbb{Z}$. Our proposed optimizations allow achieving high-speed performance through the compact storage of sparse vectors, leveraging hardware-supported vector-executed XOR (exclusive OR) instructions, and replacing the modulo operation with specialized loop transformations. These techniques aim to enhance the performance of the convolutions involved in software implementations of BIKE and decoding of QC-MDPC codes.

1. Cryptographic research motivation

Modern algorithms for asymmetric encryption and electronic digital signature are based on assumptions about the computational complexity of solving the discrete logarithm problem in a finite group of large order and the problem of factoring a large composite number into a product of two large prime numbers. However, it turned out that these assumptions are confirmed so far only for the computation model on a Turing machine, while for the quantum computation model, P. Shor's efficient algorithm for solving these problems is known. In 2016, the US National Institute of Standards and Technology (NIST) announced a competition to develop asymmetric encryption and digital signature algorithms that would be resistant to attacks carried out using quantum computing. One of the finalists in the NIST competition for an asymmetric encryption scheme is the BIKE cipher [6], whose security is based on the difficulty of decoding a random binary QC-MDPC code of length $2r$, $r \in \mathbb{N}$. Note, that some computational operations during encryption and decryption in BIKE can be implemented using cyclic convolutions. Recall, that for ring $\mathcal{R}$ and vectors $a, b \in \mathcal{R}^r$ the cyclic convolution $a \star b$ of $a = (a_0, \ldots, a_{r-1})$ and $b = (b_0, \ldots, b_{r-1})$ is defined as follows

$$c = a \star b = (c_0, \ldots, c_{r-1}),\ c_i = \sum_{j=0}^{r-1} a_{(i-j) \bmod r} \cdot b_j,\ i = 0, \ldots, r-1. \tag{1}$$

The cryptosystem BIKE uses convolutions over rings $\mathbb{Z}$ (vectors of integers) and GF(2) (binary vectors), so the notation $\star_i$ and $\star_b$ are used for the corresponding convolutions. Note, that for GF(2), the sum operator in (1) is $\oplus$, and in the case $\mathbb{Z}$ sum is common operation with integers. Multiplication is common operation with integers in both cases. To show which operations in BIKE can be calculated using convolutions, we briefly describe this cryptosystem.

Let $\mathrm{GF}(2) = \{0, 1\}$ be a Galois field with additive operation $\oplus$: $0 \oplus 0 = 1 \oplus 1 = 0$, $1 \oplus 0 = 0 \oplus 1 = 1$. Denote by $R_n = \mathrm{GF}(2)[x]/(x^r - 1)$ the cyclic factoring ring and let us consider the mapping $L : \mathrm{GF}(2)^r \to R_n$ such that for $a = (a_0, \ldots, a_{r-1}) \in \mathrm{GF}(2)^r$, $L(a) = \sum_{i=0}^{r-1} a_i x^i$. (Here and below, vector coordinates are numbered starting from zero.) The number of non-zero elements in vector $a \in \mathrm{GF}(2)^r$ we denote as $\mathrm{wt}(a)$. The secret key in the BIKE system is a pair of polynomials $(h_0, h_1) \in R_n^2$, where each polynomial is chosen randomly and equiprobably such that $\mathrm{wt}(L^{-1}(h_0)) = \mathrm{wt}(L^{-1}(h_1)) = w/2 \approx \sqrt{2r}/2$ and h_0 must be invertible in R_n. The public key is the polynomial $h = h_1 \cdot h_0^{-1}$, where multiplication is taken in R_n. The plain text is represented as a pair $(e_0, e_1) \in \mathrm{GF}(2)^r \times \mathrm{GF}(2)^r$, where $\mathrm{wt}(e_0) + \mathrm{wt}(e_1) = t \approx \sqrt{2r}$. The corresponding ciphertext $s \in \mathrm{GF}(2)^r$ is obtained by the rule

$$s = e_0 \oplus L^{-1}(L(e_1) \cdot h), \tag{2}$$

where for the binary vectors $a = (a_0, \ldots, a_{r-1})$ and $b = (b_0, \ldots, b_{r-1})$ the result of operation $a \oplus b$ is the binary vector $(a_0 \oplus b_0, \ldots, a_{r-1} \oplus b_{r-1})$. When decrypting, the appropriate decoder for QC-MDPC codes is used, which takes the vector s, the secret key (h_0, h_1) and returns the vector (e_0, e_1) or $\perp$ if decoding failure occurs.

The product $L(e_1) \cdot h$ in encryption rule (2) is a multiplication of polynomials in the R_n, which can be realized as a cyclic convolution of a sparse vector e_1 and a dense vector $L^{-1}(h)$. So the rule (2) can be rewritten as $s = e_0 \oplus (e_1 \star_b L^{-1}(h))$. Some operations in known decoders of QC-MDPC codes can also be implemented using cyclic convolutions. Such operations may include calculating the current value of the syndrome and unsatisfied parity check (UPC) counters (see pseudo code of some known decoding algorithms for QC-MDPC-codes for example in [7]). For example, in Algorithm 1 the pseudo code of *BitFlip* decoding algorithm is shown, where cyclic convolutions are calculated at each iteration: two convolutions over GF(2) (calculating the current value of the syndrome s') and two over $\mathbb{Z}$ (calculating the UPCs upc_0 and upc_1).

It is worth noting that fast convolution algorithms can speed up the key generation time of a system BIKE when using combinations of iterative and majority logic decoding algorithms, as considered in [8, 9]. Indeed, to ensure a low decoding failure rate (DFR) in such systems, it is necessary to select keys with a large majority margin. Therefore, the task of quickly calculating this boundary for randomly generated keys is relevant. It is known that such a margin for each key can be calculated using cyclic convolution over $\mathbb{Z}$.

Therefore, optimizing convolution algorithms can improve the speed of key generation, encryption and decryption operations in BIKE. Note that, the similar techniques can also be employed in other modern cryptographic primitives like HQC, NTRU, LWE, etc. As the length of vectors is not a power of 2 and is not a product of coprime numbers in many encryption schemes, the use of FFT algorithms for convolution calculations is impractical and new optimization algorithms are needed.

2. Fast cyclic convolution algorithms

This section presents algorithms for optimizing convolution computation. The algorithms are implemented in C language. It is assumed that the convolution is calculated for vectors of length n. The direct implementation *GF2_noonpt* of convolution over GF(2) is presented in Listing 1. The direct implementation *Z_noopt* of convolution over $\mathbb{Z}$ looks similar without taking modulo 2.

2.1. Generic optimizations for GF(2) and $\mathbb{Z}$

The direct implementation of convolution can be improved. Indeed, all convolution algorithms used in the decoder 1 involve at least one sparse vector. To store a sparse binary vector, we will use an integer

Algorithm 1. *BitFlip*: iterative decoder for QC-MDPC codes

Input: The vector $s \in \mathrm{GF}(2)^r$, the pair $(h_0, h_1) \in R_n \times R_n$ and the number of iterations I.

Output: $(e_0, e_1) \in \mathrm{GF}(2)^r \times \mathrm{GF}(2)^r$ or fail message $\perp$.

$u \leftarrow \mathbf{0}, v \leftarrow \mathbf{0}$; /* Initialize vectors u and v by zero value $\mathbf{0} \in \mathrm{GF}(2)^r$ */
$s' \leftarrow s$;
for $i = 1, \ldots, I$ **do**
  compute threshold T by s';
  $upc_0 \leftarrow s' \star_i L^{-1}(h_0)$, $upc_1 \leftarrow s' \star_i L^{-1}(h_1)$; /* Two cyclic convolutions over $\mathbb{Z}$ */
  for $j = 0, \ldots, r-1$ **do**
    if $upc_{0,j} \geqslant T$ **then**
      $u_j \leftarrow u_j \oplus 1$; /* Flipping jth bit in u */
    if $upc_{1,j} \geqslant T$ **then**
      $v_j \leftarrow v_j \oplus 1$; /* Flipping jth bit in v */
  $s' \leftarrow s \oplus (u \star_b L^{-1}(h_0)) \oplus (v \star_b L^{-1}(h_1))$; /* Two cyclic convolutions over GF(2) */
  if $s' = \mathbf{0}$ **then**
    return (u, v)
return $\perp$;

Listing 1. Direct implementation of convolution over GF(2) according to (1)

```
void GF2_noopt(unsigned short n,
               unsigned short *a,
               unsigned short *b,
               unsigned short *c) {
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            c[j] = (c[j] + b[(j - i + n) % n] * a[j]) % 2;
        }
    }
}
```

vector called *inda*, where each element will contain the position of a non-zero element (i.e., 1) in the original vector, in ascending order. Let's denote the length of the resulting vector as *arrSize*, which is smaller than n. For example, if the vector a looked like this $a = (0, 1, 0.0, 0, 1, 0, 1)$, then the vector *inda* would be $(1, 5, 7)$. Therefore, to accelerate the computations, the sparse vector a has been replaced with the *inda* vector of size *arrSize* (see listing for *GF2_opt1* in Listing 2 for convolutions over GF(2)). Using this new vector, we can reduce the number of iterations and eliminate the multiplication operation. The code *Z_opt1* for computing convolution over $\mathbb{Z}$, excluding the modulo 2, looks similar to *GF2_opt1*.

2.2. Optimizations for convolutions over GF(2)

In C language there are bitwise operators ~, &, |, and ^, which correspond to bitwise negation, bitwise AND, bitwise OR, and bitwise XOR operations, respectively. For example, the XOR operation (operator ^) can be used to replace addition of two numbers and taking the modulo 2 (see the listing *GF2_opt2* in Listing 3). In the listings *GF2_noopt*, *GF2_opt1* and *GF2_opt2* the bit vector b is represented as an array of type *unsigned short*, where each element occupies 16 bits but stores only one bit of useful information. Let us

Listing 2. Optimization 1 over GF(2): using sparsity of one vector

```
void GF2_opt1(unsigned short n,
              int arrSize,
              const unsigned short *inda,
              unsigned short *b,
              unsigned short *c) {
    for (int i = 0; i < arrSize; i++) {
        for (int j = 0; j < n; j++) {
            c[j] = (c[j] + b[(j - inda[i] + n) % n]) % 2;
        }
    }
}
```

Listing 3. Optimization 2 over GF(2): using XOR in *GF2_opt1* instead of sum modulo 2

```
void GF2_opt2(unsigned short n,
              int arrSize,
              const unsigned short *inda,
              unsigned short *b,
              unsigned short *c) {
    for (int i = 0; i < arrSize; i++) {
        for (int j = 0; j < n; j++) {
            c[j] = (c[j] ^ b[(j - inda[i] + n) % n]);
        }
    }
}
```

transform vector b in such a way that each bit in the vector corresponds to the corresponding element of the original vector. The resulting vector is denoted by $b2$ and has a length of $m = \lceil n/16 \rceil$. Thus, the first element of $b2$ contains the first 16 elements of vector b. Note that 16 is the length of *unsigned short* type in bits. However, since the data type can be changed to any other type of unsigned number, the code uses the variable $elementSize = 16$. Let's replace the dense vector b with $b2$. Since the convolutions used involve adding a cyclically shifted vector, we need the ability to shift the vector by n bits to the right. The output vector *res* is also compact, just like $b2$. Due to the changes in the input and output vectors, we need to be able to assemble new elements of the output vector. For this purpose, we will introduce two arrays of masks: *arrSize* and *negMasks*. Each element in the *arrSize* array represents a number whose binary representation contains *elementSize* ones, shifted to the left by the element's index. Therefore, $masks[0]$ is a number with *elementSize* ones in its binary representation, $masks[1]$ has $elementSize - 1$ ones, and so on. Similarly, *negMasks* is an array where each element can be obtained by bitwise negating the corresponding element in the *arrSize* array. Variables *mod*, *modNeg* and *it* are service variables for joining vector concatenation. In particular, *mod* represents the remainder of dividing $inda[i]$ by *elementSize*, which is necessary to select the appropriate mask and correctly shift the element since the shift may not always be a multiple of *elementSize*. This change allows for the computation of 16 elements in a single iteration of the outer loop, rather than just 1 element, resulting in significant acceleration. An optimized version of the convolution calculation is shown in Listing 4.

Listing 4. Optimization 3 over GF(2): use compact representation of vectors in *GF2_opt2*

```
void GF2_opt3(unsigned short n,
              int arrSize, int m,
              const unsigned short *inda,
              unsigned short *b2,
              unsigned short *res) {
    unsigned short modG=n % elementSize;
    unsigned short modGNeg=(elementSize - modG) % elementSize;
    unsigned short lastElementShifted =
            (b2[m - 1] >> modGNeg) +
            ((b2[m - 2] & negMasks[modGNeg]) << modG);
    for (int i = 0; i < arrSize; i++) {
        unsigned short mod = inda[i] % elementSize;
        unsigned short modNeg = (elementSize - mod) % elementSize;
        int start = inda[i] / elementSize;
        res[start] = res[start] ^ ((b2[0] >> mod) +
                ((lastElementShifted & negMasks[mod]) << modNeg));
        int j = 1;
        for (int it = start + 1; it < m; it++, j++) {
            res[it] = res[it] ^ (((b2[j]) >> mod)+
                                 ((b2[j-1] & negMasks[mod]) << modNeg));
        }
        int targetElement = (start - 1) * elementSize + n - inda[i] + 15;
        int modBack = targetElement % elementSize;
        int backStartTarget = targetElement / elementSize;
        int newMod = elementSize - modBack - 1;
        int newModNeg = (elementSize - newMod) % elementSize;
        j = backStartTarget;
        for (int it = start - 1; it >= 0; it--, j--) {
            res[it] = res[it] ^ (b2[j] >> newMod) ^
                    ((b2[j - 1] & negMasks[newMod]) << newModNeg);
        }
    }
    res[m - 1] = res[m - 1] & masks[modGNeg];
}
```

2.3. Optimizations for convolutions over $\mathbb{Z}$

To speed up the calculation of convolution over $\mathbb{Z}$, we use a partition of the iteration space into two triangular ones to get rid of the operation of taking the modulus (see Listing 5).

3. Experimental results

Testing for the correctness of the decoder 1 implementation was performed using known answer tests for two sets of parameters (r, w, t) of BIKE system: $(r = 12323, w = 142, t = 134)$ and $(r = 24659, w = 206, t = 199)$. Note that these parameters sets correspond to the cases BIKE Level-1 and BIKE Level-3. Performance testing was performed with the same initial states of the pseudorandom number generators. Testing was carried out both as a separate evaluation of the convolution operation and as part of the decoder 1 for

Listing 5. Optimization 2 over $\mathbb{Z}$: rearrange loops in *Z_opt1*

```
void Z_opt2(unsigned short n,
            int arrSize,
            const unsigned short *inda,
            const unsigned short *b,
            unsigned short *c) {
    unsigned short *pointer_c, *end = inda + arrSize;
    int j, new_j;
    for (unsigned short *i = inda; i != end; i++) {
        pointer_c = c;
        j = 0;
        new_j = n - *i;
        for (; j < *i; j++) {
            *pointer_c = (*pointer_c + b[new_j]);
            new_j++;
            pointer_c++;
        }
        new_j = j - *i;
        for (; j < n; j++) {
            *pointer_c = (*pointer_c + b[new_j]);
            new_j++;
            pointer_c++;
        }
    }
}
```

Table 1. CPU computer specification

Manufacturer	AMD
Number of cores	6
Number of threads	12
Frequency	3.60 Ghz / 3600 Mhz
Turbo Core	4.20 Ghz / 4200 Mhz
L1 cache	384Kb (6 x 32Kb + 6 x 32Kb)
L2 cache	3Mb (6 x 512Kb)
L3 cache	32Mb
Core (architecture)	Matisse (Zen2, x86-64)
Process	7 nm
PCIe controller	PCI Express 4.0 (16 lines)

number of iteration $I = 100$; total number of tests is 1 000. In the experiments, the threshold T, used in the decoder 1, was not dynamically calculated, but was specified by an appropriate constant. All measurements were performed using the CPU with the parameters, presented in the Table 1.

The program code was compiled using *clang* compiler for C/C++. For each set of parameters, the running time of the algorithms *GF2_noopt*, *GF2_opt1*, *GF2_opt2*, *GF2_opt3*, *Z_noopt*, *Z_opt1* and *Z_opt2* was estimated without compiler optimization, as well as using optimization options *O3* and *Ofast*. Operating speed is given

Table 2. Average performance for $(r = 12323, w = 142, t = 134)$

Operation	*O0*		*O3*		*Ofast*	
	seconds	cycles	seconds	cycles	seconds	cycles
GF2_noopt	0.228056	228056	0.224785	224784	0.220196	220196
GF2_opt1	0.001029	1029	0.001094	1094	0.001035	1035
GF2_opt2	0.000742	742	0.000748	748	0.000792	792
GF2_opt3	0.000017	17	0.000017	17	0.000017	17
Z_noopt	0.237824	237824	0.239727	239727	0.246001	246001
Z_opt1	0.000810	810	0.000862	862	0.000815	815
Z_opt2	0.000043	43	0.000043	43	0.000044	44
One iter. of alg. (1) (no opt.)	0.933065	933065	0.900846	900845	0.880317	880317
One iter. of alg. (1) (max. opt.)	0.000156	155	0.000153	153	0.000150	150

Table 3. Average performance for $(r = 24659, w = 206, t = 199)$

Operation	*O0*		*O3*		*Ofast*	
	seconds	cycles	seconds	cycles	seconds	cycles
GF2_noopt	0.759139	759139	0.758156	758156	0.750196	750196
GF2_opt1	0.003079	3079	0.003022	3022	0.002988	2988
GF2_opt2	0.002202	2202	0.002172	2172	0.002171	2171
GF2_opt3	0.000047	47	0.000043	43	0.000041	41
Z_noopt	0.737514	737514	0.714715	714714	0.716012	716012
Z_opt1	0.002426	2426	0.002381	2381	0.002354	2354
Z_opt2	0.000170	170	0.000125	125	0.000122	122
One iter. of alg. (1) (no opt.)	2.891910	2891910	2.884335	2884334	2.907707	2907706
One iter. of alg. (1) (max. opt.)	0.000409	409	0.000398	397	0.000403	402

in seconds and processor cycles. Evaluation of the influence of the applied optimizations in the calculation of convolutions gave the results shown in Tables 2 and 3.

Experimental results show that non-optimized convolution calculations over rings GF(2) and $\mathbb{Z}$ have no significant difference in computation speed. This is due to the fact that both in the case GF(2) and in the case $\mathbb{Z}$, the elements are represented by the same data type (unsigned short). The transition to special representations leads, on the one hand, to a significant acceleration of calculations, and on the other hand, the calculation speed already depends on the ring $\mathcal{R}$. It is also clear that compiler optimization makes it possible to speed up the calculation of convolutions for both rings only in the case $r = 24659$. In the case of estimating the speed of one iteration of the decoder 1, a slight acceleration of the work is observed when using compiler optimization.

4. Fast convolutions in security evaluation of BIKE

The security of cryptographic algorithms is defined as the ability to withstand specific unauthorized actions by an attacker. For example, if a cipher is resistant to finding the plaintext by one ciphertext, then the cipher is OW-secure (the cipher has the One-Way property). If the attacker cannot distinguish which of the two plaintexts m_0 or m_1 corresponds to the given ciphertext c, then the cipher is IND-CPA secure (INDistinguishable under Chosen Plain text Attack). The strongest ciphers are those with IND-CCA security (INDistinguishable under Chosen Cipher text Attack), when the attacker cannot distinguish which of the two plaintexts m_0 or m_1 corresponds to the given ciphertext c, even if the attacker can send requests to the decryption oracle (but cannot send c). The cipher is said to have security level λ (OW, IND-CPA, IND-CCA) if the probability of success of the corresponding attack does not exceed $1/2^{\lambda}$. For example, currently ciphers with a security level of $\lambda \geqslant 128$ are considered secure.

The disadvantage of BIKE system is that for a QC-MDPC code, the decoder may incorrectly decode the received vector s, which means that a legitimate user of the BIKE system cannot decrypt the message (e_0, e_1). Thus, BIKE is characterized by a non-zero decoding failure rate (DFR). However, as shown in the work [10] for binary case and then in [9] for q-ary case, with a high DFR an reaction attack is possible that allows to find the secret key of the cryptosystem BIKE. Thus, to guarantee the IND-CCA security of the BIKE system with the security parameter λ, the DFR value must be less than $1/2^\lambda$. However, at present, for the fast decoder from [6], a theoretical estimate on DFR has not been obtained, and an experimental study of this probability for $\lambda = 128$ means at least 2^{128} experiments (one experiment includes generating a QC-MDPC code, encoding a random vector, and decoding), which is currently computationally impossible. Therefore, IND-CCA security cannot yet be guaranteed for BIKE. Note that in [6] the IND-CCA security of the BIKE is proved under the assumption that DFR is less than $1/2^\lambda$ at security level λ. There are a number of approaches to assessing DFR. In [11], a lower theoretical DFR estimate was obtained for an ML decoder (Maximum Likelihood decoder), which differs from the iterative decoder from [6]. However, this estimate may be redundant: the real DFR may be lower when using modern fast decoders. The second approach to DFR estimation is experimental estimation for small code parameters with subsequent extrapolation to a higher level of security. In this case, extrapolation may lead to the choice of falsely strong parameters of the cryptosystem: the real DFR may be higher than the estimate obtained using an extrapolation. This is due to the fact that there is an inflection point of the decoding error probability versus code length curve: there is such a code length, starting from which the rate of decrease in the error probability decreases. The third approach is an extension of the second and is related to the study of the inflection point. However, it is not computationally possible to estimate the inflection point for $\lambda = 128$, as noted above. In this regard, in [12, 13] the inflection point is studied only for $\lambda = 20$, and for large values of λ it is proposed to perform extrapolation. The study, including experimental, of the inflection point for values of λ exceeding 20, seems to be an urgent task, since this can allow more accurate extrapolation to the values of λ corresponding to the values recommended in practice ($\lambda = 128, 192, 256$). Note that the efficient implementation of the BIKE cryptosystem is beneficial for obtaining estimates of the probability of decryption errors [11–13]. The parallel computations mentioned in [12, 13] alone may not provide significant acceleration. The performance of programs largely depends on data localization [14]. Moreover, one should not rely on good code optimization by an optimizing compiler [15]. One of the main engineering tasks in such a study is the efficient implementation of algorithms for encoding and decoding the QC-MDPC code, which makes it possible to carry out the study in a reasonable time. It seems that the greatest computational resources are consumed during encoding and decoding, while the generation of QC-MDPC code does not require large computational costs. As follows from the description above, the encoder and decoder of the QC-MDPC code can be implemented based on the calculation of convolutions. That is why the study of ways to speed up the calculation of convolutions is important in the study of the cryptosystem BIKE. The use of fast implementations of convolutions developed in this work will make it possible to estimate the time required to conduct 2^λ experiments on a single processor with parameters (r, w, t) corresponding to the given security level λ. For example, with $\lambda = 30$, the experiment time on one processor will be about 24 days, taking into account that one decoder operation spends 0.000409 seconds (taken from Table 3) and the decoder uses 5 iterations when decoding. This is a rough estimate, since, on the one hand, for $\lambda = 30$ the decoder operating time will be less than 0.000409, but on the other hand, during decoding, threshold T are usually calculated dynamically (see decoder 1), which can only increase the time of one decoding iteration. However, such approach allows us to estimate in advance the value of λ for which experiments can be carried out under conditions of limited computing and time resources.

Conclusion

It seems that further acceleration of convolutions can be achieved through, for example, the use of PCLMULQDQ instruction of the processor, which performs the multiplication of polynomials of degree no higher than 63. This approach is used, for example, in [16] to accelerate decoder for QC-MDPC codes. Note that the relevance of researching ways to optimize the QC-MDPC encoding/decoding algorithms is associated not only with the problems of accelerating key generation/encryption/decryption algorithms and refining security of BIKE system. It seems that the results of such study can make it possible to formulate requirements for microcircuits that implement these algorithms in hardware, and these results can be transferred to versions of the BIKE cryptosystem, where quasi-cyclic or quasi-group codes over finite fields of higher order are used. The presented algorithms for fast vector convolution, can also be used in other areas as well. For example, it seems that these algorithms can accelerate speed of data embedding in digital images in adaptive steganographic algorithms.

References

[1] T. Holton, *Digital signal processing: Principles and applications*. Cambridge University Press, 2021, 1058 pp.

[2] D. S. Taubman, M. W. Marcellin, and M. Rabbani, "JPEG2000: Image compression fundamentals, standards and practice", *Journal of Electronic Imaging*, vol. 11, no. 2, pp. 286–287, 2002.

[3] V. Holub, J. Fridrich, and T. Denemark, "Universal distortion function for steganography in an arbitrary domain", *EURASIP Journal on Information Security*, vol. 2014, no. 1, p. 1, 2014.

[4] Intel. "Intel® oneAPI Deep Neural Network Library". (2023), [Online]. Available: https://software.intel.com/content/www/us/en/develop/articles/intel-mkl-dnn-part-1-library-overview-and-installation.html.

[5] N. R. Council *et al.*, *Getting up to speed: The future of supercomputing*. National Academies Press, 2005, 306 pp.

[6] N. Aragon *et al.*, *BIKE: Bit Flipping Key Encapsulation*, Submission to the NIST post quantum standardization process, Dec. 2017. [Online]. Available: https://hal.science/hal-01671903.

[7] T. B. Paiva and R. Terada, "Faster constant-time decoder for MDPC codes and applications to BIKE KEM", *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2022, no. 4, pp. 110–134, 2022.

[8] P. Santini, M. Battaglioni, M. Baldi, and F. Chiaraluce, "Analysis of the error correction capability of LDPC and MDPC codes under parallel bit-flipping decoding and application to cryptography", *IEEE Transactions on Communications*, vol. 68, no. 8, pp. 4648–4660, 2020.

[9] K. Vedenev and Y. Kosolapov, "Theoretical analysis of decoding failure rate of non-binary QC-MDPC codes", in *Code-Based Cryptography*, Springer, 2023, pp. 35–55.

[10] Q. Guo, T. Johansson, and P. S. Wagner, "A key recovery reaction attack on QC-MDPC", *IEEE Transactions on Information Theory*, vol. 65, no. 3, pp. 1845–1861, 2018.

[11] M. Baldi, A. Barenghi, F. Chiaraluce, G. Pelosi, and P. Santini, "Performance bounds for QC-MDPC codes decoders", in *Code-Based Cryptography Workshop*, Springer, 2021, pp. 95–122.

[12] S. Arpin, T. R. Billingsley, D. R. Hast, J. B. Lau, R. Perlner, and A. Robinson, “A study of error floor behavior in QC-MDPC codes”, in *International Conference on Post-Quantum Cryptography*, Springer, 2022, pp. 89–103.

[13] S. Arpin, T. R. Billingsley, D. R. Hast, J. B. Lau, R. Perlner, and A. Robinson. “Raw data and decoder for the paper ”a study of error floor behavior in QC-MDPC codes””. (2022), [Online]. Available: https://github.com/HastD/BIKE-error-floor.

[14] A. Vasilenko, V. Veselovskiy, E. Metelitsa, N. Zhivykh, B. Steinberg, and O. Steinberg, “Precompiler for the acelan-compos package solvers”, in *Parallel Computing Technologies: 16th International Conference, PaCT 2021, Kaliningrad, Russia, September 13–18, 2021, Proceedings 16*, Springer, 2021, pp. 103–116.

[15] Z. Gong *et al.*, “An empirical study of the effect of source-level loop transformations on compiler stability”, *Proceedings of the ACM on Programming Languages*, vol. 2, no. OOPSLA, pp. 1–29, 2018.

[16] N. Drucker and S. Gueron, “A toolbox for software optimization of QC-MDPC code-based cryptosystems”, *Journal of Cryptographic Engineering*, vol. 9, no. 4, pp. 341–357, 2019.

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 4, 2023
journal homepage: www.mais-journal.ru

COMPUTING METHODOLOGIES AND APPLICATIONS

Modeling the influence of external influences on the process of automated landing of a UAV-quadcopter on a moving platform using technical vision

A. V. Ryabinov[1], A. I. Saveliev[1], D. A. Anikin[1]

DOI: 10.18255/1818-1015-2023-4-366-381

[1]St. Petersburg Federal Research Center of the Russian Academy of Sciences (SPC RAS), 39, 14th Line, VI, St. Petersburg, 199178, Russia.

MSC2020: 68T40
Research article
Full text in Russian

Received November 7, 2023
After revision November 23, 2023
Accepted November 29, 2023

This article describes a series of experiments in the Gazebo simulation environment aimed at studying the influence of external weather conditions on the automatic landing of an unmanned aerial vehicle (UAV) on a moving platform using computer vision and a previously developed control system based on PID and polynomial controllers. As part of the research, methods for modeling external weather conditions were developed and landing tests were carried out simulating weather conditions such as wind, light, fog and precipitation, including their combinations. In all experiments, successful landing on the platform was achieved; during the experiments, landing time and its accuracy were measured. The graphical and statistical analysis of the obtained results revealed the influence of illumination, precipitation and wind on the UAV landing time, and the introduction of wind into the simulation under any other external conditions led to the most significant increase in landing time. At the same time, the study failed to identify a systemic negative influence of external conditions on landing accuracy. The results obtained provide valuable information for further improvement of autonomous automatic landing systems for UAVs without the use of satellite navigation systems.

Keywords: unmanned aerial vehicle; quadcopter; marker detection; automated landing; computer vision; navigation

INFORMATION ABOUT THE AUTHORS

Artyom V. Ryabinov	orcid.org/0000-0002-3572-4493. E-mail: ryabinov.a@iias.spb.su Programmer.
Anton I. Saveliev corresponding author	orcid.org/0000-0003-1851-2699. E-mail: saveliev@iias.spb.su Senior Researcher.
Dmitriy A. Anikin	orcid.org/0009-0007-6998-5687. E-mail: anikin.d@iias.spb.su Programmer.

For citation: A. V. Ryabinov, A. I. Saveliev, and D. A. Anikin, "Modeling the influence of external influences on the process of automated landing of a UAV-quadcopter on a moving platform using technical vision", *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 366-381, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 4, 2023
сайт журнала: www.mais-journal.ru

Моделирование влияния внешних воздействий на процесс автоматизированной посадки БпЛА-квадрокоптера на подвижную платформу с использованием технического зрения

А. В. Рябинов[1], А. И. Савельев[1], Д. А. Аникин[1]

DOI: 10.18255/1818-1015-2023-4-366-381

[1]Санкт-Петербургский Федеральный исследовательский центр Российской академии наук, 14-я линия В. О. д. 39, г. Санкт-Петербург, 199178, Россия.

УДК 623.746.-519
Научная статья
Полный текст на русском языке

Получена 7 ноября 2023 г.
После доработки 23 ноября 2023 г.
Принята к публикации 29 ноября 2023 г.

В данной статье проводится описание серии экспериментов в симуляционной среде Gazebo, направленных на исследование влияния внешних погодных условий на автоматическую посадку беспилотного летательного аппарата (БпЛА) на движущуюся платформу с использованием компьютерного зрения и разработанной ранее системы управления, основанной на ПИД и полиномиальных регуляторах. В рамках исследования разработаны методы моделирования внешних погодных условий, и проведены тесты посадки с имитацией таких погодных условий, как ветер, освещенность, туман и осадки, включая их комбинации. Во всех экспериментах была достигнута успешная посадка на платформу, в ходе экспериментов измерялось время посадки и ее точность. Проведенный графический и статистический анализ полученных результатов выявил влияние освещенности, осадков и ветра на время посадки БпЛА, а введение ветра в симуляцию при любых других внешних условиях привело к наиболее значительному увеличению времени посадки. При этом в ходе исследования не удалось выявить системного негативного влияния внешних условий на точность посадки. Полученные результаты представляют ценную информацию для дальнейшего совершенствования систем автономной автоматической посадки БпЛА без использования спутниковых систем навигации.

Ключевые слова: беспилотный летательный аппарат; квадрокоптер; детектирование маркера; автоматизированная посадка; компьютерное зрение; навигация

ИНФОРМАЦИЯ ОБ АВТОРАХ

Артём Валерьевич Рябинов	orcid.org/0000-0002-3572-4493. E-mail: ryabinov.a@iias.spb.su программист.
Антон Игоревич Савельев автор для корреспонденции	orcid.org/0000-0003-1851-2699. E-mail: saveliev@iias.spb.su старший научный сотрудник.
Дмитрий Андреевич Аникин	orcid.org/0009-0007-6998-5687. E-mail: anikin.d@iias.spb.su программист.

Для цитирования: A. V. Ryabinov, A. I. Saveliev, and D. A. Anikin, “Modeling the influence of external influences on the process of automated landing of a UAV-quadcopter on a moving platform using technical vision”, *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 366-381, 2023.

Введение

В современном мире наблюдается растущий спрос на автономные технологии и беспилотные системы, которые находят широкое применение в различных областях, таких как сельское хозяйство [1], [2], транспортная промышленность [3], [4], картографирование местности [5], [6], обследование линий электропередач [7], сейсмология [8]. Особый интерес представляют мультироторные БпЛА с четырьмя винтами, известные как квадрокоптеры. Они обладают рядом преимуществ, таких как возможность вертикального взлета и посадки, что делает их идеальными для работы в условиях ограниченного пространства. Квадрокоптеры также отличаются высокой маневренностью, способностью летать в любом направлении и поворачивать на месте. Это делает их идеальными для выполнения задач в городских условиях, таких как доставка товаров, съемка видео, инспекция зданий и т. д.

Для разработчиков мультироторных беспилотных летательных аппаратов (БпЛА) одна из серьезных проблем — создание системы автоматической посадки на заданную точку, которая бы адаптировалась к различным условиям, таким как ветер, освещение, препятствия и т. д. Сейчас большинство коммерческих и открытых систем автопилотирования используют систему глобального позиционирования (Global Positioning System, GPS) для посадки, но это имеет свои недостатки. Для посадки по GPS БпЛА должен иметь спутниковый сигнал и достаточное количество места для посадки. Однако в некоторых случаях GPS может быть недоступен или иметь погрешность, например, в городских зонах или рядом с высокими зданиями. Это может привести к неудачной посадке и повреждениям БпЛА. К тому же посадка по GPS не гарантирует высокую точность и не учитывает возможные изменения в окружающей среде в процессе посадки. Еще одна проблема — это то, что иногда нужно садиться не на неподвижную платформу, а на движущуюся (например, другое роботизированное средство). Перспективным для решения этих проблем выступает использование технологий компьютерного зрения, которые позволят БпЛА распознавать место посадки и анализировать ситуацию в реальном времени, что увеличит точность и избавит от необходимости использования GPS. В предыдущем исследовании [9] была изучена возможность реализации автоматической системы посадки БпЛА-квадрокоптера на заранее определенную подвижную посадочную платформу в условиях различной высоты полета, проведена разработка и апробация соответствующих алгоритмов в ходе экспериментов по моделированию автоматизированной посадки с различной высоты в среде Gazebo (https://gazebosim.org/about) в идеальных условиях. Текущее же исследование фокусируется на влиянии различных гетерогенных погодных условий (ветер, освещенность, дождь, туман) и их комбинаций на процесс посадки. Была проведена серия экспериментов, включающая в себя посадку без внешних условий как контрольную, и ряд посадок с различными внешними условиями, в ходе которых были получены и проанализированы такие показатели, как время посадки и точность пространственного позиционирования БпЛА после посадки. В результате проведенного графического и статистического анализа выявлено значительное влияние на время посадки таких внешних факторов, как ветер, дождь и пониженная освещенность. Данное исследование имеет большую практическую значимость, так как улучшение производительности системы посадки БпЛА позволит повысить автономность, эффективность контроля и наблюдения за территориями, а также снизить риски возникновения аварийных ситуаций при посадке.

Данная статья организована следующим образом: в разделе 1 приводится анализ литературы по смежным исследованиям, а также постановка задачи. Раздел 2 содержит описание используемых для проведения экспериментов методах, алгоритмах и средах. Раздел 3 включает в себя описание и результаты экспериментов. В разделе 4 приведен анализ полученных экспериментальных данных и выводы. Наконец, в заключении подводятся итоги и обозначается вектор дальнейшего развития исследования.

1. Анализ существующих исследований в предметной области и постановка задачи

Авторы статьи [10] представили метод на основе компьютерного зрения для навигации БпЛА, позволяющий обнаруживать, следовать и приземляться на движущейся наземной робототехнической платформе с использованием маркеров ArUcO. Они провели тестирование в симуляционной и реальной среде, однако не предоставили результаты времени посадки и точности. Максимальная высота полета составила четыре метра. Планируется исследование масштабируемости метода для сценариев с несколькими воздушными и наземными роботами.

В работе [11] представлена автономная система для взлета и посадки БпЛА на движущейся платформе с использованием маркеров ArUcO. Авторы создали локальный планировщик движения, генерирующий бесколлизионные траектории, и разработали автономный конечный автомат для выполнения задач взлета, отслеживания и посадки. Исследования проводились в симуляторе Prometheus5 (ROS/Gazebo) и в реальных экспериментах. Максимальная высота в симуляции составила 4,5 метра, а в реальности — один метр. Скорость полета — 0,83 м/с. Однако информация о времени посадки и точности отсутствует. Также не предоставлены данные о скорости ветра и учете внешних условий. В дальнейших исследованиях авторы планируют сфокусироваться на задачах более высокой скорости и меньшей точности посадки на движущиеся цели, с улучшением точности приземления.

В статье [12] авторы представили полностью автономную систему на основе машинного зрения, которая решает задачу посадки беспилотного квадрокоптера на движущуюся платформу в условиях турбулентных ветров. Их метод интегрирует локализацию, планирование и управление, учитывая воздействие ветра. Авторы предлагают два режима оценки положения платформы: на большом расстоянии — с использованием расширенного фильтра Калмана и смоделированных GPS-измерений, и вблизи — с использованием фидуциальных меток AprilTag. Для управления в условиях ветровых порывов разработан вспомогательный контроллер, учитывающий граничный слой воздуха. Исследования включают имитационное моделирование с ветрами устойчивого типа и реальные эксперименты в помещении с искусственно созданными ветрами. Высота полета составляет 0,5 м, а скорость посадки занимает 5,7 секунд для статической платформы и 11,5 секунд для движущейся. Однако авторами не были представлены данные о точности посадки. В будущем они планируют расширить метод на более разнообразные условия, включая посадку на движущемся пикапе на открытом воздухе, с использованием визуально-инерционной одометрии для оценки состояния на борту.

В работе [13] представлен метод автономной посадки микро-БпЛА на движущуюся платформу с использованием модельно-предсказывающего управления. Этот метод требует широкополосных контуров управления с обратной связью для обеспечения безопасной посадки в условиях различных неопределенностей и ветровых помех. Архитектура системы включает в себя динамическое моделирование БпЛА, применение фильтра Калмана для оптимальной локализации мобильной платформы и разработку прогнозирующего управления моделями для наведения БпЛА. Исследования проводились с использованием симуляций как с ветром, так и без воздействия внешних условий. Высота полета составляла три метра, а скорость ветра варьировалась до 5 м/с. В симуляции без ветра точность посадки составила 0,21 метра, а время посадки составило 44,25 секунды. При наличии ветра точность уменьшилась до 0,37 метра, а время посадки увеличилось до 105,24 секунд. Однако авторы не предоставили информацию о работе системы в реальных условиях и не уделили внимание сбоям и восстановлению, что является предметом дальнейших исследований.

В работе [14] представлено решение для точной вертикальной посадки беспилотных летательных аппаратов с использованием маркеров ArUco. Авторы исследовали возможность посадки с высоты более 20 метров, чтобы компенсировать большие погрешности GPS. Для имитационного моде-

лирования была разработана собственная симуляционная платформа и использован пакет Ardusim. Высота полета составляла 20 метров, и средняя точность посадки составила 11 сантиметров. Время посадки заняло 162 секунды при скорости ветра 2,7 м/с. В дальнейших работах авторы планируют варьировать скорость снижения в зависимости от высоты БПЛА и использовать одновременное управление тангажом и креном, чтобы летательный аппарат мог следовать по диагональным линиям при посадке. Также авторы планируют оптимизировать алгоритм для обеспечения посадки в менее благоприятных погодных условиях.

В работе [15] представлена система автономной посадки мультироторного БпЛА на движущуюся платформу с использованием визуального следящего устройства. Эта система объединяет GPS-положение как БпЛА, так и платформы, чтобы обеспечить навигацию БпЛА, даже когда ориентир находится за пределами зоны видимости камеры или находится в ее слепой зоне. Высота полета БпЛА составляла четыре метра, а время посадки составило 35 секунд с точностью в 30 см. В работе не было представлено информации о влиянии внешних факторов, несмотря на проведение экспериментов в реальной среде. В будущем авторы намерены проводить совместные эксперименты с несколькими БпЛА и несколькими движущимися платформами для дальнейшего улучшения системы автономной посадки.

Проведя анализ предложенных решений, можно прийти к выводу, что во многих работах не указывается точность позиционирования при посадке и не учитывается влияние воздействия внешней среды, помимо ветра. В большинстве работ учитывается только одна скорость ветра и не принимается во внимание возможное изменение направления. Также почти все работы исследуют посадку на малых высотах (до 6 м). Помимо этого, предложенное нами решение во многом преуспевает либо в точности посадки, либо в скорости.

Исходя из вышесказанного, целесообразно осуществить исследование данной области, изменяя высоту и учитывая другие внешние параметры (затемнение, туман, различное направление и скорость ветра). Рассмотрим экспериментальную среду $\mathrm{S} = \{\tau, \varepsilon\}$, характеризующуюся временем посадки τ, выраженным как время в секундах, прошедшее с момента начала эксперимента до посадки БпЛА на платформу, и пространственной точностью позиционирования ε, выраженной как расстояние в метрах от центра БпЛА до центра платформы в момент завершения посадки. Эта среда характеризуется набором моделируемых внешних факторов $F = \{f_1, f_2, f_3, f_4\}$, а именно:

- ветер: $f_1(d, i)$, где $d \in \{1, 2, \ldots, 8\}$ — направление ветра (север, северо-запад, запад, юг и т. д.), $i \in \mathbb{R}$ — интенсивность ветра в м/с;
- освещенность: $f_2(p)$, где $p \in [0, 100]$ — степень затемнения;
- дождь: $f_3(r)$, где $r \in [1, 256]$ — интенсивность дождя;
- туман: $f_4(\theta)$, где $\theta \in [0, 1]$ — интенсивность тумана.

Время посадки и точность позиционирования БпЛА ε можно выразить как функционалы от этих факторов:

$$\tau = \varphi_\tau \left[f_1(d, i), f_2(p), f_3(r), f_4(\theta) \right],$$

$$\varepsilon = \varphi_\varepsilon \left[f_1(d, i), f_2(p), f_3(r), f_4(\theta) \right].$$

Тогда экспериментальная среда S определяется следующим образом:

$$S = \{\tau, \varepsilon\} = \{\varphi_\tau \left[f_1(d, i), f_2(p), f_3(r), f_4(\theta) \right], \ \varphi_\varepsilon \left[f_1(d, i), f_2(p), f_3(r), f_4(\theta) \right]\}.$$

Таким образом, задача исследования состоит в изучении функционалов $\varphi_\tau(F)$ и $\varphi_\varepsilon(F)$ в контексте экспериментальной среды S с использованием статистического анализа данных, полученных в ходе экспериментов. Это позволит определить, какие факторы или их комбинации оказывают влияние на время (τ) и точность (ε) посадки.

2. Материалы и методы

2.1. Используемые модели

Эксперименты проводятся в среде моделирования Gazebo. В качестве подвижной посадочной платформы используется модифицированная модель дифференциального колесного робота Clearpath Husky UGV (https://clearpathrobotics.com/husky-unmanned-ground-vehicle-robot), на которой размещен фрактальный ArUco маркер размером 1×1 метр с пятью уровнями вложенности. В работе используется модель квадрокоптера 3DR Iris (https://docs.px4.io/v1.12/en/simulation/gazebo_vehicles.html#quadrotor), предоставляемая PX4 SITL-симуляцией. Дополнительно на модель установлена нижняя RGB камера разрешением 640×480 и частотой 30 кадров/с.

2.2. Алгоритм контроля посадки

В работе используется алгоритм описанный в [9]. Наведение БпЛА на центр платформы осуществляется путем задания линейных скоростей по осям X, Y, Z контроллером посадки автопилоту. Контроллер посадки состоит из двух адаптивных по высоте пропорционально-интегрально-дифференцирующих (ПИД) регуляторов, генерирующих линейные скорости по осям X, У и логарифмически полиномиального регулятора (ЛП), генерирующего линейные скорости по оси Z в зависимости от полученных относительных координат $\Delta r = [\Delta x,\ \Delta y,\ \Delta z]^T$.

Алгоритм посадки функционирует следующим образом: когда маркер попадает в поле зрения камеры, ПИД-регуляторы горизонтального движения начинают генерировать линейные скорости, необходимые для преследования платформы. В это же время ЛП-регулятор движения по вертикали остается неактивным, т. е. БпЛА преследует платформу на изначальной высоте. Когда БпЛА находится в пределах 0,45 м от центра платформы, что считается оптимальным отклонением для снижения, активизируется ЛП-регулятор, происходит одновременное преследование платформы и снижение. Процесс посадки считается законченным, когда отклонение по оси Z становится менее 0,1 м и сохраняется условие для снижения. При более низкой высоте маркер становится нераспознаваемым. Далее происходит отключение моторов.

В случаях выпадения платформы из поля зрения камеры после детектирования по истечению трех секунд производится попытка взлета вверх со скоростью 5 м/с с целью увеличения поля зрения камеры. Если в течение двух секунд обнаружить платформу повторно не удалось, производится аварийное возвращение на исходную GPS позицию для повторного маневра.

2.3. Способы имитации погодных условий

Параметры окружающей среды могут оказывать значительное влияние на систему компьютерного зрения. Плохая освещенность, атмосферные явления, такие как осадки или туман, ухудшают качество изображения и уменьшают видимость, что затрудняет стабильное отслеживание платформы. Имитация погодных условий осуществляется путем добавления визуальных эффектов и манипуляцией над кадром с помощью инструментов OpenCV.

2.4. Моделирование влияния ветровых возмущений

Для моделирования влияния ветра $f_1 = (d, i)$ на БпЛА использовался плагин libgazebo wind (https://docs.px4.io/v1.12/en/simulation/gazebo.html#change-wind-speed), предоставляемый PX4 SITL симуляцией. Скорость ветра i передается в виде трехмерного вектора скоростей, состоящего из постоянной и переменной части, а направление ветра d передается в виде трехмерного вектора направления. Для моделирования случайного фактора добавляется отклонение скорости ветра на основе нормального распределения. Плагин добавляет влияние ветра в модель двигателя, рассматривая ветер как часть расчета сопротивления ротора.

а) На 80 %

б) На 50 %

в) На 20 %

Fig. 1. Brightness decrease

Рис. 1. Уменьшение яркости

2.5. Моделирование плохой освещенности

Для симуляции плохой освещенности $f_2(p)$ и имитации темного времени суток осуществляется перевод каждого кадра из цветовой модели RGB (Red, Green, Blue) в модель HSV (Hue, Saturation, Value), манипуляция над каналом Value, определяющим яркость изображения и принимающим значения от 1 до 100, и обратный перевод кадра в модель RGB. Предельное экспериментальное значение уменьшения яркости p составило 80 %. Дальнейшее уменьшение яркости приводит к некорректной работе алгоритма распознавания маркера на близких дистанциях, ориентировочно менее 5 м. Примеры кадров после манипуляции приведены на рисунке 1.

Данные примеры позволяют делать выводы о достоверности симуляции плохой освещенности предложенным методом.

2.6. Моделирование осадков

Для моделирования осадков $f_3(r)$ использовался метод из [16], который заключается в комбинировании исходного кадра и синтетических осадков. Сперва генерируется изображение N, равномерно распределенных случайных чисел в диапазоне от 0 до 256 той же размерности, что исходный кадр. Оно будет представлять изображение осадков, добавляемое к кадру. Для регулирования интенсивности осадков используется зануление сгенерированных чисел, удовлетворяющих следующему условию $N(x) < 256 - r$, где $N(x)$ — сгенерированное значение для пикселя x, r — коэффициент регулировки уровня шума, $r \in (1, 256)$. Далее необходимо осуществить свертки каждого пикселя сгенерированного изображения для получения эффекта дождя. Сперва производится свертка со следующим ядром:

$$k = \begin{bmatrix} 0.0 & 0.1 & 0.0 \\ 0.1 & 8.0 & 0.1 \\ 0.0 & 0.1 & 0.0 \end{bmatrix}.$$

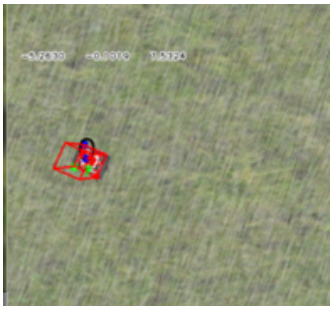
Затем генерируется новое ядро путем аффинного преобразования единичной матрицы E и сгенерированной матрицы вращения T, свойства которых определяются заданной длиной полосы и углом падения осадков. Наконец к ядру применяется размытие по Гауссу для задания ширины полос и производится повторная свертка.

Комбинирование исходного кадра с синтетическими осадками производится оператором линейного наложения (https://docs.opencv.org/3.4/d5/dc4/tutorial_adding_images.html):

$$g(x) = (1 - \alpha) f_0(x) + \alpha f_1(x),$$

где $g(x)$ — новый кадр, $f_0(x)$ — исходный кадр, $f_1(x)$ — синтетические осадки, α — коэффициент наложения.

а) При v = 90

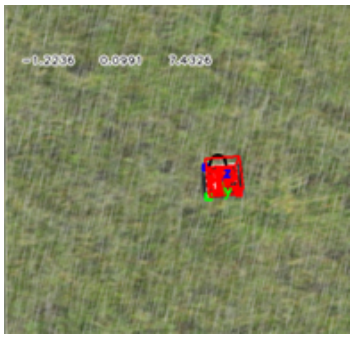


б) При v = 60

в) При v = 30

Fig. 2. Rain

Рис. 2. Дождь



а) При θ = 0,8

б) При θ = 0,5

в) При θ = 0,2

Fig. 3. Fog

Рис. 3. Туман

Примеры кадров с эффектом дождя приведены на рисунке 2.

Как видно из примеров, достигается правдоподобная симуляция осадков в кадре камеры.

2.7. Моделирование тумана

Имитация тумана $f_4(\theta)$ также осуществляется путем комбинирования исходного кадра и синтетического тумана. Генерация синтетического тумана формируется следующим образом:

1. Создается изображение, содержащие белые пиксели, той же размерности, что исходный кадр.
2. К белому изображению применяется размытие по Гауссу, размерность ядра для свертки изображения — (25, 25).
3. Сгенерированный туман комбинируется с исходным кадром с помощью оператора линейного наложения. Коэффициент наложения θ определяет плотность тумана.

Примеры кадров с синтетическим туманом приведены на рисунке 3.

Примеры показывают, что предложенный метод симуляции тумана справляется со своей задачей.

3. Эксперименты

3.1. Посадка с различной высоты без внешних факторов

В предшествующем исследовании было проведено тестирование посадки на различных высотах: 5 м, 10 м, 15 м, 20 м в количестве 40 попыток. Во всех экспериментах посадка производилась на платформу, движущуюся со скоростью 1 м/с по комплексной замкнутой траектории, состоящей из двух прямолинейных участков длиной 60 м, замкнутых с помощью двух дуг с радиусом кривизны 4 м. При каждой попытке проводилось измерение времени τ, необходимого на осуществление посадки, и точности позиционирования БпЛА ε, вычисляемой как расстояние между центром плат-

формы и центром масс БпЛА. В результате экспериментов высота 15 метров была максимальной, при которой достигалась 100 % успешность попыток, кроме того, дальнейшее увеличение высоты приводило к значительному увеличению затраченного на посадку времени (с 48,51 ± 4,526 с до 121,01 ± 14,188 с). Поэтому в текущем исследовании рассмотрена посадка с высоты 15 м.

3.2. Влияние ветра на процесс посадки

Было рассмотрено влияние ветра $f_1 = (d, i)$ со значениями i в диапазоне от 6 до 15 м/с с шагом в 3 м/с с каждой стороны горизонта d при посадке с высоты в 15 метров как стабильной высоты для посадки. Для каждого эксперимента была получена скорость посадки и точность позиционирования. Результаты представлены в таблице 1 (прочерком обозначена проваленная попытка).

Исходя из результатов, наибольшее влияние во всех тестах оказывают западные и восточные ветра, как ветра, действующие перпендикулярно направлению движения БпЛА, что приводит к необходимости дополнительно использовать энергетические ресурсы и время для нахождения в центральной области 45×45 см, так как БпЛА и платформа значительную часть пути двигаются на юг. По этой же причине южный ветер также затрудняет и замедляет процесс посадки, как ветер, движущийся противоположно направлению БпЛА. Практически во всех опытах время посадки превысило максимальное значение, полученное из предыдущих опытов. Среднее значение времени посадки с ветром превышает значение без ветра на 33,13 с (увеличилось на 68 %). Также снизилась точность. Среднее значение точности посадки с ветром превышает значение без ветра на 0,06 м (увеличилось на 23 %).

Table 1. The influence of wind disturbances on landing time and accuracy

Таблица 1. Влияние ветровых возмущений на время и точность посадки

i (м/с)	d	τ (с)	ε (м)
6	Северный	44,34	0,06
	Южный	65,95	0,25
	Западный	54,10	0,31
	Восточный	71,36	0,11
	Северо-западный	63,05	0,35
	Юго-восточный	80,08	0,37
9	Северный	46,32	0,31
	Южный	56,90	0,40
	Западный	61,70	0,32
	Восточный	79,74	0,37
	Северо-западный	55,65	0,23
	Юго-восточный	63,34	0,33
12	Северный	49,40	0,35
	Южный	71,79	0,17
	Западный	64,34	0,39
	Восточный	84,70	0,42
	Северо-западный	76,22	0,26
	Юго-восточный	141,24	0,15
15	Северный	—	—
	Южный	69,49	0,21
	Западный	—	—
	Восточный	127,80	0,63
	Северо-западный	—	—
	Юго-восточный	286,98	0,49

3.3. Влияние плохой освещенности на процесс посадки

Были проведены эксперименты по посадке с высоты 15 м с моделированием плохой освещенности $f_2(p)$ при значениях p 20 %, 50 % и 80 %. Дальнейшее уменьшение яркости приводит к неработоспособности алгоритма распознавания маркера на близких дистанциях, ориентировочно менее 5 м. Для каждого значения процента уменьшения яркости было проведено по 10 попыток посадки, все попытки оказались удачными. Для каждого эксперимента подсчитывалось время для посадки и точность позиционирования.

В таблице 2 представлены средние значения точности позиционирования ε и времени посадки τ для каждого значения процента уменьшения освещенности.

Table 2. The effect of poor lighting on landing time and accuracy

Таблица 2. Влияние плохой освещенности на время и точность посадки

p (%)	τ (с)	ε (м)
20	45,11 ± 4,250	0,18 ± 0,078
50	41,05 ± 1,771	0,20 ± 0,077
80	44,39 ± 4,620	0,27 ± 0,076

Таким образом, понижение уровня освещенности оказывает слабое влияние на распознавание и процесс посадки, однако очень слабая освещенность, например в ночное время суток, приводит к тому, что маркер перестает распознаваться на малых расстояниях (ориентировочно пять метров и ниже), но при этом сохраняется способность распознавания на большой высоте, в результате чего снижается точность посадки.

3.4. Влияние дождя на процесс посадки

Были проведены эксперименты по посадке с высоты 15 м с моделированием дождя $f_3(r)$ при значениях r 30, 60 и 90. Угол падения осадков в моделировании варьировался в диапазоне $(-30, 30)$. Для каждого значения r было проведено по 10 измерений точности позиционирования и времени посадки, все попытки оказались удачными. В таблице 3 представлены средние значения точности позиционирования ε и времени посадки τ для каждого значения интенсивности осадков.

Table 3. The influence of precipitation on landing time and accuracy

Таблица 3. Влияние осадков на время и точность посадки

r	τ (с)	ε (м)
30	46,89 ± 6,554	0,20 ± 0,080
60	52,46 ± 3,336	0,22 ± 0,093
90	60,76 ± 7,934	0,21 ± 0,049

При влиянии дождя наблюдается увеличение времени, затраченного на посадку при увеличении степени интенсивности осадков, однако дождь не влияет на точность окончательного позиционирования БпЛА относительно маркера.

3.5. Влияние тумана на процесс посадки

Были проведены эксперименты по посадке с высоты 15 м с моделированием тумана $f_4(\theta)$ при значениях θ 0,2, 0,5 и 0,8. Для каждого значения интенсивности тумана было проведено по 10 измерений точности позиционирования и времени посадки, все попытки оказались удачными. В таблице 4 представлены средние значения точности позиционирования ε и времени посадки θ для каждого значения интенсивности тумана.

Исходя из сравнения средних значений времени и точности посадки, влияние тумана на процесс распознавания маркера и посадки на платформу незначительно.

Table 4. Effect of fog on landing time and accuracy

Таблица 4. Влияние тумана на время и точность посадки

θ	τ (с)	ε (м)
0,2	46,01 ± 3,274	0,21 ± 0,080
0,5	46,06 ± 4,311	0,22 ± 0,086
0,8	48,56 ± 3,787	0,21 ± 0,082

3.6. Исследование комплексного влияния погодных условий на процесс посадки

Для исследования комплексного влияния нескольких внешних факторов, было проведено моделирование посадки БпЛА с высоты 15 м со следующими дополнительными внешними факторами:

- уменьшение освещенности на 80 % + комплексный ветер 12 м/с;
- осадки с интенсивностью v = 1 + комплексный ветер 12 м/с;
- уменьшение освещенности на 50 % + туман с интенсивностью a = 0,5;
- уменьшение освещенности на 50 % + осадки с интенсивностью v = 1:
- уменьшение освещенности на 50 % + осадки с интенсивностью v = 1 + комплексный ветер 9 м/с.

Значение интенсивности осадков при исследовании комплексного влияния погодных условий было сведено к минимальному значению (v = 1), так как предварительное моделирования показало, что прочие значения интенсивности осадков с совокупности с другими явлениями приводило к невозможности совершения посадки. Значение интенсивности v = 1 характеризует очень слабые осадки, моросящий дождь.

В каждом случае было произведено по 10 измерений точности и времени посадки, все попытки оказались удачными. В таблице 5 представлены средние значения точности позиционирования ε и времени посадки θ для каждого случая.

Table 5. The influence of complex conditions on landing time and accuracy

Таблица 5. Влияние комплексных условий на время и точность посадки

Внешние факторы F	τ (с)	ε (м)
Уменьшение освещенности на 80 % + комплексный ветер 12 м/с	105,50 ± 18,933	0,29 ± 0,087
Осадки с интенсивностью v = 1 + комплексный ветер 12 м/с	188,72 ± 34,100	0,24 ± 0,080
Уменьшение освещенности на 50 % + туман с интенсивностью a = 0,5	44,32 ± 4,467	0,20 ± 0,086
Уменьшение освещенности на 50 % + осадки с интенсивностью v = 1	56,61 ± 3,529	0,25 ± 0,077
Уменьшение освещенности на 50 % + осадки с интенсивностью v = 1 + комплексный ветер 9 м/с	112,88 ± 7,976	0,24 ± 0,082

Данные результаты показывают, что наибольшее влияние на процесс посадки как с точки зрения времени, так и с точки зрения точности позиционирования, оказывает введение ветра, поскольку ветровые возмущения напрямую влияют на управление БпЛА, а не на распознавание маркера как другие внешние факторы. Наихудшие результаты демонстрируются при комбинации осадков, которые сами по себе также продемонстрировали влияние на процесс посадки, и ветра.

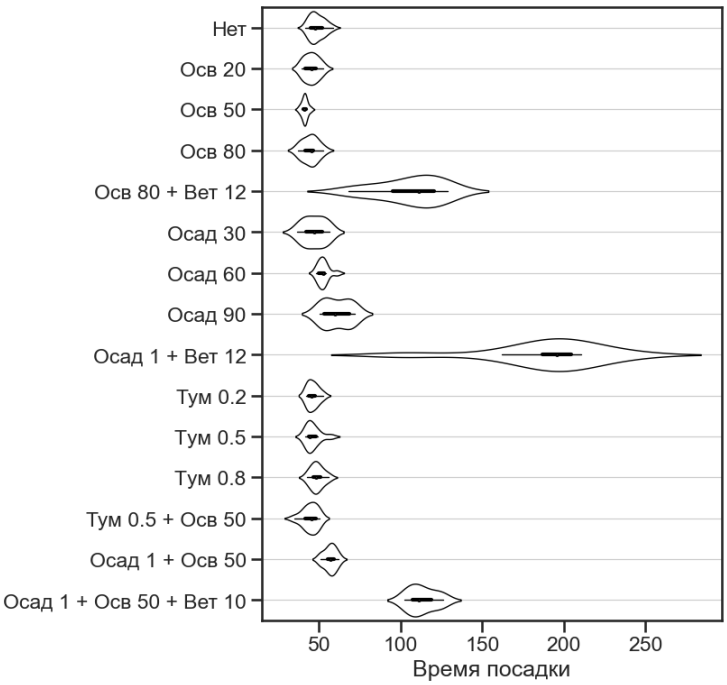


Fig. 4. Violin plots of landing time data in experiments with various external factors

Рис. 4. Скрипичные графики данных времени посадки в экспериментах с различными внешними факторами

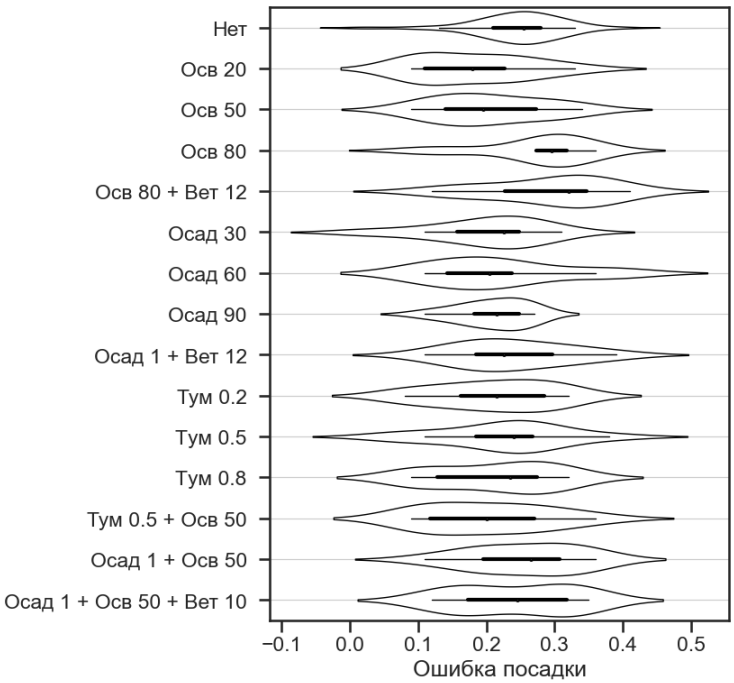


Fig. 5. Violin plots of fit accuracy data in experiments with various external factors

Рис. 5. Скрипичные графики данных точности посадки в экспериментах с различными внешними факторами

	Нет	Осв 20	Осв 50	Осв 80	Осв 80 + Вет 12	Осад 30	Осад 60	Осад 90	Осад 1 + Вет 12	Тум 0.2	Тум 0.5	Тум 0.8	Тум 0.5 + Осв 50	Осад 1 + Осв 50	Осад 1 + Осв 50 + Вет 10
Нет	-	0.06	**8.09E-06**	**0.032**	**1.31E-06**	0.489	**0.013**	**8.97E-05**	**1.31E-06**	0.121	0.083	0.846	**0.032**	**8.11E-05**	**1.31E-06**
Осв 20		-	**0.045**	0.762	**1.83E-04**	0.597	**0.002**	**4.40E-04**	**1.83E-04**	0.678	0.791	0.121	0.94	**3.30E-04**	**1.83E-04**
Осв 50			-	0.089	**1.83E-04**	**0.045**	**1.83E-04**	**1.83E-04**	**1.83E-04**	**0.002**	**0.002**	**3.30E-04**	**0.049**	**1.83E-04**	**1.83E-04**
Осв 80				-	**1.83E-04**	0.385	**0.001**	**4.40E-04**	**1.83E-04**	0.364	0.734	0.064	1.0	**3.30E-04**	**1.83E-04**
Осв 80 + Вет 12					-	**1.83E-04**	**1.83E-04**	**4.40E-04**	**0.001**	**1.83E-04**	**1.82E-04**	**1.83E-04**	**1.83E-04**	**1.83E-04**	0.623
Осад 30						-	0.076	**0.004**	**1.83E-04**	0.91	0.91	0.521	0.427	**0.005**	**1.83E-04**
Осад 60							-	**0.021**	**1.83E-04**	**0.001**	**0.004**	**0.031**	**0.001**	**0.021**	**1.83E-04**
Осад 90								-	**1.83E-04**	**4.40E-04**	**0.001**	**0.002**	**1.83E-04**	0.326	**1.83E-04**
Осад 1 + Вет 12									-	**1.83E-04**	**1.82E-04**	**1.83E-04**	**1.83E-04**	**1.83E-04**	**0.002**
Тум 0.2										-	0.734	0.14	0.521	**3.30E-04**	**1.83E-04**
Тум 0.5											-	0.14	0.734	**0.001**	**1.82E-04**
Тум 0.8												-	0.064	**0.001**	**1.83E-04**
Тум 0.5 + Осв 50													-	**1.83E-04**	**1.83E-04**
Осад 1 + Осв 50														-	**1.83E-04**
Осад 1 + Осв 50 + Вет 10															-

Fig. 6. Results of pairwise Mann-Whitney test for landing time samples

Рис. 6. Результаты попарного теста по критерию Манна—Уитни для выборок времени посадки

4. Анализ полученных результатов

Для резюмирования результатов экспериментов и оценки влияния различных внешних условий проведем графический и статистический анализ полученных экспериментальных данных при посадке с высоты 15 м без внешних факторов и с включением всех различных внешних факторов. На рисунках 4, 5 представлены скрипичные графики времени посадки и точности позиционирования при посадке во всех проведенных экспериментах.

На графиках наблюдается значительное влияние ветра и осадков на время посадки, чего нельзя сказать о точности посадки. Также исходя из графического анализа можно предположить, что не все данные следуют нормальному распределению.

Для дальнейшего анализа результатов принято решение обратиться к статистическим методам для выяснения наличия статистически значимой разницы в полученных результатах. Для начала, была проведена первоначальная оценка данных по критерию Шапиро—Уилка [17]. Она показала, что данные не следуют нормальному распределению. Это означает, что применение традиционных статистических методов, таких как критерий Стьюдента, становится неприемлемым. Вместо этого, был использован критерий Манна—Уитни [18], который позволяет проводить статистический анализ без предположений о распределении данных и не чувствителен к размеру выборки. За нулевую гипотезу было взято отсутствие значимых различий между выборками, данная гипотеза отвергалась при p-значении проведенного теста меньше либо равного 0,05. Тест производился попарно между двумя выборками каждый с каждым. Результаты теста приведены на рисунках 6, 7.

Исходя из полученных результатов, можно заключить следующее:

1. На текущем этапе визуальная оценка и полученные результаты моделирования позволяют делать выводы об адекватности используемых моделей погодных условий. БпЛА ожидаемо реагирует на моделируемые внешние факторы, что описано в таблицах 1–5 и наблюдается на рисунках 4, 5. Оценка соответствия моделей работы БпЛА при осадках и тумане в реальной

	Нет	Осв 20	Осв 50	Осв 80	Осв 80 + Вет 12	Осад 30	Осад 60	Осад 90	Осад 1 + Вет 12	Тум 0.2	Тум 0.5	Тум 0.8	Тум 0.5 + Осв 50	Осад 1 + Осв 50	Осад 1 + Осв 50 + Вет 10
Нет	-	**0.03**	0.129	0.084	0.068	0.074	0.126	0.052	0.551	0.291	0.481	0.362	0.142	0.78	1.0
Осв 20		-	0.544	**0.028**	**0.017**	0.495	0.425	0.363	0.198	0.57	0.225	0.544	0.622	0.082	0.14
Осв 50			-	0.103	0.058	0.91	0.85	0.82	0.427	0.97	0.65	0.94	0.909	0.161	0.405
Осв 80				-	0.427	**0.028**	0.198	**0.019**	0.325	0.081	0.089	0.075	0.082	0.677	0.623
Осв 80 + Вет 12					-	0.037	0.15	**0.034**	0.173	**0.049**	0.112	**0.041**	0.058	0.363	0.198
Осад 30						-	0.94	0.88	0.449	0.82	0.448	0.677	0.94	0.212	0.344
Осад 60							-	0.791	0.57	1.0	0.57	1.0	0.82	0.383	0.623
Осад 90								-	0.495	0.82	0.519	0.733	0.85	0.185	0.406
Осад 1 + Вет 12									-	0.596	0.91	0.622	0.384	0.705	0.91
Тум 0.2										-	0.85	0.94	0.85	0.226	0.405
Тум 0.5											-	0.97	0.649	0.449	0.623
Тум 0.8												-	0.879	0.272	0.344
Тум 0.5 + Осв 50													-	0.197	0.325
Осад 1 + Осв 50														-	0.733
Осад 1 + Осв 50 + Вет 10															-

Fig. 7. Results of pairwise Mann-Whitney test for landing time samples

Рис. 7. Результаты попарного теста по критерию Манна—Уитни для выборок времени посадки

среде являются частью дальнейших исследований, где будет проведено сравнение моделирование и реальных экспериментов.

2. При сравнении выборки без внешних условий с каждой другой выборкой, наблюдается статистически значимое влияние на время посадки таких внешних факторов, как освещенность (при снижении интенсивности от 50), осадки (при интенсивности от 60) и ветер. В результате введения ветра ухудшается способность распознавания маркера в темное время суток в результате чего БпЛА требуется больше времени для сближения с платформой.
3. Туман не оказывает значимого влияния на время посадки сам по себе, однако в комплексе со снижением освещенности, начинает влиять на время посадки.
4. Введение осадков приводит к статистически значимым изменениям времени посадки по сравнению со всеми прочими экспериментами, что также указывает на высокую степень влияния данного внешнего фактора. С ростом интенсивности осадков уменьшается производительность детектирования маркера как на больших расстояниях, так и на малых, с уменьшением расстояния способность распознавать маркер ослабевает, более того чем интенсивнее осадки, тем сложнее сблизиться с платформой из-за периодических проблем с распознаванием.
5. Любые ветровые возмущения в совокупности с дестабилизацией распознавания препятствуют совершению посадки. Поэтому маневр возможен только при очень слабых осадках. Таким образом возможность посадки на маркер при дожде с ветром ставится под сомнение. Что является критической ситуацией для БпЛА и должно учитываться при его функционировании. Введение ветра в комбинацию с любым внешним фактором приводит к значительному увеличению времени посадки.
6. Исходя из данных на рисунке 7, можно сделать вывод, что рассмотренные внешние условия не оказывают значительного влияния на пространственную точность позиционирования БпЛА при посадке, кроме комбинации внешних факторов освещенности и ветра, где продемонстрированы статистически значимые различия с множеством других выборок.

Заключение

Целью данной работы было исследование влияния внешних условий на процесс автоматизированной посадки БпЛА-квадрокоптера на подвижную платформу с использованием компьютерного зрения. Для этого были проведены эксперименты с различными комбинациями внешних факторов, таких как освещенность, туман, осадки и ветер. Были измерены время посадки и пространственная точность позиционирования по завершению посадки. Был проведен статистический анализ полученных данных с использованием критерия Манна—Уитни. Основные результаты исследования можно сформулировать следующим образом:

1. Внешние условия сильно влияют на время посадки БпЛА, но не на точность посадки.
2. Освещенность, осадки и ветер — самые важные факторы, которые нужно учитывать при посадке.
3. Туман усиливает эффект от других внешних условий, а дождь и ветер вместе могут сделать посадку невозможной.

Таким образом, данное исследование показало, что внешние условия являются важным фактором, который нужно учитывать при разработке системы автоматической посадки БпЛА на подвижную платформу. Для повышения эффективности и безопасности такой системы необходимо разработать алгоритмы, которые будут адаптироваться к различным условиям и корректировать параметры посадки в зависимости от них. Также необходимо провести дополнительные эксперименты с другими типами маркеров, которые могут быть более устойчивы к внешним факторам. Кроме того, необходимо исследовать возможность использования других источников информации для определения места посадки, например, инерциальных датчиков или радиосигналов.

References

[1] A. Motienko, I. Vatamaniuk, and A. Saveliev, “Development of technical appearance of human-machine interface for group control of unmanned robots when performing agricultural tasks”, *Robotics and Technical Cybernetics*, vol. 16, no. 4, pp. 299–311, 2021.

[2] A. Ronzhin and A. Saveliev, “Artificial intelligence systems for solving problems of agro-industrial complex digitalization and robotization”, *Agricultural Machinery and Technologies*, vol. 16, no. 2, pp. 22–29, 2022.

[3] A. Saveliev, V. Lebedeva, I. Lebedev, and M. Uzdiaev, “An approach to the automatic construction of a road accident scheme using UAV and deep learning methods”, *Sensors*, vol. 22, no. 12, p. 4728, 2022.

[4] A. Nosov *et al.*, “Case study of transporting blood components using an unmanned aerial vehicle”, *Disaster Medicine*, vol. 3, pp. 65–69, 2022.

[5] A. Saveliev, E. Aksamentov, and E. Karasev, “Automated terrain mapping based on mask R-CNN neural network”, *International Journal of Intelligent Unmanned Systems*, vol. 10, no. 2/3, pp. 267–277, 2022.

[6] V. Lebedeva, K. Kamynin, I. Lebedev, L. Kuznetsov, and A. Saveliev, “Method for distributed mapping of terrain by a heterogeneous group of robots based on google cartographer”, in *Proceedings of the Computational Methods in Systems and Software*, vol. 596, 2022, pp. 584–597.

[7] A. Saveliev and I. Lebedev, “Method of autonomous survey of power lines using a multi-rotor UAV”, in *Frontiers in Robotics and Electromechanics*, Springer, 2023, pp. 359–376.

[8] Y. Vasunina, D. Anikin, and A. Saveliev, “Algorithm of UAV trajectory creation for data collecting from seismological sensors”, in *International Russian Automation Conference (RusAutoCon)*, 2023, pp. 747–752.

[9] D. Anikin, A. Ryabinov, A. Saveliev, and A. Semenov, "Autonomous landing algorithm for UAV on a mobile robotic platform with a fractal marker", in *Interactive Collaborative Robotics (ICR)*, 2023, pp. 357–368.

[10] J. Morales, I. Castelo, R. Serra, P. U. Lima, and M. Basiri, "Vision-based autonomous following of a moving platform and landing for an unmanned aerial vehicle", in *Sensors*, vol. 23, 2023, p. 829.

[11] P. Wang, C. Wang, J. Wang, and M. Meng, "Quadrotor autonomous landing on moving platform", in *Procedia Computer Science*, vol. 209, 2022, pp. 40–49.

[12] A. Paris, B. Lopez, and J. How, "Dynamic landing of an autonomous quadrotor on a moving platform in turbulent wind conditions", in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020, pp. 9577–9583.

[13] Y. Feng, C. Zhang, S. Baek, S. Rawashdeh, and A. Mohammadi, "Autonomous landing of a UAV on a moving platform using model predictive control", in *Drones*, vol. 2, 2018, p. 34.

[14] J. Wubben *et al.*, "A vision-based system for autonomous vertical landing of unmanned aerial vehicles", in *23rd International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*, 2019, pp. 1–7.

[15] Z. Zhao *et al.*, "Vision-based autonomous landing control of a multi-rotor aerial vehicle on a moving platform with experimental validations", in *IFAC-PapersOnLine*, vol. 55, 2022, pp. 1–6.

[16] T. Matsui and M. Ikehara, "Gan-based rain noise removal from single-image considering rain composite models", in *28th European Signal Processing Conference (EUSIPCO)*, 2021, pp. 665–669.

[17] S. Shapiro and M. Wilk, "An analysis of variance test for normality", in *Biometrika*, vol. 52, 1965, pp. 591–611.

[18] H. Mann and D. Whitney, "On a test of whether one of two random variables is stochastically larger than the other", in *The annals of mathematical statistics.*, vol. 18, 1947, pp. 50–60.

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 4, 2023
journal homepage: www.mais-journal.ru

ARTIFICIAL INTELLIGENCE

Extracting named entities from Russian-language documents with different expressiveness of structure

M. D. Averina[1], O. A. Levanova[1]

DOI: 10.18255/1818-1015-2023-4-382-393

[1]P.G. Demidov Yaroslavl State University, 14 Sovetskaya str., Yaroslavl 150003, Russia.

MSC2020: 68T50
Research article
Full text in Russian

Received October 13, 2023
After revision November 10, 2023
Accepted November 15, 2023

This work is devoted to solving the problem of recognizing named entities for Russian-language texts based on the CRF model. Two sets of data were considered: documents on refinancing with a good document structure, semi-structured texts of court records. The model was tested under various sets of text features and CRF parameters (optimization algorithms). In average for all entities, the best F-measure value for structured documents was 0.99, and for semi-structured ones 0.86.

Keywords: named entity extraction; CRF

INFORMATION ABOUT THE AUTHORS

Maria D. Averina
corresponding author
orcid.org/0009-0005-3111-1488. E-mail: maverina518@gmail.com
graduate student.

Olga A. Levanova
orcid.org/0000-0001-8078-4447. E-mail: olaydy@gmail.com
PhD, Associate professor.

For citation: M. D. Averina and O. A. Levanova, “Extracting named entities from Russian-language documents with different expressiveness of structure”, *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 382-393, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 4, 2023
сайт журнала: www.mais-journal.ru

Извлечение именованных сущностей из русскоязычных документов с различной выраженностью структуры

М. Д. Аверина[1], О. А. Леванова[1]

DOI: 10.18255/1818-1015-2023-4-382-393

[1]Ярославский государственный университет им. П.Г. Демидова, ул. Советская, д. 14, г. Ярославль, 150003 Россия.

УДК 004.912
Научная статья
Полный текст на русском языке

Получена 13 октября 2023 г.
После доработки 10 ноября 2023 г.
Принята к публикации 15 ноября 2023 г.

Данная работа посвящена решению задачи распознавания именованных сущностей для русскоязычных текстов на основе модели CRF. Рассмотрены два набора данных: документы о рефинансировании с хорошей структурой документа, слабоструктурированные тексты судебных протоколов. Было проведено тестирование модели при различных наборах текстовых признаков и параметрах CRF (алгоритмов оптимизации). В среднем по всем сущностям лучшее значение F-меры для структурированных документов составило 0.99, а для слабоструктурированных 0.86.

Ключевые слова: извлечение именованных сущностей; CRF

ИНФОРМАЦИЯ ОБ АВТОРАХ

Мария Дмитриевна Аверина автор для корреспонденции	orcid.org/0009-0005-3111-1488. E-mail: maverina518@gmail.com аспирант.
Ольга Александровна Леванова	orcid.org/0000-0001-8078-4447. E-mail: olaydy@gmail.com канд. физ.-мат. наук, доцент.

Для цитирования: M. D. Averina and O. A. Levanova, “Extracting named entities from Russian-language documents with different expressiveness of structure”, *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 382-393, 2023.

Введение

В современную эпоху цифровизации всё большую популярность обретают технологии электронного документооборота. В этой связи усиливается интерес к автоматизации обработки электронных документов путём извлечения из них информации. Например, бывает полезно выделить все даты и автоматически отсортировать документы. Или же, после сканирования банковских документов удобно автоматически определить не только ФИО клиента, но и имя сотрудника, паспортные данные клиента или сумму кредита, такая информация может существенно упростить процесс обработки электронных документов.

Задача распознавания именованных сущностей (NER) состоит в автоматическом выявлении текстовых фрагментов, которые несут определенную смысловую нагрузку. Например, в классической задаче выделены следующие классы: человек (PER), местоположение (LOC), организация (ORG) и другие (MISC) [1]. Для других предметных областей сущности могут быть менее обобщенными и более сложными. Так, для судебных протоколов может быть полезно выделить не просто все фамилии в документе, а более конкретно — ФИО истца и судьи. Примером сложной сущности может быть решение суда, которое «размазано» по тексту (сущности соответствует не сплошной фрагмент текста).

Данная статья рассматривает решение задачи распознавания именованных сущностей для русскоязычных судебных и банковских документов. Тексты в таких документах отличаются от текстов в других предметных областях и не похожи между собой. Стоит отметить, что банковские документы имеют стандартную структуру, а судебные протоколы написаны в более свободном стиле и плохо структурированы.

Задача извлечения именованных сущностей особенно актуальна для русского языка, поскольку почти все существующие системы и библиотеки хорошо работают с английскими текстами, а для других языков результаты [2] значительно хуже (или же их вообще нет). Статей посвященных задаче NER для не классических сущностей крайне мало. Наибольший интерес представляет статья языке для новостных текстов на казахском [3], в которой ищется 25 сущностей.

Классическим подходом для решения поставленной задачи является модель CRF [4]. Однако ее качество очень сильно зависит от предварительной обработки текста и внутренних параметров алгоритма. В статье исследованы различные текстовые признаки и параметры модели CRF.

1. Наборы данных

Для данного исследования были использованы два набора данных: судебные протоколы и договоры о консолидации и рефинансировании задолженности. Рассмотрим каждый тип документа подробнее.

1.1. Судебные протоколы

Из-за специфики задачи в документах были выделены сущности, которые характерны для юридической области. Так, например, из текста необходимо было выделить не просто ФИО всех людей, но также дифференцировать их: ответчик это, истец или судья. Данные документы были взяты из открытой базы судебной статистики (http://www.cdep.ru), которая содержит анонимизированные судебные решения. Такой выбор был сделан из-за большого количества имеющихся документов и характера текстов, а именно: тексты содержат множество различных имён, дат, номиналов и сумм. Стоит отметить, что данные документы практически не структурированы, что усложняет решение поставленной задачи.

№ <*doc num, 2-1606/2018*>

РЕШЕНИЕ
Именем Российской федерации
<*date court, 02 ноября 2018 года*> <*court, Кировский районный суд г.Томска*> в составе:
председательствующего судьи <*judge, Алиткиной Т.А.*>,
при секретаре Бондаревой Е.Е.,
с участием представителя процессуального истца старшего прокурора Кировского района г.Томска Морарь И.В., материального истца <*plaintiff, Полищука Э.Г.*>, представителя ответчика <*defendant, Семенова С.М*>., действующего на основании доверенности от 17.05.2017 (срок действия доверенности три года),

...

собственному желанию, взыскании задолженности по заработной плате, компенсации, предусмотренной ст.236 ТК РФ, компенсации за задержку выплаты заработной платы, взыскании денежной компенсации морального вреда <*court decision, удовлетворить частично*>.

Fig. 1. Example of a court record. Beginning of text.

Рис. 1. Пример судебного протокола. Начало текста.

<*payment fine, задолженность по заработной плате*> за период с 26.12.2017 по 22.01.2018 в размере <*payment amount, 20 520,00 руб.*>; <*payment fine,денежную компенсацию, предусмотренную ст.236 ТК РФ*>, в размере <*payment amount,796,20 руб.*>; <*payment fine,утраченный заработок за время вынужденного прогула*> в размере <*payment amount, 91 198,80 руб.*>, денежную <*payment fine,компенсацию морального вреда* >в размере <*payment amount, 3 000 руб*>.

...

Решение может быть обжаловано в Томский областной суд путем подачи апелляционной жалобы через Кировский районный суд г.Томска <*appeal time, в течение 1 месяца со дня изготовления решения в мотивированном виде*>.

Судья: (подпись) <*judge, Т.А.Алиткина*>

Fig. 2. Example of a court record. End of text.

Рис. 2. Пример судебного протокола. Конец текста.

На основе данной базы была сформирована выборка из 344 файлов (документы состоят из 4–7 страниц), и размечена с помощью инструмента BRAT (онлайн-инструмент для разметки письменных текстов https://brat.nlplab.org). На рисунках 1 и 2 показан пример разметки документа. Все сущности условно можно разделить на группы, характеризующие сложность их распознавания. В группу простых сущностей входят номер документа (doc num), решение суда, судья и другие (рис. 1). Встречаются и сложные сущности, которые содержат более двух слов, такие как время обжалования, истец, штраф, сумма платежа, ответчик и суд.

Среднее количество слов для каждой сущности и процент документов, содержащих соответствующие объекты представлены в таблице 1. Некоторые сущности встречаются в меньшем числе документов, что может сказаться на качестве распознавания.

Более того, некоторые сущности вариативны и зависят от содержания документа. Например, ответчиком может быть физическое лицо, организация или представитель ответчика. Уплата штрафа

Table 1. Entities analysis for court records

Таблица 1. Анализ сущностей судебных протоколов

Сущность	Размер сущности	Процент встречаемости в документах
истец	2.4	98
статья или тип штрафа	2.4	77
сумма выплаты	8.2	70
судья	1.9	98
номер документа	0.9	78
ответчик или представитель	6.0	98
дата суда	2.6	93
суд	4.2	98
решение суда	1.9	94
срок обжалования	9.6	95

Table 2. Entities analysis of refinancing agreements

Таблица 2. Анализ сущностей для договоров о рефинансировании

Сущность	Размер сущности	Процент встречаемости в документах
клиент 1	3.0	100.0
номер паспорта	1.0	63.4
номер документа	3.5	99.0
число выплат	1.0	90.0
клиент 2	6.0	100.0
сумма прощения	2.0	100.0
дата документа	1.5	97.0
день рождение	1.6	77.0
месячная выплата	2.0	99.0
номер кредитного договора	1.0	100.0
имя банка	3.0	100.0
дата кредитного договора	1.0	98.5
итоговая сумма	4.0	100.0
дата первого платежа	1.0	93.0
день оплаты	4.0	96.0
адрес клиента	7.0	63.0

в разных документах может иметь различные значения, такие как статья закона, платеж или компенсация. Сложные сущности могут быть прерывистыми, например штраф и сумма платежа (рис. 2), и поэтому их труднее распознать.

1.2. Договоры о консолидации и рефинансировании задолженности

Второй набор данных был предоставлен одной из коммерческих организаций, заинтересованной в автоматизации процессов документооборота. Данные договоры можно отнести к полуструктурированному типу, поскольку часть сущностей располагается в таблицах, а другие в сплошном

тексте. Стоит отметить, что структура документа у различных организаций отличается, поэтому результаты выделения именованных сущностей могут отличаться.

Экспертами было размечено 200 договоров о рефинансировании при помощи инструмента BRAT. В выборке преобладают документы состоящие из одной страницы, но также встречаются и двухстраничные. Пример такого документа представлен на рисунке 3 (документ изменен в целях конфиденциальности информации). Для поставленной задачи требовалось выделить 16 сущностей, в основном простых. Из таблицы 2 видно, что практически все сущности содержатся в каждом документе, кроме номера паспорта и адреса клиента.

2. Извлечение признаков

Первым этапом работы с текстом является его предобработка: токенизация, удаление лишних слов (например, стоп-слов) или символов. Затем необходимо извлечь признаки из текста. Существует несколько подходов: основанные на регулярных выражениях, морфологических признаках, синтаксическом и семантическом анализе. Наиболее очевидным и простым решением является извлечение признаков при помощи регулярных выражений. Например, можно извлечь информацию о ближайших знаках препинания или регистре букв: номер документа «№ 11255588» делится на «№» и «11255588», где № идентифицируется как специальный символ. В работе были использованы следующие специальные символы: @, #, №, \, %, $, |.

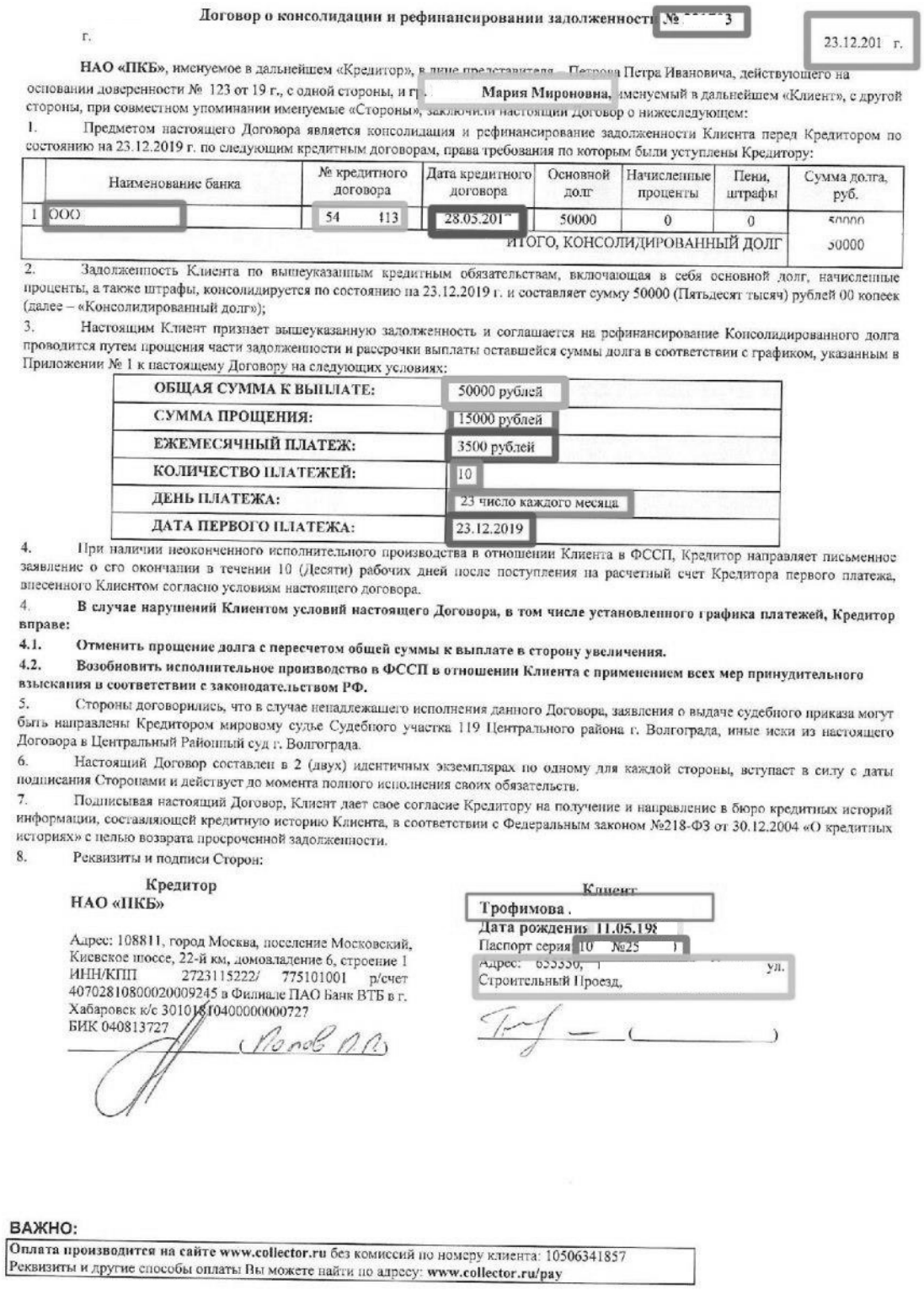
Ниже представлен список признаков, основанных на регулярных выражениях:

- первая буква прописная;
- первая буква маленькая;
- все буквы маленькие;
- все буквы заглавные;
- наличие @ внутри слова;
- наличие запятой и (или) точки в конце (начале) слова;
- есть ли в слове цифры.

Все слова в тексте имеют различные падежи и склонения, что в свою очередь может усложнять работу модели. Одним из способов решения данной проблемы является нормализация — приведение слова в начальную форму при помощи лемматизации или стемминга. При этом для каждого слова сохраняются признаки такие как: число и род и т. д. Стоит отметить, что часть речи также является морфологическим признаком, и в случае слов-спецсимволов каждому символу присваивается уникальное значение части речи.

Так же в качестве признака можно использовать само «слово», но данный признак не является информативным из-за большой вариативности. Например, если в используемых документах часто фигурирует фамилия судьи и в выборке много дел с одним судьей, то на других данных (с другими судьями) фамилия судьи находиться не будет.

Следующим по популярности подходом к вычислению признаков является векторизация слов — представление слов в виде вектора чисел. Для данного подхода были выбраны алгоритмы Word2Vec [5] и FastText [6], основанные на контекстной близости слов. Алгоритм Word2Vec работает с большими текстовыми данными и по определенным правилам, присваивает каждому слову уникальный набор чисел — семантический вектор. Спустя время были представлены улучшенные модификации данного алгоритма, одной из них является FastText. FastText исправляет недостаток Word2Vec, заключающийся в невозможности представления слова в виде вектора, если его не было в обучающем наборе.



Договор о консолидации и рефинансировании задолженности № 3

г. 23.12.201 г.

НАО «ПКБ», именуемое в дальнейшем «Кредитор», в лице представителя Петрова Петра Ивановича, действующего на основании доверенности № 123 от 19 г., с одной стороны, и г **Мария Мироновна**, именуемый в дальнейшем «Клиент», с другой стороны, при совместном упоминании именуемые «Стороны», заключили настоящий Договор о нижеследующем:

1. Предметом настоящего Договора является консолидация и рефинансирование задолженности Клиента перед Кредитором по состоянию на 23.12.2019 г. по следующим кредитным договорам, права требования по которым были уступлены Кредитору:

	Наименование банка	№ кредитного договора	Дата кредитного договора	Основной долг	Начисленные проценты	Пени, штрафы	Сумма долга, руб.
1	ООО	54 113	28.05.201	50000	0	0	50000
				ИТОГО, КОНСОЛИДИРОВАННЫЙ ДОЛГ			50000

2. Задолженность Клиента по вышеуказанным кредитным обязательствам, включающая в себя основной долг, начисленные проценты, а также штрафы, консолидируется по состоянию на 23.12.2019 г. и составляет сумму 50000 (Пятьдесят тысяч) рублей 00 копеек (далее – «Консолидированный долг»);

3. Настоящим Клиент признает вышеуказанную задолженность и соглашается на рефинансирование Консолидированного долга проводится путем прощения части задолженности и рассрочки выплаты оставшейся суммы долга в соответствии с графиком, указанным в Приложении № 1 к настоящему Договору на следующих условиях:

ОБЩАЯ СУММА К ВЫПЛАТЕ:	50000 рублей
СУММА ПРОЩЕНИЯ:	15000 рублей
ЕЖЕМЕСЯЧНЫЙ ПЛАТЕЖ:	3500 рублей
КОЛИЧЕСТВО ПЛАТЕЖЕЙ:	10
ДЕНЬ ПЛАТЕЖА:	23 число каждого месяца
ДАТА ПЕРВОГО ПЛАТЕЖА:	23.12.2019

4. При наличии неоконченного исполнительного производства в отношении Клиента в ФССП, Кредитор направляет письменное заявление о его окончании в течении 10 (Десяти) рабочих дней после поступления на расчетный счет Кредитора первого платежа, внесенного Клиентом согласно условиям настоящего договора.

4. **В случае нарушений Клиентом условий настоящего Договора, в том числе установленного графика платежей, Кредитор вправе:**

4.1. Отменить прощение долга с пересчетом общей суммы к выплате в сторону увеличения.

4.2. Возобновить исполнительное производство в ФССП в отношении Клиента с применением всех мер принудительного взыскания в соответствии с законодательством РФ.

5. Стороны договорились, что в случае ненадлежащего исполнения данного Договора, заявления о выдаче судебного приказа могут быть направлены Кредитором мировому судье Судебного участка 119 Центрального района г. Волгограда, иные иски из настоящего Договора в Центральный Районный суд г. Волгограда.

6. Настоящий Договор составлен в 2 (двух) идентичных экземплярах по одному для каждой стороны, вступает в силу с даты подписания Сторонами и действует до момента полного исполнения своих обязательств.

7. Подписывая настоящий Договор, Клиент дает свое согласие Кредитору на получение и направление в бюро кредитных историй информации, составляющей кредитную историю Клиента, в соответствии с Федеральным законом №218-ФЗ от 30.12.2004 «О кредитных историях» с целью возврата просроченной задолженности.

8. Реквизиты и подписи Сторон:

Кредитор
НАО «ПКБ»

Адрес: 108811, город Москва, поселение Московский, Киевское шоссе, 22-й км, домовладение 6, строение 1
ИНН/КПП 2723115222/ 775101001 р/счет 40702810800020009245 в Филиале ПАО Банк ВТБ в г. Хабаровск к/с 30101810400000000727
БИК 040813727

Петров П.П.

Клиент
Трофимова .
Дата рождени 11.05.198
Паспорт серия 10 №25
Адрес: 655550, г ул. Строительный Проезд,

( )

ВАЖНО:

Оплата производится на сайте www.collector.ru без комиссий по номеру клиента: 10506341857
Реквизиты и другие способы оплаты Вы можете найти по адресу: **www.collector.ru/pay**

Fig. 3. Example of a refinancing agreement.

Рис. 3. Пример договора о рефинансировании.

В дальнейшем мы будем использовать следующие обозначения признаков: «регулярные выражения» — *r*, «само слово» — *v*, часть речи — *m*, нормализация — *n*, Word2Vec — *w*, FastText — *f*. Для повышения качества распознавания сущностей полезно учитывать контекст слова в тексте. Таким образом, в качестве характеристик слова мы также использовали признаки его соседей. Цифра после буквы обозначает количество рассматриваемых соседей влево и вправо. Например, *f3* означает, что использовалось 7 признаков FastText — для текущего слова и для 3-х соседей с каждой стороны.

3. Метод CRF

Как уже было сказано, наиболее популярным и традиционным подходом для решения задачи распознавания именованных сущностей является метод conditional random field (CRF) [7, 8]. Данный алгоритм оптимизирует всю цепочку меток целиком, а не каждый элемент по отдельности, учитывая особенности и взаимозависимости в данных. Поэтому данная модель хорошо подходит для решения задач сегментации и маркировки последовательностей. Основная идея CRF заключается в том, чтобы предсказать последовательность меток или классов для входной последовательности на основе набора наблюдаемых признаков. У CRF есть несколько ключевых особенностей:

1. Условная зависимость: CRF моделирует условные вероятности меток в зависимости от признаков. Модель учитывает контекст и взаимодействие между соседними элементами входной последовательности при принятии решения о метках.
2. Марковские свойства: CRF основывается на марковских свойствах, что означает, что вероятность текущей метки зависит только от предыдущих меток в последовательности. В контексте последовательностей, CRF моделирует условные вероятности меток, основываясь на предыдущих метках, а также на наблюдаемых признаках.
3. Графическая структура: CRF представляет собой графическую модель, в которой узлы соответствуют элементам входной последовательности, а ребра представляют зависимости между ними. Эта структура позволяет моделировать сложные зависимости между метками.

Рассмотрим модель подробнее. Пусть у нас есть наблюдаемые переменные (входные признаки) $X = x_0, ..., x_T$ и скрытые переменные (метки) $Y = y_0, ..., y_T$, где T - длина последовательности. CRF моделирует условные вероятности $P(Y|X)$ с использованием следующей формулы:

$$P(Y|X) = \frac{1}{Z(X)} \exp\left(\sum_k \sum_i \lambda_k f_k(y_{i-1}, y_i, x_i)\right),$$

где $Z(X)$ — нормализующий множитель, λ_k — параметры модели, f_k — функции признаков.

Нормализующий множитель гарантирует, что сумма всех возможных состояний меток в последовательности будет равна 1. Функции признаков f_k могут быть заданы различными способами, и выбор конкретных функций зависит от предметной области и структуры данных. Они могут учитывать контекстную информацию, зависимости между соседними элементами, а также другие свойства данных.

Обучение CRF модели включает настройку параметров λ_k с использованием метода максимального правдоподобия. Для нахождения оптимальных параметров модели часто применяются методы оптимизации, такие как градиентный спуск или условный градиентный спуск.

В данной работе была использована реализация CRF из библиотеки sklearn crfsuite. Ниже представлен список поддерживаемых оптимизаторов:

- *l2sgd* — стохастический градиентный спуск с регуляризацей *L2*;
- *lbfgs* — градиентный спуск с использованием алгоритма Бройдена-Флетчера-Гольдфарба-Шанно (*L1* и *L2*);

- *ap* — усредненный персептрон (Averaged Perceptron);
- *pa* — passive aggressive, алгоритм, основанный на бинарной классификации [9];
- *arow* — метод адаптивной регуляризации весов вектора.

4. Тестирование

Перечислим сначала библиотеки, которые использовались при реализации: NLTK [10], gensim [11], pymorphy2 [12], sklearn crfsuite (http://www.chokkan.org/software/crfsuite/).

Прежде чем перейти к тестированию, стоит обсудить оценку качества распознавания. Существует два подхода к оценке качества решения задачи NER: F1-мера, предложенная на конференции CoNLL [13], которая учитывает всю цепочку слов сущности; и стандартная F1-мера для каждой сущности по отдельности.

Поскольку в наших данных есть разрывные сущности, было решено использовать стандартную F1-меру для каждой сущности по отдельности. Данный подход дает несколько более оптимистичные результаты, однако тестирование показало, что метрики не сильно отличаются на наших наборах данных (0.04 ± 0.02).

4.1. Результаты на документах судебной статистики

Обучение проводилось на 241 судебном протоколе, а отложенная выборка состояла из 103 документов. В данной статье авторами были протестированы различные наборы признаков и алгоритмов оптимизации. В таблице 3 приведен анализ оптимизаторов, при фиксированных наборах признаков.

Table 3. Quality analysis. Court records.

Таблица 3. Анализ качества распознавания. Судебные протоколы.

Сущность	r1, v1 pa	r3, v3 pa	r3, v3, m3 pa	f3(10), r3, m3 lbfgs
срок обжалования	0.92	0.93	0.93	0.92
суд	0.94	0.97	0.97	0.91
решение суда	0.71	0.71	0.71	0.60
дата суда	0.89	0.94	0.92	0.82
ответчик или представитель	0.52	0.61	0.67	0.57
номер документа	0.95	0.98	0.97	0.95
судья	0.97	0.97	0.97	0.95
сумма выплаты	0.64	0.73	0.77	0.72
статья или тип штрафа	0.64	0.69	0.68	0.55
истец	0.85	0.86	0.86	0.82
F-macro	0.82	0.85	0.86	0.80

Следует отметить, что изменение количества соседей с 1 на 3 (столбцы 1 и 2) повышает качество модели. Добавление признака *m3* улучшает качество, особенно это видно для суммы платежа (прерывистая сущность) и ответчика. *Само слово* не является хорошим признаком, однако, качество сильно понизилось при замене *v3* (слово с соседями 3) на *f3* (FastText с соседями 3).

Таким образом, наименьший разброс метрики F1 для разных сущностей наблюдается на наборе *(r3,v3,m3)* с оптимизатором *pa*. Сущность *ответчик* показала наихудшее качество распознавания, по-видимому, это связано с разнообразием значений сущностей в разных документах (ли-

цо или представитель). Также видно, что качество у простых и сложных лучше, чем у прерывистых («оплата штраф/сумма»).

Table 4. Analysis of recognition quality. Refinancing agreements.

Таблица 4. Анализ качества распознавания. Договоры о рефинансировании.

Сущность	m3, r3 pa	m3, v3 pa	r1, v1, m1 pa
клиент 1	1.00	0.99	1.00
номер паспорта	0.98	0.99	0.98
номер документа	0.99	0.99	0.99
число выплат	0.98	1.00	0.98
клиент 2	0.96	0.98	0.96
сумма прощения	0.95	1.00	0.95
дата документа	0.97	0.98	0.96
день рождение	0.99	1.00	0.99
месяц оплаты	0.96	1.00	0.96
номер кредитного договора	0.98	1.00	0.98
имя банка	0.99	0.99	0.99
дата кредитного договора	0.96	0.98	0.96
итоговая сумма	0.98	1.00	0.98
дата первого платежа	0.96	0.96	0.96
день оплаты	0.98	1.00	0.98
адрес клиента	0.97	1.00	0.98
F-macro	0.98	0.99	0.98

4.2. Результаты на договорах о рефинансировании

Перейдем ко второму набору данных, в таблице 4 приведены результаты тестирования на различных наборах признаков. Обучающая выборка составила 160 документов, а отложенная — 40 документов. Стоит заметить, что метрика по всем сущностям значительно лучше, чем для предыдущего набора данных. По нашему мнению это закономерно в виду структурированности банковских документов. Как видно из таблицы, авторами был достигнут высокий результат на малом наборе данных. Однако отметим, что, поскольку документы были представлены одной организацией, то данная модель может плохо работать на документах другой.

Так как сущности в данной выборке менее вариативны, то использование самого слова как признака дает улучшение качества. Данную особенность можно увидеть во втором столбце, при использовании *v3*. Однако наиболее универсальной будет модель с признаками *(m3, r3)* (столбец 2). Последний столбец показывает, что даже используя малое количество соседей можно добиться высокого качества.

Стоит отметить, что для этих данных точного решения можно достичь с минимальным количеством трудозатрат, и поэтому он не является репрезентативным для решения задачи NER.

Заключение

В статье рассмотрен подход к решению задачи извлечения именованных сущностей из русскоязычного текста. Для решения поставленной задачи был использован метод CRF, был проведен сравнительный анализ различных алгоритмов оптимизации, алгоритм *pa* показал лучшее качество.

Авторами были использованы признаки на основе регулярных выражений, морфологии, различных векторных представлений слов, а также признаки их соседей.

Конечной целью исследования является создание универсального инструмента для извлечения различных сущностей из широкого спектра документов, поэтому в работе было проанализировано два набора данных: банковские документы имеют ярко выраженную структуру, а судебные протоколы — нет.

В ходе тестирования на различных наборах признаков было выявлено, что на неструктурированных данных качество модели растет с увеличением количества используемых соседей (но и время обучения растет полиномиально). Во время эксперимента для улучшения качества распознавания именованных сущностей хорошо себя показали признаки на основе регулярных выражений и морфологии. Заметим, что использование векторного представления существенно увеличивает время обучения и теряется возможность использования информации о соседних словах. Для структурированных документов можно получить хорошее качество даже не используя большое количество соседей.

В качестве одного из путей улучшения результатов планируется формировать дополнительные признаки. Наиболее перспективным направлением исследования является применение альтернативных подходов к извлечению именованных сущностей на основе современных нейросетевых архитектур BiLSTM и CRF.

References

[1] E. Leitner, G. Rehm, and J. Moreno-Schneider, “Fine-grained Named Entity Recognition in legal documents”, in *International Conference on Semantic Systems*, Springer, 2019, pp. 272–287.

[2] J. Straková, M. Straka, and J. Hajič, “Neural architectures for nested NER through linearization”, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5326–5331, 2019.

[3] R. Yeshpanov, Y. Khassanov, and H. A. Varol, *KazNERD: Kazakh Named Entity Recognition dataset*, 2022. arXiv: 2111.13419 [cs.CL].

[4] S. Zheng *et al.*, “Conditional Random Fields as Recurrent Neural Networks”, in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1529–1537.

[5] K. W. Church, “Word2vec”, *Natural Language Engineering*, vol. 23, no. 1, pp. 155–162, 2017.

[6] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, “Enriching word vectors with subword information”, *Transactions of the association for computational linguistics*, vol. 5, pp. 135–146, 2017.

[7] C. Sutton, A. McCallum, *et al.*, “An introduction to Conditional Random Fields”, *Foundations and Trends® in Machine Learning*, vol. 4, no. 4, pp. 267–373, 2012.

[8] J. Lafferty, A. Mccallum, and F. Pereira, “Conditional Random Fields: Probabilistic models for segmenting and labeling sequence data”, in *Proceedings of the Eighteenth International Conference on Machine Learning*, 2001, pp. 282–289.

[9] M. Collins, “Discriminative training methods for hidden Markov models: Theory and experiments with perceptron algorithms”, in *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, Association for Computational Linguistics, 2002, pp. 1–8.

[10] S. Bird, "NLTK: The natural language toolkit", in *Proceedings of the COLING/ACL on Interactive Presentation Sessions*, ser. COLING-ACL '06, Association for Computational Linguistics, 2006, pp. 69–72.

[11] R. Řehůřek and P. Sojka, "Software framework for topic modelling with large corpora", in *Proceedings of LREC 2010 workshop New Challenges for NLP Frameworks*, 2010, pp. 46–50.

[12] M. Korobov, "Morphological analyzer and generator for Russian and Ukrainian languages", in *Analysis of Images, Social Networks and Texts*, Springer, 2015, pp. 320–332.

[13] J. Li, A. Sun, J. Han, and C. Li, "A survey on deep learning for Named Entity Recognition", *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 1, pp. 50–70, 2020.

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 4, 2023
journal homepage: www.mais-journal.ru

ARTIFICIAL INTELLIGENCE

Semantic rule-based sentiment detection algorithm for Russian publicism sentences

A. Y. Poletaev[1], I. V. Paramonov[1], E. I. Boychuk[1]

DOI: 10.18255/1818-1015-2023-4-394-417

[1]P.G. Demidov Yaroslavl State University, 14, Sovetskaya str., Yaroslavl, Yaroslavl Region, 150003, Russia.

MSC2020: 68T50
Research article
Full text in Russian

Received November 6, 2023
After revision November 24, 2023
Accepted November 29, 2023

The article is devoted to the task of sentiment detecton of Russian sentences, which is understood as the author's attitude on the sentence topic expressed through linguistic expression features. Today most studies on this subject utilize texts of colloquial style, limiting the applicability of their results to other styles of speech, particularly to the publicism.
To fill the gap, the authors developed a novel publisism sentences oriented sentiment detection algorithm. The algorithm recursively applies appropriate rules to sentence parts represented as constituency trees. Most of the rules were proposed by a philology expert, based on knowledge on the expression features from Russian philology, and algorithmized using constituency trees generated by the algorithm. A decision tree and a sentiment vocabulary are also used in the work. The article contains the results of evaluation of the algorithm on the publicism sentences corpus OpenSentimentCorpus, F-measure is 0.80. The results of errors analysis are also presented.

Keywords: sentiment analysis; sentiment detection; semantic rules; publicism; constituency tree

INFORMATION ABOUT THE AUTHORS

Anatoliy Y. Poletaev corresponding author	orcid.org/0000-0003-0116-4739. E-mail: anatoliy-poletaev@mail.ru Post-graduate student.
Ilya V. Paramonov	orcid.org/0000-0003-3984-8423. E-mail: ilya.paramonov@fruct.org PhD, Associate professor.
Elena I. Boychuk	orcid.org/0000-0001-6600-2971. E-mail: elena-boychouk@rambler.ru Doctor of Science, Associate professor.

Funding: The reported study was funded by the grant of Russian Science Foundation No. 23-21-00495.

For citation: A. Y. Poletaev, I. V. Paramonov, and E. I. Boychuk, "Semantic rule-based sentiment detection algorithm for Russian publicism sentences", *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 394-417, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 4, 2023
сайт журнала: www.mais-journal.ru

ARTIFICIAL INTELLIGENCE

Алгоритм определения тональности предложений публицистического стиля на русском языке на основе семантических правил

А. Ю. Полетаев[1], И. В. Парамонов[1], Е. И. Бойчук[1]

DOI: 10.18255/1818-1015-2023-4-394-417

[1]Ярославский государственный университет им. П.Г. Демидова, ул. Советская, д. 14, г. Ярославль, Ярославская область, 150003, Россия.

УДК 004.912+10.02.21
Научная статья
Полный текст на русском языке

Получена 6 ноября 2023 г.
После доработки 24 ноября 2023 г.
Принята к публикации 29 ноября 2023 г.

Статья посвящена задаче определения тональности предложения на русском языке, понимаемой как отношение автора предложения к его теме, выраженное с помощью языковых средств. В настоящий момент большинство исследований по этой теме проводятся на текстах разговорного стиля речи, что ограничивает применимость их результатов для других стилей, в частности, публицистического.
Для того, чтобы заполнить этот пробел, авторами был разработан алгоритм определения тональности, ориентированный на применение к предложениям публицистического стиля речи. Алгоритм рекурсивно применяет подходящие правила к составным частям предложения, представленным в виде дерева синтаксических единиц. Большинство правил было построено на основе знаний эксперта-филолога относительно средств выражения тональности, известных русской лингвистике, и выбора тех из них, которые достаточно формализованы для того, чтобы их можно было алгоритмизировать с использованием генерируемых в рамках алгоритма деревьев синтаксических единиц. Также применялись дерево решений и тональный словарь. В статье приведены результаты эксперимента по апробации предложенного алгоритма на корпусе предложений публицистического стиля OpenSentimentCorpus, F-мера составила 0.80, а также результаты анализа ошибок алгоритма.

Ключевые слова: анализ тональности; определение тональности; семантические правила; публицистический стиль; дерево синтаксических единиц

ИНФОРМАЦИЯ ОБ АВТОРАХ

Анатолий Юрьевич Полетаев автор для корреспонденции	orcid.org/0000-0003-0116-4739. E-mail: anatoliy-poletaev@mail.ru аспирант.
Илья Вячеславович Парамонов	orcid.org/0000-0003-3984-8423. E-mail: ilya.paramonov@fruct.org кандидат физико-математических наук, доцент.
Елена Игоревна Бойчук	orcid.org/0000-0001-6600-2971. E-mail: elena-boychouk@rambler.ru доктор филологических наук, доцент.

Финансирование: Исследование выполнено за счет гранта Российского научного фонда № 23-21-00495.

Для цитирования: A. Y. Poletaev, I. V. Paramonov, and E. I. Boychuk, “Semantic rule-based sentiment detection algorithm for Russian publicism sentences”, *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 394-417, 2023.

Введение

Автоматический анализ тональности предложений — одна из важных и популярных задач компьютерной лингвистики, заключающаяся в определении отношения автора предложения к его теме [1]. В ходе автоматического анализа тональности конкретному предложению сопоставляется один из 2–4 классов тональности (положительный, отрицательный, нейтральный, смешанный).

Подавляющее большинство методов определения тональности предложений используют либо машинное обучение, либо сформулированные экспертами семантические правила [2—4], причём первая категория методов преобладает. Методы, основанные на семантических правилах, крайне немногочисленны даже для английского языка [5, 6] и практически полностью отсутствуют для русского.

Кроме того, бóльшая часть методов разработана для анализа предложений, извлечённых из записей в социальных сетях или интернет-отзывов на товары и услуги, относящихся преимущественно к разговорному стилю речи [7, 8]. В то же время, алгоритмы, достаточно хорошо работающие с предложениями разговорного стиля, значительно менее качественно определяют тональность предложений, относящихся к другим стилям речи. В частности, в предыдущей работе авторов [9] F_1-мера для рекурсивного алгоритма выведения тональности на корпусе интернет-отзывов на отели составила 0.75, тогда как на корпусе публицистических текстов OpenCorpora — лишь 0.70.

Такая разница в результатах, скорее всего, обусловлена тем, что для выражения авторской позиции в публицистическом стиле используется иной спектр речевых средств, чем в разговорном. Эта гипотеза подтверждается лингвистическими исследованиями, осуществленными на материале отзывов и публицистических текстов. Так, для определения тональности в отзывах лингвистами были выделены такие речевые средства, как частотное употребление специфических для данного жанра лексем, в особенности с положительной или отрицательной эмоционально-экспрессивной окраской, сочетаемость слов в предложении, роль отрицания, грамматических форм глаголов, употребление количественных наречий, прилагательных в превосходной степени [7, 10], также выделяются конкретные средства выражения положительной, отрицательной и нейтральной тональностей в интернет-отзывах [8]. Что касается публицистического текста, то его характер априори предполагает меньшую степень эмотивности, что может вести за собой проблему имплицитности проявления тональных средств, а также изменение их характера и перечня в целом [11]. Ввиду этого алгоритм определения тональности, хорошо работающий со средствами, активно используемыми в разговорном стиле, не всегда будет так же хорошо работать со средствами публицистического стиля. Данные соображения, в совокупности с общей неисследовательностью вопросов, связанных с алгоритмами анализа тональности русскоязычных текстов, основанными на семантических правилах, и определили направленность настоящей работы.

Цель данной статьи — разработать алгоритм определения тональности русскоязычного предложения, основанный на семантических правилах и ориентированный на применение к текстам в публицистическом стиле в широком смысле (новости, статьи в СМИ, записи блогов и т. п.), а также оценить его эффективность. Тональность понимается как отношение автора предложения к его теме, выраженное с помощью языковых средств. При разработке алгоритма использовались идеи рекурсивного алгоритма выведения тональности, описанного в предыдущей работе авторов [9], при этом заключает в себе более глубокий анализ языковых средств, используемых для выражения авторского отношения, в том числе в публицистическом стиле речи. Для оценки эффективности используется OpenSentimentCorpus, построенный с использованием краудсорсинга на основе корпуса публицистических текстов OpenCorpora [12].

Оставшаяся часть работы организована следующим образом. В разделе 1 приводится информация о лингвистических ресурсах, используемых в работе. В разделе 2 описывается предлагаемый

алгоритм. Раздел 3 содержит результаты экспериментов по оценке качества определения тональности алгоритмом, в том числе анализ ошибок. В заключении подведены итоги работы.

1. Лингвистические ресурсы

1.1. Корпус

Для экспериментов в работе используется OpenSentimentCorpus — корпус предложений на русском языке, извлечённых из новостных и публицистических текстов, относящихся к различным предметным областям, с разметкой по тональности [12]. Каждое предложение, входящее в OpenSentimentCorpus, оценивалось как минимум тремя разметчиками. Разметчики могли отметить предложение как имеющее положительную, отрицательную или смешанную (положительную и отрицательную) тональность, как нейтральное или как сомнительное, если его смысл было определить невозможно. В OpenSentimentCorpus вошли те предложения, для которых подавляющее большинство оценивавших их разметчиков сошлись во мнениях, и ни один разметчик не оценил их как сомнительное. Таким образом, в корпус были включены только предложения, мнение о тональности которых достаточно единообразно, и эксперименты, проводимые на такой разметке, можно считать достаточно показательными. Так как в данной работе решается задача классификации на три класса тональности, в ходе экспериментов использовались только предложения из OpenSentimentCorpus с положительной, отрицательной или нейтральной тональностью.

Поскольку точность исходной разметки для лингвистических методов является существенной, перед началом работы над алгоритмом OpenSentimentCorpus был выборочно перепроверен экспертом-лингвистом. Для перепроверки было случайно выбрано 240 предложений (по 80 каждого класса), разметка которых проверялась на соответствие используемому определению тональности. Было обнаружено, что 18 из проверенных экспертом предложений были размечены неверно, для них разметка была исправлена. Большая часть встреченных ошибок относилась к предложениям, содержащим косвенное цитирование, и ошибочно принятым разметчиками за тональные, например: «В ряде случаев, по её мнению, законопроекты вступают в противоречие с действующим законодательством и создают условия для ограничения конституционных прав и свобод граждан» или «Женщина считает, что руководство клиники, таким образом, нарушило её конституционное право на свободу вероисповедания».

Всего OpenSentimentCorpus содержит 4487 предложений, из которых 533 имеют положительную тональность, 1495 — отрицательную, а оставшиеся 2459 нейтральны.

1.2. Словари тональных слов

Для русского языка доступны два достаточно развитых словаря тональных слов: RuSentiLex-2017 [13] и KartaSlovSent [14]. Поскольку для анализа тональности крайне важно качество используемого тонального словаря, была проведена выборочная экспертная оценка этих словарей. В ходе экспертной оценки из каждого словаря было выбрано по 250 слов каждого из классов тональности. Эксперты определяли, действительно ли те слова, которые обозначены в словаре как имеющие положительную или отрицательную тональность, являются признаками выражения автором эмоции или вынесения оценок вне зависимости от контекста, в котором они употреблены. Также в ходе оценки определялось, насколько сильно в словарях пересекаются наборы тональных слов.

Было выявлено, что набор одиночных тональных слов в KartaSlovSent гораздо шире, чем в RuSentiLex-2017, но в RuSentiLex также есть достаточно много тональных слов, отсутствующих в KartaSlovSent. Также был обнаружен существенный недостаток KartaSlovSent: достаточно большому числу нейтральных слов, которые могут использоваться как в положительном, так и в отрицательном контексте, присвоены положительные метки тональности, например, глаголам «уметь», «основать», «изучить», существительному «сила». Это может быть вызвано тем, что разметчики, видевшие эти слова вне контекста, считали, что чаще всего эти слова встречаются в положи-

тельном контексте, и присваивали им положительную метку. Однако поскольку сами по себе эти слова нейтральны и, более того, могут встречаться в языке в фразах совершенно разной тональности, например, «сила гравитации», «сила благотворительности» и «сила коррупции» — включение их в словарь тональных слов, используемый алгоритмом, основанным на правилах, будет приводить к ошибкам определения тональности предложения. В то же время экспертная оценка показала, что состав слов, которым в KartaSlovSent назначена отрицательная метка тональности, достаточно точен.

2. Алгоритм определения тональности

В данной работе используются наработки рекурсивного алгоритма выведения тональности предложений на русском языке, основанного на использовании дерева синтаксических единиц [9]. Этот алгоритм рассматривается в качестве исходного. Данная работа содержит ряд существенных улучшений исходного алгоритма, основанных на более тонком анализе механизмов выражения тональности в русском языке, характерных в том числе для предложений публицистического стиля.

2.1. Общая схема работы

Алгоритм определяет тональность фразы (узла дерева синтаксических единиц) по тональностям её частей (потомков этого узла) с помощью набора правил, описывающего, как именно в языке выражается тональность. Тональности отдельных слов (листьев дерева синтаксических единиц), а также устойчивых словосочетаний, определяется с помощью тонального словаря. Опишем схему работы алгоритма более формально. Пусть N — синтаксическая единица (фраза), $C(N)$ — множество дочерних синтаксических единиц, из которых состоит N, а $S(N)$ — тональность N. Алгоритм начинает свою работу от корня дерева синтаксических единиц N_r, представляющего собой всё предложение; $S(N_r)$ — искомая тональность предложения. Тональность $S(N)$ для заданного N вычисляется следующим образом:

- если текст N присутствует в тональном словаре, то $S(N)$ — словарная тональность N;
- иначе, если $C(N) = \emptyset$, т. е. фраза состоит из одного слова, отсутствующего в тональном словаре, то $S(N)$ — нейтральная тональность;
- иначе рекурсивно вычислить $S(N_c)$ для каждого $N_c \in C(N)$, выбрать подходящее правило и вычислить по нему $S(N)$.

В зависимости от того, как тональность фразы определяется по тональностям её частей, было выделено две группы правил.

В первой группе тональность фразы является результатом простого соединения тональностей своих частей. Например, нейтральное подлежащее «законопроект», соединяясь с положительным определением «своевременный», делает тональность фразы «своевременный законопроект» положительной. Результат соединения может быть различным в зависимости от состава фразы. Создание этой группы правил рассмотрено в подразделе 2.4.

Вторая группа правил описывает определение тональности фраз, содержащих специальные языковые средства выражения тональности, то есть таких, тональность которых не является результатом простого соединения тональностей их частей. Наиболее большую группу среди них составляют фразы с отрицаниями, разработка правила для определения тональности которых рассмотрена в подразделе 2.2. Остальные правила описаны в подразделе 2.5.

Ещё одна группа относится к фразам, состоящим из однородных членов предложения, то есть представленным синтаксическими единицами типов однородные-подлежащие, однородные-сказуемые и т. п., например, «ум, честь и совесть» (положительная тональность), «разработанный коллективно и обречённый на неудачу» (отрицательная тональность). Для данных случаев было составлено следующее правило определения тональности: фраза имеет положительную тональность, если хотя бы одна из её однородных частей имеет положительную тональность, и тональность

Table 1. Baseline sentiment classification performance

Таблица 1. Качество работы исходного алгоритма

Класс предложений	Точность	Полнота	F_1-мера	Количество предложений
Положительный	0.46	0.39	0.43	533
Нейтральный	0.76	0.76	0.76	2459
Отрицательный	0.69	0.72	0.71	1495
Среднее	0.64	0.62	0.63	4487
Взвешенное среднее	0.70	0.70	0.70	4487

Доля правильных ответов алгоритма (accuracy) = 0.70

Table 2. Baseline sentiment classification confusion matrix

Таблица 2. Матрица ошибок исходного алгоритма

предсказ. / реальн.	Положит.	Нейтр.	Отрицат.	Всего
Положительная	209	249	75	533
Нейтральная	179	1873	407	2459
Отрицательная	62	355	1078	1495

ни одной из частей не отрицательна; фраза имеет отрицательную тональность, если хотя бы одна из её однородных частей имеет отрицательную тональность, и тональность ни одной из частей не положительна; иначе фраза нейтральна.

В данной работе используется построитель деревьев синтаксических единиц, описанный в [15], использующий более точную, по сравнению с [9], грамматику — в частности, он различает прямые и косвенные дополнения, а также различные типы придаточных предложений. Показатели качества работы исходного алгоритма приведены в таблицах 1 и 2. Как можно видеть, такой алгоритм достаточно неплохо отделяет предложения с отрицательной тональностью от нейтральных, однако с трудом обнаруживает положительную тональность.

2.2. Правило для фраз с отрицаниями

В рамках данного исследования фраза с отрицанием — синтаксическая единица, состоящая из двух частей, одна из которых — отрицание «не», «нет», «ни один»; например, «не ошибаться» или «не был удачным» (более сложные варианты фраз с отрицанием не рассматриваются). Наиболее сложный вопрос обработки таких фраз — определение тональности фразы, отрицаемая часть которой нейтральна. Если отрицание фразы с положительной тональностью, как правило, имеет отрицательную тональность, а отрицание фразы с отрицательной тональностью — положительную, то отрицание нейтральной информации может иметь отрицательную тональность, например, «Объяснений от Павла мы не получили», либо быть нейтральным, например, «Пик работы АСВ не пройден».

Для того, чтобы построить правило для определения тональности фраз с отрицаниями, из корпуса были выбраны все уникальные фразы-отрицания (синтаксические единицы). Всего таких фраз оказалось 861. Каждая фраза была вручную размечена по тональности двумя экспертами, и для 239 фраз их разметка совпала. Эта разметка была принята как близкая к объективной. Поскольку, как показано в таблице 3, в ней оказалось достаточно много как отрицательных, так и нейтральных отметок, дальнейшая работа проводилась именно на этих 239 фразах.

В первую очередь, на рассматриваемых фразах были проверены два правила для обработки отрицаний, использующие только информацию о тональности отрицаемой части: первое считало тональность фразы с отрицаемой нейтральной частью отрицательной (т. е. было аналогично тому, которое использовалось в работе [9]) и показало точность определения итоговой тональности фразы 0.51; второе считало тональность фразы с отрицаемой нейтральной частью отрицательной и показало точность 0.46. Поскольку такой точности явно недостаточно для использования этих правил

Table 3. Distribution of sentiment marks of negated parts of phrases with negations and entire phrases with negations

Таблица 3. Распределение оценок тональности отрицаемой части фраз с отрицаниями и фраз с отрицаниями целиком

Тональность	Положительная	Нейтральная	Отрицательная
отрицаемой части	25	190	24
фразы целиком	38	84	117

в алгоритме, было принято решение использовать для более точного выведения тональности фраз с отрицаниями дополнительную информацию.

Для построения более точного алгоритма для каждой из 239 фраз была собрана следующая информация:

- тональность отрицаемой части,
- тип синтаксической единицы отрицаемой части,
- частеречная разметка (PoS-теги) всех слов в отрицаемой части,
- время всех глаголов отрицаемой части, если они в ней присутствуют,
- лицо всех слов отрицаемой части, которые изменяются по лицу.

Для того, чтобы определить, по каким из имеющихся признаков можно наиболее точно определить тональность фразы в целом, на этих данных было построено дерево решений [16]. Дерево решений строится итеративно, на каждом шаге среди всех возможных критериев присвоения наблюдению класса (т. е. определения тональности фразы с отрицанием) выбирается тот, с помощью которого можно наиболее точно определить класс наибольшего числа наблюдений среди тех, которым на предыдущих шагах класс присвоен ещё не был. Процесс останавливается, когда достигается максимальное число критериев, предварительно выбираемое вспомогательным алгоритмом кросс-валидации так, чтобы минимизировать количество неверно классифицированных наблюдений на каждой из валидационных выборок. Каждому из критериев дерева решений была дана лингвистическая интерпретация, и на их основе был построен следующий алгоритм.

Определяя тональность фразы с отрицанием, алгоритм последовательно перебирает пункты следующего списка, пока не встретит пункт, условие которого выполняется для анализируемой фразы, после чего определяет тональность в соответствии с этим пунктом:

1. Если отрицаемая часть положительна, то тональность всей фразы отрицательна; если отрицаемая часть отрицательна, то тональность всей фразы положительна.
2. Если отрицаемая часть — подлежащее или определение, то тональность всей фразы нейтральна, например:
   - Не все в правящей партии согласны с решением Асо (отрицаемая часть — подлежащее «все»).
   - «Прощание в Стамбуле» и пока не изданные «Гавани Луны» — очень североамериканские романы, очень любовно-криминальные, очень мейлеровские; «Табор уходит», напротив, очень латиноамериканский (отрицаемая часть — определение «изданные»).
   - Вивек Кундра, однако не единственный помощник Обамы по IT-вопросам (отрицаемая часть — определение «единственный»).

   Это условие можно интерпретировать так: подлежащее и определение в предложении, в первую очередь, задают тему, которой посвящено предложение, поэтому если в них и отрицается какая-либо нейтральная информация, то фраза остаётся нейтральной и не влияет на общую тональность предложения.
3. Если в отрицаемой части присутствует существительное, то тональность всей фразы отрицательна, например:
   - Делегация США была не в состоянии прокомментировать вопросы, касающиеся предполагаемых секретных тюрем (отрицаемая часть — «быть в состоянии»).

- Дело в том, что птицам стало не хватать пищи (отрицаемая часть — «стало хватать пищи»).
- В фильме нет ощущения реальности происходящего (отрицаемая часть — «ощущение реальности»).

Это условие можно интерпретировать так: сказуемые и дополнения сообщают о каких-либо свойствах происходящих процессов, действий, и если в них встречается существительное с отрицанием, то такая фраза чаще всего выражает мысль автора о том, что этот объект для действия важен, но он отсутствует, что придаёт фразе отрицательную окраску; поскольку основной смысл передаётся именно сказуемым — ремой предложения, то и сообщение об отсутствии какого-либо объекта часто приводит к формированию отрицательной тональности. Особым случаем является последний пример: в нём отрицание «нет» само по себе является сказуемым. Для публицистического стиля вообще нехарактерна такая краткость и категоричность, поэтому если в нём встречается такое построение предложения, то это является признаком того, что автор ожидал наличие какого-то объекта и для него крайне важным оказалось то, что он отсутствовал, что характеризует отрицательную тональность. Можно сказать, что такое применение отрицания — доведённый до крайности вариант с существительным в составе сказуемого, в котором отрицательное отношение выражалось через то, что для какого-либо процесса не хватало объекта, а в случае, когда отрицание само по себе является сказуемым, это показывает, что отсутствие объекта важно для нарратива как такового.

4. Если в отрицаемой части отсутствует краткое причастие, деепричастие или глагол в личной форме, то тональность всей фразы нейтральна, например:
   - Предложенное решение поддержано не всеми: некоторые астрономы предлагают провести черту за Нептуном, а ледяные карлики наподобие Плутона отнести к транснептуновым объектам в поясе Койпера (отрицаемая часть — местоимение «всеми»).
   - 31 год — возраст для самолёта не маленький (отрицаемая часть — прилагательное «маленький»).

   В то же время, наличие в отрицаемой части краткого причастия, деепричастия или глагола в личной форме может свидетельствовать об отрицательной тональности:
   - Объяснений от Павла мы до сих пор не получили (в отрицаемой части присутствует глагол в личной форме «получили»).
   - Замминистра напомнил, что доказательств наличия российских войск на Украине так и не было представлено (в отрицаемой части присутствует глагол в личной форме «было»).
   - Подготовленный план так и не был принят (в отрицаемой части присутствует краткое причастие «принят»).

   Эта часть правила позволяет отделить фразы, в которых автор сообщает, что упоминаемый объект не совершил какого-либо действия, так как такие фразы могут оказаться отрицательными, от всех остальных фраз с отрицаниями, для которых нет причин считать их выражающими отрицательное авторское отношение.
5. Если отрицаемая часть — составное сказуемое или обстоятельство, то тональность всей фразы отрицательна, например:
   - Мы должны ясно сказать руководству России, что она не может рассчитывать на партнёрство с Западом (отрицаемая часть — составное сказуемое «может рассчитывать»).
   - Во дворах всегда есть какой-то участок, который дворники не хотят убирать, потому что не могут решить, кому он принадлежит (отрицаемая часть — составное сказуемое «хотят убирать»).

- Небольшие независимые издательства, которые отчасти выполняют эту роль, живут за счёт государственных грантов, не рассчитывая на рынок (отрицаемая часть — обстоятельство «рассчитывая»).
- В 1842 году мать Натальи Дмитриевны умрёт, не встретившись с дочерью (отрицаемая часть — обстоятельство «встретившись»).

Это условие можно интерпретировать так: с помощью составного глагольного сказуемого автор обычно сообщает о характеристиках какого-либо действия, например, возможно ли оно, желаемо ли оно, поэтому отрицание при такой фразе чаще всего свидетельствует о том, что упоминаемое действие не соответствует какой-то ожидаемой или желаемой характеристике, и с помощью него автор выражает своё отрицательное отношение. С помощью составного именного сказуемого автор обычно сообщает о свойстве объекта-подлежащего; если в публицистическом тексте автор явно делает целью высказывания (ремой предложения) сообщение о свойстве объекта и категорично употребляет именно отрицание, то это чаще всего свидетельствует, как упоминалось ранее, о том, что автор ожидал, что подлежащее будет обладать каким-либо свойством, но это оказалось не так, и это часто служит признаком отрицательного авторского отношения. Аналогичная ситуация с отрицаниями в обстоятельствах — они часто выражают отрицательное авторское отношение за счёт того, что явно говорят о том, что действие-сказуемое не обладает каким-то свойством, которого можно было бы ожидать.

6. Если в отрицаемой части присутствует краткое причастие, то тональность всей фразы нейтральна, например:
   - Объяснений от Павла нами получено не было (отрицаемая часть — «было получено»).
   - Все книги из собрания Баварской библиотеки, которые более не защищены авторским правом, будут переведены в цифровой формат (отрицаемая часть — «защищены»).
   - Пик работы АСВ не пройден (отрицаемая часть — «пройден»).
7. Если отрицаемая часть — простое глагольное сказуемое (т. е. ни одно из предыдущих условий не выполняется), то тональность всей фразы отрицательна, например:
   - Объяснений от Павла мы не получили (отрицаемая часть — «получили»).
   - Я не верю никакому телевидению, я не слушаю никакого радио и не читаю никаких газет (отрицаемые части речи — «верю», «слушаю», «читаю»).
   - Руду и книги все эти имиджмейкеры и спичрайтеры точно не производят (отрицаемая часть — «производят»).

   Эта часть правила основывается на учёте разницы между активным и пассивным залогом — в случае пассивного залога, выраженного с использованием краткого причастия, в предложении публицистического стиля фраза часто служит простым сообщением о том, что оно не совершалось; предложение становится близким к констатации факта, к официально-деловому стилю, который преимущественно нейтрален. Тогда как в случае использования автором активного залога предложение становится не таким «отстранённым», чаще явно видится, что автор не равнодушен к тому, о чём говорит, что видно, если сравнить высказывания «Объяснений от Павла нами получено не было» и «Объяснений от Павла мы не получили» — во втором гораздо более явно видна авторская позиция, выражающаяся в том, что объяснений, очевидно, ждали, но Павел их не предоставил.

Точность определения правилом тональности итоговой тональности фразы составила 0.64, что значительно выше, чем у обоих более простых правил, ориентировавшихся только на тональность отрицаемой части.

Метрики качества анализа тональности предложений и матрица ошибок с новым правилом определения тональности фраз с отрицаниями приведены в таблицах 4 и 5. Поскольку с новым правилом F_1-мера выросла на 1 %, новое правило включено в алгоритм. Однако нужно отметить,

Table 4. Sentiment classification performance with negation phrases sentiment detection rule

Таблица 4. Качество работы алгоритма с внедрённым правилом выведения тональности фраз с отрицаниями

Класс предложений	Точность	Полнота	F_1-мера	Количество предложений
Положительный	0.47	0.41	0.44	533
Нейтральный	0.76	0.78	0.77	2459
Отрицательный	0.72	0.70	0.71	1495
Среднее	0.65	0.63	0.64	4487
Взвешенное среднее	0.71	0.71	0.71	4487

Доля правильных ответов алгоритма (accuracy) = 0.71

Table 5. Sentiment classification confusion matrix with negation phrases sentiment detection rule

Таблица 5. Матрица ошибок алгоритма с внедрённым правилом выведения тональности фраз с отрицаниями

предсказ. / реальн.	Положит.	Нейтр.	Отрицат.	Всего
Положительная	220	244	69	533
Нейтральная	184	1926	349	2459
Отрицательная	63	379	1053	1495

что, хотя для класса положительных предложений точность и полнота также увеличились, они всё ещё остались достаточно низкими, а значительное число предложений с отрицательной тональностью всё так же определяются алгоритмом как нейтральные.

2.3. Расширение тонального словаря

Поскольку алгоритм ошибочно определяет значительную долю тональных предложений предложений как нейтральные, для построения качественного алгоритма необходимо улучшить используемый тональный словарь. Поскольку экспертная оценка показала, что состав слов, которым в KartaSlovSent назначена отрицательная метка тональности, достаточно точен и при этом значительно шире, чем в RuSentiLex-2017, было выдвинуто предположение, что качество обнаружения отрицательных предложений может быть повышено, если дополнить используемый словарь RuSentiLex-2017 словами с отрицательной тональностью из KartaSlovSent.

Для этого из KartaSlovSent были выбраны слова, для которых доля разметчиков, которые не смогли определить тональность слова (dunno) составила не более 0.2, агрегированный показатель тональности (value, изменяется от -1 до 1) составил максимум -0.75, а показатель расхождения оценок между разметчиками (pstvNgtvDisagreementRatio) составил не более 0.05. Таких слов оказалось 5853, среди них 2414 не входили в RuSentiLex-2017; эти 2414 слов были добавлены в тональный словарь.

Метрики качества анализа тональности и матрица ошибок для алгоритма с расширенным тональным словарём приведены в таблицах 6 и 7. Поскольку полнота определения предложений с отрицательной тональностью за счёт расширения словаря возросла и вместе с тем увеличилась и средняя F_1-мера, расширенный словарь был включён в итоговую версию алгоритма.

2.4. Автоматически подобранный набор правил рекурсивного выведения тональности

Используемый в исходном алгоритме набор рекурсивных правил выведения тональности фразы по тональностям её составляющих составлен для упрощённой грамматики всего с 5 типами синтаксических единиц — например, в нём не отличаются прямые дополнения от косвенных. Была предпринята попытка повысить качество определения тональности алгоритмом за счёт построения набора правил для грамматики с 10 типами синтаксических единиц, в соответствии с которой работает построитель дерева синтаксических единиц [15]. Такой набор может позволить более качественно определять итоговую тональность за счёт того, что для разных типов синтаксических единиц будут использоваться различные правила и появится возможность учитывать больше зако-

Table 6. Sentiment classification performance with extended sentiment dictionary

Таблица 6. Качество работы алгоритма с расширенным тональным словарём

Класс предложений	Точность	Полнота	F_1-мера	Количество предложений
Положительный	0.49	0.41	0.45	533
Нейтральный	0.77	0.77	0.77	2459
Отрицательный	0.72	0.75	0.73	1495
Среднее	0.66	0.65	0.65	4487
Взвешенное среднее	0.72	0.72	0.72	4487

Доля правильных ответов алгоритма (accuracy) = 0.72

Table 7. Sentiment classification confusion matrix with extended sentiment dictionary

Таблица 7. Матрица ошибок алгоритма с расширенным тональным словарём

предсказ. / реальн.	Положит.	Нейтр.	Отрицат.	Всего
Положительная	220	244	69	533
Нейтральная	184	1926	349	2459
Отрицательная	63	379	1053	1495

номерностей русского языка, например, то, что прямое и косвенное дополнение могут по-разному влиять на тональность — тональность фраз «остановить бомбардировки» и «остановить бомбардировками», очевидно, различна.

Поскольку количество возможных в грамматике сочетаний синтаксических единиц при этом значительно больше (22 вместо 6) и составить все соответствующие правила вручную с помощью экспертов становится сложнее, набор правил подбирался автоматически. Для каждой пары синтаксических единиц необходимо было вывести 7 правил: по одному правилу для каждой из шести возможных комбинаций несовпадающих меток тональности и одно правило для случая, когда тональность обеих частей синтаксической единицы отрицательна. Такое решение обусловлено тем, что в случае, когда обе синтаксические единицы положительны или нейтральны, тональность полученной в результате их соединения фразы практически всегда является, соответственно, положительной или нейтральной, тогда как для отрицательной тональности это далеко не всегда верно, например, фраза «ненавидеть лжецов» отрицательной не является, хотя обе её части и отрицательны.

Для того, чтобы выбрать среди всех возможных наборов правил тот, который лучше всего отражает существующие в русском языке закономерности проявления тональности, использовалась кросс-валидация. Весь корпус был разбит на 5 частей, и было сформировано 5 пар обучающих и валидационных наборов предложений. Для каждого из обучающих наборов был определён набор правил, обеспечивающий максимальную F_1-меру на обучающем наборе; после этого среди пяти таких наборов был выбран тот, для которого наивысшей оказалась F_1-мера на соответствующем валидационном наборе. Существенного снижения показателей качества на валидационных наборах по сравнению с обучающими отмечено не было.

Метрики качества и матрица ошибок алгоритма с автоматически подобранным набором правил рекурсивного выведения тональности приведены в таблицах 8 и 9. Поскольку среднее качество классификации увеличилось, а существенного снижения не было замечено ни для одного из классов, исходное предположение было признано верным, и новый набор правил был включён в алгоритм.

2.5. Отдельные правила для обработки различных средств выражения тональности

Правила, описанные в данном разделе, были построены на основе знаний эксперта-филолога (одного из авторов работы) относительно средств выражения тональности, известных русской лингвистике, и выбора тех из них, которые достаточно формализованы для того, чтобы их можно было

Table 8. Sentiment classification performance with novel recursive sentiment detection ruleset

Таблица 8. Качество работы алгоритма с новым набором правил рекурсивного выведения тональности

Класс предложений	Точность	Полнота	F_1-мера	Количество предложений
Положительный	0.50	0.44	0.47	533
Нейтральный	0.79	0.78	0.78	2459
Отрицательный	0.73	0.77	0.75	1495
Среднее	0.67	0.66	0.67	4487
Взвешенное среднее	0.73	0.74	0.73	4487

Доля правильных ответов алгоритма (accuracy) = 0.74

Table 9. Sentiment classification confusion matrix with novel recursive sentiment detection ruleset

Таблица 9. Матрица ошибок алгоритма с новым набором правил рекурсивного выведения тональности

предсказ. / реальн.	Положит.	Нейтр.	Отрицат.	Всего
Положительная	235	233	65	533
Нейтральная	179	1912	368	2459
Отрицательная	55	284	1156	1495

алгоритмизировать с использованием генерируемых в рамках алгоритма деревьев синтаксических единиц.

1. Обработка синтаксических единиц (фраз) с частицей «бы».
   - Сама по себе частица «бы» не влияет на тональность фразы.
   - Синтаксическая единица, одна из частей которой «хотелось бы» или «хорошо бы», нейтральна, например (здесь и далее подчёркнута синтаксическая единица, тональность которой определяется):
     – Конечно, очень хотелось бы сказать, что открылись ворота, и к нам хлынули прекрасные фильмы Вуди Аллена или Фрэнсиса Копполы, но чёрта с два.
     – А во второй части мне хотелось бы написать про подготовку к отъезду и рассказать подробно о самой школе Marktoberdorf Summer School.
     – Хотелось бы самому посмотреть сочинские объекты.
     – Конечно, против Китая хорошо бы дружить напрямую с США, у которых с военной мощью всё хорошо.

     Во всех предложениях фраза, вводимая с помощью «хотелось бы», нейтральна: в первом предложении отрицательная тональность выражается с помощью фразеологизма «чёрта с два», второе и третье предложения нейтральны. Можно объяснить это тем, что конструкция «хотелось бы» употребляется, как правило, просто для сообщения о каком-то событии, и реальное отношение, если оно присутствует, автор выражает с помощью других фраз и конструкций. Само «хотелось бы» может как просто сообщать о намерении, например, «мне хотелось бы поехать в Дубай», так свидетельствовать о намерении, которое не осуществилось, например, «мне хотелось бы поехать в Дубай, но еду в Геленджик» — но в этом случае важно противительное придаточное с союзом «но»; с помощью «хотелось бы» автор может даже сказать о том, что его намерение осуществилась — «мне хотелось бы написать и я пишу». Используя оборот с «хорошо бы», автор выражает положительное отношение к возможному событию, но не говорит о том, реализовалось ли оно, и может ли реализоваться вообще, поэтому такую фразу в контексте анализа тональности предложений предпочтительнее считать нейтральной.
   - Синтаксическая единица, одна из частей которой — «лучше бы» или «лишь бы», имеет отрицательную тональность, например:

  - Он лучше бы за подчинёнными следил.
  - Лишь бы это не привело к печальным итогам.
  - Правительства склонны делать всё, лишь бы смотреться не хуже других.

  Фраза «лучше бы» выражает, что сейчас кто-то поступает неправильно, и автор этим эмоционально недоволен. Отличие «лучше бы» от «хорошо бы», в первую очередь, в том, что «хорошо бы» просто сообщает о некотором желаемом событии, а «лучше бы» явно сообщает, что сейчас дела идут не так, как хотелось бы автору. «Лишь бы», как и другие конструкции с «бы», придаёт неуверенность, говорит о том, что действие ещё не случилось, отчасти формирует сослагательное наклонение. «Лишь», в свою очередь, показывает отрицательное отношение к действию, что оно либо является не лучшим, с точки зрения автора, исходом, как в первом примере, либо что оно само по себе не ценно, тогда как усилия, затрачиваемые на него, чрезмерны, как во втором примере.
- Синтаксическая единица, состоящая из двух частей, в состав одной из которых входит «хотя бы», а другой — «могли бы», имеет отрицательную тональность, что обусловлено вкладываемым смыслом, заключающимся в том, что надежды автора не оправданы, например:
  - Хотя бы «Матрицу» авторы могли бы посмотреть.

2. Тональность синтаксической единицы, одна из частей которой — синтаксическая группа глагола «отобрать» или образованного от него слова, а вторая — группа косвенного дополнения с предлогом «у», отрицательна, например:
   - Спор возникает из-за отобранных у Мексики территорий.
   - Каждого такого бизнесмена я бы обложила специальным целевым пенсионным налогом, из которого делала бы доплаты к пенсии всем людям, работавшим в советский период, чтобы хотя бы частично скомпенсировать то, что у них отобрали.

   Глагол «отобрать» ближе к разговорной речи, употребляя его в публицистическом тексте, автор явно показывает, что лишение кого-либо чего-либо произошло не по доброй воле и достаточно грубо, и он не считает это лишение правильным.
3. Тональность фразы, описывающей переход из одного состояния в другое, зависит от тональностей исходного состояния и результата изменений.
   - Если результат имеет отрицательную тональность, то тональность фразы также отрицательна, например: «Поэтому вся переаттестация превратилась в фарс».
   - Если результат имеет положительную тональность, то тональность фразы также положительна, например: «Практика динамической медитации преобразует гнев в сострадание».
   - Если результат имеет нейтральную тональность, а исходное состояние — положительную, то тональность фразы отрицательна, например: «Акунин из автора стильной детективной прозы превратился в обыкновенного массового писателя».
   - Если результат имеет нейтральную тональность, а исходное состояние — отрицательную, то тональность фразы положительна, например: «Индонезия превратилась из диктатуры в нормальную страну».
   - Если тональности исходного состояния и результата совпадают, то фраза имеет такую же тональность, например: «Недавно страна превратилась из экспортёра нефти в покупателя».

   Правило применяется к синтаксической единице, одна из частей которой — синтаксическая группа сказуемого «превратить», «превратиться» или «преобразовать», а вторая — косвенное дополнение с предлогом «в». Это косвенное дополнение называет результат изменений. Если у сказуемого нет возвратного суффикса, то исходное состояние — прямое дополнение в составе синтаксической группы сказуемого («превращает диктатуру в нормальную страну»);

если же у сказуемого есть возвратный суффикс, то исходное состояние — косвенное дополнение с предлогом «из» в синтаксической группе сказуемого («превратилась из диктатуры в нормальную страну»), а если его нет, то исходное состояние — подлежащее («диктатура превращается в нормальную страну»).

4. Тональность фразы с глаголом «позволять» и образованными от него словами, сообщающей о возможности некоторого нейтрального действия, положительна, если называется тот, у кого есть эта возможность, например:
   - Сделка позволяет компании получать средства для своих сервисов.
   - Размещение научной статьи в журнале открытого доступа позволит учёному заявить об авторстве на идею.
   - Это разработка, позволяющая любому человеку написать программу для мобильной платформы Android.

   В то же время, если объект, у которого существует возможность, не называется, то такая фраза нейтральна:
   - Новые методы позволяют определять присутствие синтетического тестостерона в организме.
   - Реклама корректирующей жидкости Tipp-Ex позволяет выбирать сценарий ролика и писать его продолжение «самостоятельно».
   - Telegram — бесплатный мессенджер для смартфонов, позволяющий обмениваться текстовыми сообщениями и файлами.

   Правило применяется к синтаксической единице, одна из частей которой — группа составного глагольного сказуемого со вспомогательной частью «позволять» или группа определения «позволяющий», к которому присоединено косвенное дополнение, выраженное глаголом в начальной форме, а вторая — группа косвенного дополнения в дательном падеже без предлога.

   Это правило отражает своеобразную «солидаризацию» автора предложения с тем, у кого возникает возможность, когда он явно сообщает, у кого именно она появилась; при сообщении же о возможности, которая просто есть у неназываемого лица, такой «солидаризации» не происходит. Разница в семантике невелика, однако при нейтральной тональности она может привести к разнице в итоговой тональности фразы.
5. Тональность фразы, сообщающей о том, что оцениваемое нейтрально событие привело к положительным последствиям, положительна, например: «Сегодня наши усилия привели к успеху». Правило применяется к синтаксической единице основы предложения с нейтральным подлежащим и сказуемым «привести», в группу которого входит косвенное дополнение с предлогом «к», тональность которого положительна.
6. Тональность синтаксической единицы, одна часть которой — глагол «использовать» или «искать» или образованное от него слово, а вторая — группа подлежащего или прямого дополнения с положительной тональностью, нейтральна, например:
   - Точность GPS до миллиметров используется в гражданских целях.
   - Я ищу актрису, которая идеально подходит по образу.

   Данное правило отражает то, что глагол «использовать» часто употребляется в информационных сообщениях, когда автор пытается сообщить о факте, а не выразить своё отношение к нему; а с помощью глагола «искать» автор сообщает, что желаемое ещё не найдено, и даже если к этому желаемому он относится положительно, радость по его поводу будет преждевременна.

7. Тональность синтаксической единицы, одна часть которой — группа глагола «измерять» или «изучать» или образованного от него слова, а вторая — подлежащее либо прямое дополнение, нейтральна, например:
   - В процессе демонстрации измерялась деградация зрительной коры головного мозга.
   - Можно решать обратную задачу: напрямую измерить удовлетворённость жизнью и посмотреть, насколько она соответствует индексам.
   - Он начал изучать изобразительное искусство в 70-х годах, посещая занятия в нескольких арт-колледжах США.

   Это правило было введено, поскольку, если автор сообщает о том, что какое-то явление измерялось или изучалось, то он, скорее, рассматривает его с позиции объективного исследователя, таким образом, такое упоминание явления будет скорее нейтральным, и оно не влияет на итоговую тональность предложения.
8. Синтаксические группы слов «понятие», «концепция» или «определение» нейтральны, например:
   - Данная форма эволюции элит соответствует концепции навязывания.
   - Для Запада понятия «демократия», «свобода слова», «независимый суд», «права человека» и т. п. сверхценностны.

   Данное правило отражает то, что, если автор предложения не называет явление напрямую, а говорит, например, о его «концепции», то он показывает, что относится к нему как исследователь, беспристрастно, и такая фраза не будет выражать ни положительное отношение автора, ни отрицательное.
9. Фразы, состоящая из группы дополнения «на тему», нейтральны вне зависимости от тональности остальных составляющих группы, например:
   - Ранее на тему допинга в России высказывались прыгунья в длину Джейд Джонсон, бегун Дэй Грин и главный тренер сборной Великобритании по легкой атлетике Петер Эрикссон.
   - Вице-премьер Дмитрий Рогозин провёл совещание на тему решения проблем с выплатой заработных плат.
10. Составные сказуемые с глаголом «является» нейтральны, если их главная часть положительна или нейтральна, например: «Наиболее знаменитым продуктом компании является усилитель NAD 3020». Это правило отражает то, что глагол «является» свойственен скорее научному или официально-деловому стилю речи и, используя его, автор скорее отстраняется от того, чем именно является упоминаемый объект.
11. Фраза-сообщение о том, что объект обладает каким-либо нейтральным свойством, обладает положительной тональностью, если она является предикативным ядром (основой) предложения, например:
   - Сейчас правящая коалиция обладает большинством в Кнессете.
   - Усилители NAD обладают звуком, который обычно присущ аппаратам более высокой ценовой категории.

   В то же время, если сообщение об обладании не является предикативным ядром предложения, то оно нейтрально:
   - Израиль — ближневосточная страна, обладающая ядерным оружием.

   Правило отражает то, что само по себе слово «обладать» является достаточно возвышенным, относящимся скорее к литературно-художественному стилю, и, если автор делает на нём акцент, вынося его в основу предложения, то это признак того, что, с точки зрения автора, факт обладания важен, и, если нет оснований считать такую фразу отрицательной, то она скорее выражает положительное авторское отношение.

12. Синтаксическая единица, одна из частей которой — прилагательное «необходимый» или образованное от него слово, нейтральна, например:
    - Для завершения программ QE необходим сильный рынок труда, пишет РБК.
    - Для использования сервиса будет необходимо заменить действующую SIM-карту телефона на специальную SIM-карту с поддержкой технологии NFC.

    Это правило отражает, что оценка «необходим» используется, как правило, в формальных или информационных сообщениях, и автор, используя его вместо разговорного «нужен», как бы отстраняется от сказанного и показывает, что эта оценка объективна, и он не имеет какого-то своего мнения по теме.
13. Тональность синтаксической единицы основы предложения, подлежащее которой — глагол в начальной форме, а сказуемое — глагол «приходится», отрицательна, например:
    - Требования в ВУЗах мягкие и боевой опыт приходится приобретать в боевой же обстановке.
    - Действовать в правовом поле приходится методом проб и ошибок.
    - Я хочу ходить в клубы потанцевать, но в итоге приходится танцевать только или дома для себя, или в узкой компании друзей.

    В таких предложениях автор явно указывает на то, что совершение действия против воли, что выражает отрицательное авторское отношение.
14. Синтаксическая единица основы предложения, подлежащее которой — глагол в начальной форме, а сказуемое — предикативом «нельзя», нейтральна, например:
    - Пока нельзя предполагать, что США отключат GPS.
    - Нельзя исключать появление мощного исламистского фронта непосредственно у границ России.
    - Вторую накопительную башню, находящуюся на улице Советской, назвать нельзя действующей.

    В публицистическом стиле такие предложения чаще всего нейтральны, они сообщают, что автор не имеет чёткого мнения по вопросу, и не хочет настаивать на какой-либо точке зрения.
15. Тональность фраз, сообщающих о противостоянии, борьбе с чем-либо, положительна, если борьба происходит с явлением, которое описывается отрицательно, отрицательная, если с явлением, описывающимся положительно, иначе нейтральной, например:
    - Тренировка убирает навязчивые черты.
    - Сейчас уменьшается количество источников правдивой информации.
    - Это будет препятствовать производству героина.
    - Мы разрушили планы террористов по проведению новых атак.
    - Традиционно любимые согражданами книги в переплёте вытесняются более дешёвыми книжками в мягкой обложке.
    - Защитников флоры атаковали около 05:00 примерно полсотни неизвестных в масках, похожих на ультраправых футбольных фанатов.
    - Потому что стальной брони нет, а эти удары ослабляют защиту от самых страшных болезней.
    - Поправки должны избавить рынок от нелегальных букмекеров и защитить игроков от мошенников.
    - И Япония, и Россия хотят, чтобы эта проблема была решена.

    Правило применяется к синтаксической единице, удовлетворяющей одному из условий:
    - одна из частей — синтаксическая группа одного из глаголов «атаковать», «вытеснять», «ослаблять», «препятствовать», «разрушать», «решать», «убирать», «угрожать», «уменьшать», «не допускать» или образованного от него слова, а вторая — прямое дополнение,

- одна из частей — синтаксическая группа одного из глаголов «защищать», «избавлять», «расчищать», а вторая — косвенное дополнение с предлогом «от».
- одна из частей — синтаксическая группа глагола «бороться» или образованного от него слова, а вторая — косвенное дополнение с предлогом «с»,

16. Фразы, сообщающие о том, что какому-либо действию предшествовало отрицательное событие, нейтральны и не учитываются при дальнейшем выведении тональности, например:
    - За 17 лет, прошедших после трагедии распада СССР, обе страны много выиграли от взаимовыгодного сотрудничества.
    - Главный герой — консьерж месье Густав (Рэйф Файнс), который после смерти одной из постоялиц получает в наследство бесценную картину.

    Правило применяется к синтаксической единице, одна из частей которой — синтаксическая группа сказуемого или определения, выраженного причастием, а вторая — синтаксическая группа обстоятельства с предлогом «после».
17. Фразы, содержащие авторскую положительную или отрицательную оценку успешности действия, являются, соответственно, положительными или отрицательными вне зависимости от самого действия, например:
    - Бортовой лазер успешно уничтожил баллистическую ракету.
    - Неудачно выступил на главном турнире года Владимир Крамник.

    Данное правило было отражает, что само по себе описание действия может содержать как положительную, так и отрицательную лексику («уничтожил», «главный турнир»), но, если сам автор явно выносит действию ту или иную оценку, то она гораздо лучше отражает его авторскую позицию, и, следовательно, именно она должна использоваться для определения тональности всей фразы.
18. Фразы, содержащие простое сообщение о мнении третьего лица, без оценки этого мнения автором предложения, нейтральны, например:
    - По мнению аналитиков, сделка перспективна для обеих компаний.
    - По мнению многих немцев, сборник Liebe ist für alle da пропагандирует порнографию, насилие и незащищённый секс.
    - Мнение Юрия Любимова о том, что нужно делать, чтобы театр не превратился в музей самого себя, что должен уметь режиссёр и как правильно разговаривать с актёрами, читайте далее.
    - Министр отметил, что борьба с допингом сейчас крайне важна.
    - Теория гласит, что получившиеся гибридные кролики были особенно выносливыми и энергичными.
    - Интернет-издание Engadget сообщает, что энтузиасты уже опробовали новую операционную систему Google Chrome OS, сделав загрузочный диск на обычной флешке.
    - Н. Кан заверил, что перестановки в правительстве помогут укрепить лидерство партии и создать сильную команду, способную реформировать страну.

    Правило отражает то, что в классическом публицистическом тексте автор предложения с помощью конструкции «по мнению», «по словам», «считает», «сообщает», «заверяет», «гласит» и аналогичных явно указывает на то, что он лишь сообщает о том, что думает какой-то человек, не выражая своего согласия или несогласия с оценками этого человека. Кроме того, такая конструкция сама по себе подразумевает, что автор не может утверждать, что дела действительно обстоят так, как кто-то говорит, наоборот, это в первую очередь причина продолжить разбор темы и поговорить о ней ещё, прежде чем вынести свою оценку. Также использование подобных фраз может сообщать, что автор предложения не уверен, что дела обстоят именно так, как говорится, и сообщает не о самом факте, а то, что у кого-то есть мнение: например,

в предложении «По мнению руководства Северной Кореи, Сеул и Вашингтон сфабриковали обвинения против Пхеньяна» автор сообщает не о сфабрикованности обвинений, иначе тональность была бы отрицательной, а о том, что руководство КНДР считает обвинения сфабрикованными; автор не говорит, так это или нет на самом деле.

Однако сообщение о мнении самого автора может быть тональным, например:

- Отметим, что резкое обвинение Юлхяэ может стать поводом для того, что российским спортсменам в Ванкувере попросту не дадут спокойно выступать, замучив их бесконечными проверками на пробы.
- Важно отметить, что наше сотрудничество с Пакистаном помогло нам выследить бен Ладена и отследить здание, в котором он скрывался.
- Следует отметить, что на слухах и сплетнях вокруг предстоящего через три года конца света нажились тысячи мошенников по всему миру.
- Я знаю, что если выделить в одной большой задаче 10 маленьких подзадач, то после этого работа идёт как по маслу.

Алгоритм считает синтаксическую единицу изъяснительного придаточного нейтральной, если для неё выполняется одно из условий:

- она присоединяется к одному из глаголов «гласить», «заверять», «знать», «отмечать», «сообщать» или «считать», стоящем в форме третьего лица,
- она присоединяется к слову «мнение», в синтаксическую группу которого не входят притяжательные местоимения первого лица («моё», «наше»).

19. Предложения, построенные по газетному шаблону «подробнее о… читайте в… », всегда нейтральны, например:
    - Подробнее о развитии конфликта на Корейском полуострове читайте в статье Частного корреспондента «Затишье перед бурей».
    - Подробнее о претензиях к телеканалу «Дождь» читайте в статье «Не столько дождь лил, сколько гром гремел».
20. Тональность фраз, содержащих изъяснительное придаточное, придаточное причины или следствия, либо определительное придаточное, присоединённое с помощью конструкции «в том, что», определяется тональностью придаточного, вне зависимости от правила рекурсивного выведения.

    Изъяснительные придаточные:
    - Уже первые минуты игры показали, что эти опасения были напрасны.
    - Это новости, которые доказывают, что даже в бедных странах существуют условия для свободного выражения мнений.
    - Два месяца опыта показывают, что лучше этой платформы на данный момент нет.

    Придаточные причины:
    - Александр сказал, что приедет ещё, потому что ему очень понравилось.
    - Исторических потому, что мы всему миру ещё раз доказали демократичность нашего общества.

    Придаточные следствия:
    - А вот «Дозоры» Лукьяненко я не читал, поэтому первый фильм посмотрел без содроганий.
    - У нас в роду сплошь мальчики, поэтому Дашенька — это большое счастье и подарок, особенно для нас с дедушкой.
    - Число генералов сократилось, поэтому можно говорить о положительных тенденциях, привнесённых переаттестацией.

- Нет закона о социальном предпринимательстве, поэтому действовать в правовом поле приходится методом проб и ошибок.

Определительные придаточные, присоединённые с помощью конструкции «в том, что»:

- Дело в том, что в программе «Кинотавра» было много хорошего кино.
- Суть сервиса в том, что он позволит создавать программы людям, которые в программировании ничего не понимают.
- Особенность усилителей NAD состоит в том, что они обладают качеством звука, который обычно присущ аппаратам более высокой ценовой категории.

21. Синтаксическая единица вопросительного предложения с положительной основой нейтральна, например:
    - Довлатов — это классик или современник?
    - Как удаётся отвлечься, чтобы удачно нырнуть в океан и поймать рыбу?
    - На чём же вы выиграть хотите?

    Это правило отображает, что вопросительное предложение не может иметь положительную тональность, поскольку оно всегда содержит в себе долю сомнения, неуверенности в сказанном, и автор, даже используя в нём положительные оценки, всё-таки показывает, что они для него находятся под сомнением. Тем не менее, для отрицательных оценок это не так, и вопросительное предложение может иметь отрицательную тональность:
    - И почему мне не хватает терпения?
    - Как только можно сравнивать сталинизм с нацизмом?
22. Синтаксическая единица восклицательного предложения с нейтральной основой положительна, например:
    - Такого старта у нас ещё не было!
    - Оказалось, что один в поле воин!
    - Назовите кого-нибудь, кто обладает таким успехом!
    - Этим космонавтом был гражданин Советского Союза, майор Юрий Алексеевич Гагарин!

    Это правило отображает, что восклицательное предложение не может быть нейтральным, поскольку восклицательные предложения в языке служат для выражения авторской экспрессии, и, если автор его использовал, то у него точно есть своё мнение по теме, которое он хочет выразить. При этом, если основу предложения нет оснований считать положительной или отрицательной, то это, скорее всего, значит, что само событие, описываемое в основе, является для автора настолько важным, что он сообщает о нём в восклицательном предложении, что свидетельствует о положительном отношении автора.
23. Тональность синтаксической единицы, одна из частей которой — шаблонное вводное слово или вводная конструкция, выражающая отрицательное авторское отношение, такое как «увы» или «к сожалению», отрицательна вне зависимости от других встреченных средств проявления тональности, например:
    - К сожалению, местная избирательная комиссия утвердила заявление.
    - Увы, добившись успеха, группа чаще стала выступать на огромных стадионах.

    Данное правило основано на том, что вводные слова, как правило, служат для связки автором предложений и выражаемых в них мыслей в отдельный текст, поэтому, если автор явно использует вводное слово, показывающее, что он недоволен сообщаемой информацией, то именно это и выражает авторское отношение, даже если сообщаемую информацию можно было бы воспринимать положительно — например, саму по себе информацию о том, что музыкальная группа стала популярной и теперь выступает на стадионах, вполне можно рассматривать как положительную, но автор явно говорит, что он предпочёл бы, чтобы этого не происходило, и лично он относится к этому отрицательно.

Table 10. Sentiment classification performance with special syntactic rules for sentiment expression means processing

Таблица 10. Качество работы алгоритма со специальными правилами для обработки различных средств выражения тональности

Класс предложений	Точность	Полнота	F_1-мера	Количество предложений
Положительный	0.66	0.64	0.65	533
Нейтральный	0.84	0.81	0.83	2459
Отрицательный	0.77	0.82	0.80	1495
Среднее	0.76	0.76	0.76	4487
Взвешенное среднее	0.80	0.80	0.80	4487

Доля правильных ответов алгоритма (accuracy) = 0.80

Table 11. Sentiment classification confusion matrix with special syntactic rules for sentiment expression means processing

Таблица 11. Матрица ошибок алгоритма со специальными правилами для обработки различных средств выражения тональности

предсказ. / реальн.	Положит.	Нейтр.	Отрицат.	Всего
Положительная	342	156	35	533
Нейтральная	135	2000	324	2459
Отрицательная	38	225	1232	1495

Метрики качества и матрица ошибок алгоритма с правилами для обработки различных средств выражения тональности приведены в таблицах 10 и 11. Использование специальных правил привело к достаточно существенному росту качества определения тональности — средняя F_1-мера увеличилась на 0.09, до 0.76, а средняя взвешенная — на 0.07, до 0.80. Этот рост обусловлен в первую очередь значительно возросшими точностью и полнотой определения предложений с положительной тональностью — на 0.16 и 0.20 соответственно. За счёт введения правил, обрабатывающих различные средства выражения положительной тональности, алгоритм стал гораздо лучше отделять положительный класс предложений от нейтрального. Точность и полнота определения нейтральных предложений и предложений с отрицательной тональностью также увеличились как минимум на 3 %. Таким образом, добавление специальных правил обработки различных средств выражения тональности позволило достаточно сильно увеличить качество работы алгоритма.

3. Анализ ошибок и обсуждение результатов

Для определения наиболее перспективных путей дальнейшего развития алгоритма была собрана информация о 180 предложениях, тональность которых была определена неверно (по 30 предложений для каждого из 6 возможных сочетаний реальной и определённой алгоритмом тональности). Они были разделены на 11 групп в зависимости от причины неверной классификации (см. таблицу 12).

В группу с некорректной разметкой тональности попали предложения, которые, несмотря на все предосторожности при построении OpenSentimentCorpus, получили неверную разметку и при этом не попали в выборку для перепроверки. Часть из этих предложений сложны для понимания и должны были быть отмечены как сомнительные при разметке, например «Вниманием, прогулками по еловому ботаническому саду», однако некоторые просто были неверно поняты разметчиками, например предложение «По тому, что у меня в списке целых три Трифонова, вы уже поняли, кого я считаю главным советским писателем» было отмечено разметчиками как нейтральное, хотя оно имеет положительную тональность. Большинство из них были отмечены как нейтральные, что говорит о необходимости при создании корпусов в будущем уделить особое внимание разметке предложений этого класса.

В группу с некорректным построением дерева синтаксических единиц попали предложения, для которых либо некорректно отработал парсер дерева синтаксических связей, либо оказался некор-

Table 12. Distribution of errors of the proposed algorhtm in %

Таблица 12. Распределение ошибок предложенного алгоритма по группам в %

Реальная тональность	Положит.		Нейтральн.		Отрицат.		В среднем
Предсказанная тональность	Нейтр.	Отр.	Пол.	Отр.	Пол.	Нейтр.	
Некорректная разметка тональности предложения	0	10	33	20	6	6	13
Некорректное построение дерева синтаксических единиц	10	23	4	4	13	20	12
Некорректное определение тональности одиночного слова	17	4	13	7	13	20	12
Некорректное определение тональности устойчивого сочетания слов	3	10	4	3	23	17	10
Несовершенство существующих правил определения тональности	23	0	0	10	4	0	6
Отсутствие правила для обработки сообщения о чужом мнении	7	0	13	20	0	0	7
Отсутствие правила для обработки семантики противостояния, борьбы	3	23	0	3	10	3	7
Отсутствие правила для обработки семантики увеличения или уменьшения	3	0	0	3	4	10	3
Отсутствие правила для обработки придаточного предложения	17	7	3	0	7	3	6
Отсутствие правила для обработки иного средства выражения тональности	17	20	7	23	13	17	16
Тональность определяется высокоуровневой структурой предложения	0	3	23	7	7	3	7

ректен алгоритм построения деревьев синтаксических единиц. Общее количество таких предложений оказалось достаточно велико, но это, как показано в [15], неизбежно при использовании современных средств построения синтаксических деревьев.

В группы с некорректным определением тональности одиночного слова и тональности устойчивого сочетания слов попали:

- тональные предложения, тональность в которых выражается с помощью использования оценочной или эмотивной лексики, которая должна была присутствовать в тональном словаре, но из-за несовершенства словаря в него не попала, например, предложение с отрицательной тональностью «Молодая мать осталась одна», тональность в котором выражается с помощью устойчивого словосочетания «остаться одному», которое отсутствует в тональном словаре;
- нейтральные предложения с лексикой, которая была отмечена в тональном словаре как имеющая положительную или отрицательную тональность, например, нейтральное предложение «Наш герой сдёргивает с него простыню, а у того вместо тела — тело свиньи», в котором употребляется слово «герой», отмеченное в тональном словаре как имеющее положительную тональность, так как оно может использоваться для вынесения автором оценки, например, «Он — наш герой», но в анализируемом приложении нейтрально.

Эти две группы ошибок оказали достаточно существенное влияние на качество работы алгоритма, в сумме приведя к 22 % допущенных им ошибок. Нужно отметить значительное число отсутствующих в тональном словаре слов с положительной тональностью (17 % положительных предложений, ошибочно определённых как нейтральные), а также большое число сочетаний слов

с отрицательной тональностью, которые не вошли в тональный словарь (в среднем 20 % ошибок при определении отрицательной тональности).

Достаточно небольшое число предложений, ошибочно классифицированных алгоритмом из-за несовершенства существующих правил определения тональности, показывает, что введённые правила всё-таки оказались достаточно точными. Нужно отметить только большое количество (23 %) положительных предложений, ошибочно определённых как нейтральные. Вероятно, для отделения этих классов предложенные правила оказались слишком грубыми.

Правильному определению тональности многих предложений также помешало отсутствие правил для обработки некоторых средств выражения тональности, например: «Для автора красота есть признак здорового тела и гармонии со Вселенной» — автор сообщает о чужом мнении, но не показывает, согласен ли он с ним; «Динамическая медитация особенно эффективна для людей, страдающих от бессонницы» — автор сообщает о семантике борьбы медитации с бессонницей; «Отметим понижение организованности процесса голосования» — отрицательная тональность выражается с помощью оценки организованности как ухудшающейся; «Мы продолжим нашу трудную работу, чтобы сделать страну сильной и единой» — положительная тональность выражается с помощью придаточного цели. Главной проблемой создания правил для обработки таких средств оказывается их относительно редкое использование в языке, что затрудняет экспертам-лингвистам определение влияния на тональность того или иного средства выразительности. Тем не менее, эта часть ошибок представляет собой одно из важнейших направлений развития — суммарно на неё приходится 29 % случаев неправильного определения тональности. Особенно важна обработка сообщения для чужом мнении для более точного определения нейтральных предложений — из-за неё допущено в среднем 17 % ошибок для нейтральных предложений.

В группу предложений, тональность которых определяется высокоуровневой структурой, попали предложения, тональность которых не может быть определена на основе тональностей их составных частей. Например, нейтральное предложение «Людей не принуждают принимать радикальную идеологию, они сами приходят к ней в результате перемен, происходящих в обществе» состоит из двух частей: в первой сообщается, что людей не принуждают принимать радикальную идеологию, и эта часть само по себе имеет положительную тональность; во второй сообщается, что люди сами приходят к ней в результате происходящих перемен, и эта часть, если рассматривать её изолированно, также скорее положительна, однако если рассматривать всё предложение в целом, то оно нейтрально. Такие ошибки относятся к ограничениям алгоритма, и, несмотря на то, что их было допущено всего 7 % от общего количества, для нейтральных предложений, ошибочно определённых как положительные, эта проблема — одна из основных.

Подводя итоги анализа ошибок, можно сделать следующие выводы.

1. Некорректная разметка оказывает значительное влияние на качество определения тональности нейтральных предложений, для остальных двух классов она малозначима. Вероятнее всего, это связано с несовершенством руководств по разметке для нейтрального класса.
2. Некорректное построение дерева синтаксических единиц, наоборот, сильнее всего влияет на анализ тональных предложений, в особенности на отделение отрицательных предложений от двух других классов. Можно предположить, что причина этого в большей синтаксической сложности отрицательных предложений: в них чаще встречаются отрицания и противительные союзы, из-за чего эти предложения сложнее для автоматического синтаксического разбора.
3. Некорректное определение тональности одиночных слов сильнее всего влияет, во-первых, на обнаружение отрицательной тональности, во-вторых, на отделение положительных предложений от нейтральных. Некорректное определение тональности устойчивых сочетаний

слов влияет преимущественно на обнаружение отрицательной тональности. Вероятно, эта часть тонального словаря нуждается в серьёзном расширении.

4. Несовершенство существующих правил определения тональности серьёзно проявляется только в том, что многие положительные предложения определяются как нейтральные. Скорее всего, это вызвано тем, что при автоматическом подборе набора правил рекурсивного выведения положительных предложений оказалось слишком мало, чтобы сформировать для них надёжные правила.
5. Недостаточность правил для обработки сообщения о чужом мнении влияет преимущественно на класс нейтральных предложений, в результате многие нейтральные предложения ошибочно определяются как тональные.
6. Недостаточность правил для обработки семантики противостояния, борьбы влияет в первую очередь на то, что тональность многих положительных предложений определяется алгоритмом как отрицательная, а во вторую — на аналогичную ошибку для отрицательных предложений.
7. Отсутствие правил для обработки семантики увеличения или уменьшения практически не оказывает влияния на качество работы алгоритма.
8. Отсутствие правил для обработки придаточных предложений влияет, в первую очередь, на обнаружение средств выражения положительной тональности.

В целом наиболее серьёзные проблемы алгоритма — недостаточность набора слов с положительной тональностью и устойчивых сочетаний слов с отрицательной тональностью в тональном словаре, а также нехватка правил для обработки различных средств выражения положительной тональности.

Заключение

В статье рассмотрена разработка алгоритма определения тональности русскоязычных предложений, основанного на применении семантических правил и ориентированного на применение к текстами в публицистическом стиле. Алгоритм рекурсивно применяет подходящие правила к составным частям предложения, представленным в виде дерева синтаксических единиц. Большинство правил было построено на основе знаний эксперта-филолога относительно средств выражения тональности, известных русской лингвистике, и выбора тех из них, которые достаточно формализованы для того, чтобы их можно было алгоритмизировать с использованием генерируемых в рамках алгоритма деревьев синтаксических единиц. Также применялись такие инструменты, как дерево решений и тональный словарь.

В экспериментах с разработанным алгоритмом удалось достичь F_1-меры, равной 0.80, что является существенным шагом вперёд в анализе тональности предложений публицистического стиля. В более ранних работах, посвящённых данному вопросу (например, [17], в которой использовался SentiStrength) была получена F_1-мера, равная 0.60. Для английского языка существуют подходы, позволяющие добиться сравнимой с предложенным алгоритмом F_1-меры, в частности, 0.76 для новостных текстов [18] при использовании LSTM и 0.71 для предложений из LiveJournal [19] при использовании набора правил, построенного с помощью генетического программирования.

Проведённый анализ ошибок позволил идентифицировать основные проблемы алгоритма и на их основе сформулировать направления для его развития — совершенствование тонального словаря и введение новых семантических правил для обработки различных средств выражения положительной тональности.

References

[1] B. Liu, *Sentiment Analysis and Opinion Mining*. Springer, 2022, 167 pp.

[2] A. Dvoybikova, A. Karpov, and O. Verkholyak, "Analytical review of methods for identifying emotions in text data", in *3rd International Conference on R. Piotrowski's Readings in Language Engineering and Applied Linguistics, PRLEAL 2019*, St. Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences, 2020, pp. 8–21.

[3] S. Smetanin and M. Komarov, "Deep transfer learning baselines for sentiment analysis in Russian", *Information Processing & Management*, vol. 58, no. 3, p. 102 484, 2021.

[4] K. Nursakitov, A. Bekishev, S. Kumargazhanova, and A. Urkumbaeva, "Review of methods for determining the tonation of texts in natural languages", *Bulletin of Shakarim University. Technical Sciences*, no. 1 (9), pp. 59–67, 2023.

[5] M. S. Başarslan and F. Kayaalp, "Sentiment analysis on social media reviews datasets with deep learning approach", *Sakarya University Journal of Computer and Information Sciences*, vol. 4, no. 1, pp. 35–49, 2021.

[6] M. Wankhade, A. C. S. Rao, and C. Kulkarni, "A survey on sentiment analysis methods, applications, and challenges", *Artificial Intelligence Review*, vol. 55, no. 7, pp. 5731–5780, 2022.

[7] E. N. Tulupova and E. V. Golovina, "Lexico-stylistic percularities of tourist's Internet commentary", *Philology. Theory & Practice*, vol. 12, no. 5, pp. 257–261, 2019, in Russian.

[8] E. I. Boychuk, "Lexical and grammatical features of Internet reviews in the Russian and English languages", *Verhnevolzhski Philological Bulletin*, no. 3 (26), pp. 107–115, 2021, in Russian.

[9] A. Y. Poletaev and I. V. Paramonov, "Recursive sentiment detection algorithm for Russian sentences", *Automatic Control and Computer Sciences*, vol. 57, no. 7, pp. 740–749, 2023.

[10] M. Eremina, "Rechevoj zhanr otzyva v kommunikativnom prostranstve interneta", *Nauchnyj dialog*, no. 5 (53), pp. 34–45, 2016, in Russian.

[11] A. R. Kalashnikova, "Informativnaya tekstovaya tonal'nost' kak opredelyayushchij faktor ritmicheskoj tekstovoj organizacii", *Izvestiya Volgogradskogo Gosudarstvennogo Pedagogicheskogo Universiteta*, vol. 3 (107), pp. 113–116, 2016, in Russian.

[12] I. V. Paramonov and A. Y. Poletaev, "Annotation of text corpora by sentiment and presence of irony within a project of citizen science", *Modelirovanie i Analiz Informatsionnykh Sistem*, vol. 30, no. 1, pp. 86–100, 2023, in Russian.

[13] N. Loukachevitch and A. Levchik, "Creating a general Russian sentiment lexicon", in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, 2016, pp. 1171–1176.

[14] D. Kulagin, "Publicly available sentiment dictionary for the Russian language KartaSlovSent", in *Computational Linguistics and Intellectual Technologies: Proceesings of the Annual "Dialog" Conference (2021)*, in Russian, 2021, pp. 1106–1119.

[15] A. Y. Poletaev, I. V. Paramonov, and E. I. Boychuk, "Algorithm of constituency tree from depencency tree construction for a Russian-language sentence", *Informatics and Automation*, vol. 22, no. 6, pp. 1323–1353, 2023, in Russian.

[16] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*. Routledge, 2017, 368 pp.

[17] O. Koltsova, S. Alexeeva, S. Pashakhin, and S. Koltsov, "PolSentiLex: Sentiment detection in socio-political discussions on Russian social media", in *Conference on Artificial Intelligence and Natural Language*, 2020, pp. 1–16.

[18] W. Souma, I. Vodenska, and H. Aoyama, "Enhanced news sentiment analysis using deep learning methods", *Journal of Computational Social Science*, vol. 2, no. 1, pp. 33–46, 2019.

[19] A. B. Junior, N. F. F. da Silva, T. C. Rosa, and C. G. Junior, "Sentiment analysis with genetic programming", *Information Sciences*, vol. 562, pp. 116–135, 2021.

MODELING AND ANALYSIS OF INFORMATION SYSTEMS, VOL. 30, NO. 4, 2023
journal homepage: www.mais-journal.ru

ARTIFICIAL INTELLIGENCE

Keyphrase generation for the Russian-language scientific texts using mT5

A. V. Glazkova[1,3], D. A. Morozov[2,3], M. S. Vorobeva[1], A. A. Stupnikov[1]

DOI: 10.18255/1818-1015-2023-4-418-428

[1]University of Tyumen, 6 Volodarskogo str., Tyumen, 625003, Russia.
[2]Novosibirsk National Research State University, 1 Pirogova str., Novosibirsk, 630090, Russia.
[3]Institute for Information Transmission Problems (Kharkevich Institute), 19/1 Bol'shoj Karetnyj pereulok str., Moscow, 127051, Russia.

MSC2020: 68T50
Research article
Full text in Russian

Received November 13, 2023
After revision November 22, 2023
Accepted November 29, 2023

In this work, we applied the multilingual text-to-text transformer (mT5) to the task of keyphrase generation for Russian scientific texts using the Keyphrases CS&Math Russian corpus. The automatic selection of keyphrases is a relevant task of natural language processing since keyphrases help readers find the article easily and facilitate the systematization of scientific texts. In this paper, the task of keyphrase selection is considered as a text summarization task. The mT5 model was fine-tuned on the texts of abstracts of Russian research papers. We used abstracts as an input of the model and lists of keyphrases separated with commas as an output. The results of mT5 were compared with several baselines, including TopicRank, YAKE!, RuTermExtract, and KeyBERT. The results are reported in terms of the full-match F1-score, ROUGE-1, and BERTScore. The best results on the test set were obtained by mT5 and RuTermExtract. The highest F1-score is demonstrated by mT5 (11,24 %), exceeding RuTermExtract by 0,22 %. RuTermextract shows the highest score for ROUGE-1 (15,12 %). According to BERTScore, the best results were also obtained using these methods: mT5 — 76,89 % (BERTScore using mBERT), RuTermExtract — 75,8 % (BERTScore using ruSciBERT). Moreover, we evaluated the capability of mT5 for predicting the keyphrases that are absent in the source text. The important limitations of the proposed approach are the necessity of having a training sample for fine-tuning and probably limited suitability of the fine-tuned model in cross-domain settings. The advantages of keyphrase generation using pre-trained mT5 are the absence of the need for defining the number and length of keyphrases and normalizing produced keyphrases, which is important for flective languages, and the ability to generate keyphrases that are not presented in the text explicitly.

Keywords: automatic text summarization; selecting keyphrases; mT5

INFORMATION ABOUT THE AUTHORS

Anna V. Glazkova corresponding author	orcid.org/0000-0001-8409-6457. E-mail: a.v.glazkova@utmn.ru Candidate of Technical Sciences, Associate Professor, University of Tyumen; Senior Researcher, Institute for Information Transmission Problems (Kharkevich Institute).
Dmitry A. Morozov	orcid.org/0000-0003-4464-1355. E-mail: morozowdm@gmail.com Junior Researcher, Novosibirsk State University and Institute for Information Transmission Problems (Kharkevich Institute).
Marina S. Vorobeva	orcid.org/0000-0002-1508-4089. E-mail: m.s.vorobeva@utmn.ru Candidate of Technical Sciences, Head of the Department of Software, University of Tyumen, Docent.
Andrey A. Stupnikov	orcid.org/0000-0001-5201-1260. E-mail: a.a.stupnikov@utmn.ru Candidate of Physical and Mathematical Sciences, Associate Professor, University of Tyumen, Docent.

Funding: The work was carried out within the framework of project No. MK-3118.2022.4, supported by a grant from the President of the Russian Federation for young scientists — candidates of science.

For citation: A. V. Glazkova, D. A. Morozov, M. S. Vorobeva, and A. A. Stupnikov, "Keyphrase generation for the Russian-language scientific texts using mT5", *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 418-428, 2023.

МОДЕЛИРОВАНИЕ И АНАЛИЗ ИНФОРМАЦИОННЫХ СИСТЕМ, ТОМ 30, № 4, 2023
сайт журнала: www.mais-journal.ru

Генерация ключевых слов для русскоязычных научных текстов с помощью модели mT5

А. В. Глазкова[1,3], Д. А. Морозов[2,3], М. С. Воробьева[1], А. А. Ступников[1]

DOI: 10.18255/1818-1015-2023-4-418-428

[1]Тюменский государственный университет, ул. Володарского, д. 6, г. Тюмень, 625003, Россия.
[2]Новосибирский национальный исследовательский государственный университет, ул. Пирогова, д. 1, г. Новосибирск, 630090, Россия.
[3]Институт проблем передачи информации РАН им. А. А. Харкевича, Большой Каретный переулок, д. 19, стр. 1, г. Москва, 127051, Россия.

УДК 004.912
Научная статья
Полный текст на русском языке

Получена 13 ноября 2023 г.
После доработки 22 ноября 2023 г.
Принята к публикации 29 ноября 2023 г.

Авторами предлагается подход к генерации ключевых слов для русскоязычных научных текстов с помощью модели mT5 (multilingual text-to-text transformer), дообученнной на материале текстового корпуса Keyphrases CS&Math Russian. Автоматический подбор ключевых слов является актуальной задачей обработки естественного языка, поскольку ключевые слова помогают читателям осуществлять поиск статей и облегчают систематизацию научных текстов. В данной работе задача подбора ключевых слов рассматривается как задача автоматического реферирования текстов. Дообучение mT5 осуществлялась на текстах аннотаций русскоязычных научных статей. В качестве входных и выходных данных выступали тексты аннотаций и списки ключевых слов, разделенных запятыми, соответственно. Результаты, полученные с помощью mT5, были сравнены с результатами нескольких базовых методов: TopicRank, YAKE!, RuTermExtract, и KeyBERT. Для представления результатов использовались следующие метрики: F-мера, ROUGE-1, BERTScore. Лучшие результаты на тестовой выборке были получены с помощью mT5 и RuTermExtract. Наиболее высокое значение F-меры продемонстрировала модель mT5 (11.24 %), превзойдя RuTermExtract на 0.22 %. RuTermExtract показал лучший результат по метрике ROUGE-1 (15.12 %). Лучшие результаты по BERTScore также были достигнуты этими двумя методами: mT5 — 76.89 % (BERTScore, использующая модель mBERT), RuTermExtract — 75.8 % (BERTScore на основе ruSciBERT). Также авторами была оценена возможность mT5 генерировать ключевые слова, отсутствующие в исходном тексте. К ограничениям предложенного подхода относятся необходимость формирования обучающей выборки для дообучения модели и, вероятно, ограниченная применимость дообученной модели для текстов других предметных областей. Преимущества генерации ключевых слов с помощью mT5 — отсутствие необходимости задавать фиксированные значения длины и количества ключевых слов, необходимости проводить нормализацию, что особенно важно для флективных языков, и возможность генерировать ключевые слова, в явном виде отсутствующие в тексте.

Ключевые слова: автоматическое реферирование; подбор ключевых слов; mT5

ИНФОРМАЦИЯ ОБ АВТОРАХ

Анна Валерьевна Глазкова автор для корреспонденции	orcid.org/0000-0001-8409-6457. E-mail: a.v.glazkova@utmn.ru доцент каф. программного обеспечения ТюмГУ; с. н. с. ИППИ РАН, к. т. н.
Дмитрий Алексеевич Морозов	orcid.org/0000-0003-4464-1355. E-mail: morozowdm@gmail.com м. н. с. Лаборатории ПЩТ НГУ и Лаборатории КЛ ИППИ РАН.
Марина Сергеевна Воробьева	orcid.org/0000-0002-1508-4089. E-mail: m.s.vorobeva@utmn.ru доцент, заведующий каф. программного обеспечения ТюмГУ, к. т. н.
Андрей Анатольевич Ступников	orcid.org/0000-0001-5201-1260. E-mail: a.a.stupnikov@utmn.ru доцент каф. программного обеспечения ТюмГУ, к. ф.-м. н.

Финансирование: Работа выполнена в рамках проекта № МК-3118.2022.4, поддержанного грантом Президента Российской Федерации для молодых ученых — кандидатов наук.

Для цитирования: A. V. Glazkova, D. A. Morozov, M. S. Vorobeva, and A. A. Stupnikov, “Keyphrase generation for the Russian-language scientific texts using mT5”, *Modeling and analysis of information systems*, vol. 30, no. 4, pp. 418-428, 2023.

Введение

Ключевые слова являются важным элементом научного текста. Использование ключевых слов позволяет облегчить поиск статей, улучшить систематизацию научных текстов и резюмировать содержание статей для читателя. Автоматизация подбора ключевых слов представляет собой актуальную задачу в условиях большого количества информационных ресурсов. В настоящее время большинство методов подбора ключевых слов протестировано на англоязычных текстовых корпусах, в то время как для анализа русскоязычных текстов используется достаточно узкий набор методов выделения ключевых слов [1]. Существует несколько подходов к подбору ключевых слов: извлечение ключевых слов непосредственно из текста, подбор ключевых слов из заранее определенного перечня тематик или рубрик и генерация ключевых слов на основе семантики текста путем его обобщения и перефразирования [2, 3]. В последнем случае задача подбора ключевых слов схожа с задачей автоматического абстрактного реферирования текстов.

Большая часть широко используемых подходов к извлечению ключевых слов основана на выделении из текста наиболее значимых слов и словосочетаний по принципу обучения без учителя (unsupervised learning). К таким подходам относятся, в частности, статистические алгоритмы, такие как YAKE! [4] и KP-Miner [5], графовые (TopicRank [6], TextRank [7]) и ряд алгоритмов, основанных на применении методов машинного обучения и современных лингвистических моделей (KEA [8], KeyBERT [9]). Несмотря на впечатляющие результаты для ряда текстовых корпусов, алгоритмы, основанные на извлечении ключевых слов, обладают некоторыми ограничениями. В частности, они не способны определять количество ключевых слов автоматически и генерировать ключевые слова, отсутствующие в тексте в явном виде. На практике же списки ключевых слов обычно включают в себя как слова и словосочетания, встречающиеся в тексте непосредственно, так и ключевые слова, семантически связанные с содержанием текста, но не упомянутые в нем явно [10]. Данные ограничения могут быть преодолены при помощи нейросетевых моделей, в том числе современных лингвистических моделей для генерации текстов. В работах [11—13] описаны модели для генерации ключевых слов с рекуррентной архитектурой, позволяющие генерировать как присутствующие, так и отсутствующие в тексте ключевые слова. В работах [14, 15] предлагаются модели для генерации ключевых слов, основанные на архитектуре Transformer [16]. В статьях [17—20] представлены эксперименты по дообучению лингвистических моделей для генерации ключевых слов. Все перечисленные подходы протестированы на англоязычных корпусах.

К настоящему моменту опубликован ряд исследований, применяющих алгоритмы обучения без учителя к русскоязычным текстам [21—25]. В статье [26] предложен подход к извлечению ключевых слов с помощью комбинации данных алгоритмов и нейросетевых моделей. Подход протестирован на мультиязычном корпусе, включающем в себя русскоязычные тексты. Более подробное исследование нейросетевых моделей затруднено ввиду недостатка русскоязычных текстовых корпусов для подбора ключевых слов. Насколько нам известно, единственными корпусами текстов для данной задачи, находящимся в открытом доступе, являются корпус аннотаций научных статей [27], описанный в работе [25], и мультиязычный корпус новостных текстов, представленный в работе [26].

Данная работа направлена на преодоление пробела в использовании современных лингвистических моделей для генерации ключевых слов для русскоязычных научных текстов. В статье представлены результаты экспериментов по генерации списка ключевых слов как последовательности токенов на примере модели mT5 [28]. Выбор модели обусловлен ее широким использованием для автоматического реферирования и, в частности, для реферирования русскоязычных текстов (например, в работах [29, 30]). Дообучение mT5 выполнялось на русскоязычном корпусе для подбора ключевых слов аналогично тому, как выполняется дообучение для задачи автоматического реферирования. Проанализированы преимущества и недостатки данного подхода. Проведено сравнение

Table 1. The characteristics of the dataset **Таблица 1.** Характеристики набора данных

Характеристика	Обучающая выборка	Тестовая выборка
Количество текстов	5 844	2 504
Средняя длина текста (токенов)	102.97±86.05	62.41±39.14
Среднее количество ключевых слов	4.39	4.2
% ключевых слов, отсутствующих в исходном тексте	53.17	54.8

полученных результатов с результатами нескольких широко используемых подходов к извлечению ключевых слов.

Статья организована следующим образом. В разделе 1 представлены основные характеристики корпуса текстов, используемого в экспериментах. Раздел 2 содержит описание параметров используемой модели для генерации ключевых слов и перечень базовых методов ключевых слов, которые были использованы для сравнения. В разделе 3 приводятся и обсуждаются полученные результаты. Выводы к работе представлены в заключении.

1. Корпус текстов

В работе использован корпус аннотаций и соответствующих им списков ключевых слов, собранный из интернет-ресурсов «Киберленинка» и MathNet и описанный в работе [25]. «Киберленинка» комплектуется научными статьями различной тематики, в то время как публикации, размещенные на MathNet, включают в себя исследования по математике, автоматике и вычислительной технике, информатике и кибернетике [31]. Для исследования были отобраны русскоязычные аннотации, которым соответствуют не менее трех ключевых слов. Дубликаты текстов были удалены. Полученный датасет был разделен на обучающую и тестовую выборки в соотношении 70:30. Характеристики полученного набора данных представлены в таблице 1. Средняя длина текстов представляет собой среднее количество токенов, полученное с помощью токенизатора модели RuBERT-base-cased[1] [32]. Доля ключевых слов, отсутствующих в тексте, характеризует долю ключевых слов, которые не встречаются в исходном тексте (тексте аннотации) в явном виде. При подсчете данной характеристики текст и ключевые слова подвергались предварительной нормализации. В обучающей и тестовой выборках содержится значительная доля ключевых слов, отсутствующих в исходном тексте (53.17 % и 54.8 % соответственно). Таким образом, списки ключевых слов часто содержат слова и словосочетания, семантически связанные с содержанием аннотации, но непосредственно не присутствующие в тексте (например, обобщающие ключевые слова, описывающие предметную область, синонимы и гиперонимы слов, встречающихся в тексте и так далее).

2. Методы подбора ключевых слов

Для генерации ключевых слов была использована модель mT5 (multilingual text-to-text transformer) [28], предварительно обученная на текстах на 101 языке. Архитектура mT5 в целом повторяет архитектуру модели T5 [33], разработанную ранее для английского языка, и использует подход Text-to-Text. Суть данного подхода заключается в том, что данные для каждой задачи, на которой проходило предварительное обучение, были представлены в текстовом виде. В данной работе была использована модель mT5-base[2], имеющая 580 млн параметров.

Дообучение mT5 для задачи генерации ключевых слов выполнялось на обучающей выборке в течение 10 эпох с использованием следующих параметров: максимальная длина входной последовательности — 256 токенов, скорость обучения — $4 \cdot 10^{-5}$, размер батча — 8. Параметр repetition penalty, влияющий на размер ошибки модели в случае генерации повторяющихся токенов, варьировался в диапазоне [1;2] с шагом 0.1. В результате экспериментов было выбрано значение repetition

[1] https://huggingface.co/DeepPavlov/rubert-base-cased
[2] https://huggingface.co/google/mt5-base

penalty, равное 1.4. В модель подавались тексты аннотаций в качестве исходных текстов и списки ключевых слов, разделенных запятыми, в качестве выходных. Таким образом, дообучение выполнялось аналогично дообучению для задачи автоматического реферирования, однако в роли рефератов для исходных текстов выступали списки соответствующих им ключевых слов. Тестирование модели выполнялось на тестовой выборке.

Результаты модели mT5 были сравнены с результатами нескольких базовых методов извлечения ключевых слов:

1. TopicRank [6], графовый алгоритм, в ходе работы которого все последовательности идущих подряд прилагательных и существительных в тексте обозначаются как потенциальные ключевые фразы, а затем объединяются в семантические группы с использованием алгомеративной иерархической кластеризации. Среди получившихся кластеров выбираются наиболее значимые при помощи алгоритма PageRank [34], лежащего в основе поисковой системы Google.
2. YAKE! (Yet Another Keyword Extractor) [4], статистический алгоритм, основанный на применении ряда вычислимых эвристик: вероятности написания слова с заглавной буквы, наиболее вероятной позиции слова в тексте, доли предложений, содержащих слов и т. д.
3. RuTermExtract[3], алгоритм, основанный на анализе морфологических характеристик слов и словосочетаний и набора правил для извлечения ключевых слов. Алгоритм представляет собой адаптированную для русского языка версию алгоритма TermExtract[4]. Для морфологического анализа в русскоязычной версии используется библиотека PyMorphy2 [35].
4. KeyBERT [9], нейросетевой алгоритм на основе Bidirectional Encoder Representations from Transformers (BERT) [36], выбирающий из текста слова и словосочетания, семантические вектора которых наиболее схожи с векторным представлением всего текста.

В данной работе была использована реализация алгоритмов TopicRank и YAKE! из библиотеки PKE [37] с подключением spacy[5]-модели ru_core_news_sm. В качестве основы для алгоритма KeyBERT использовалась модель ruSciBERT[6] [38]. С помощью каждого из базовых алгоритмов были извлечены по 5, 10 и 15 ключевых слов. Для TopicRank и YAKE! также варьировалась максимальная длина n-грамм ($N = 1$ и $N = 3$, N — максимальная длина ключевого слова). Тестирование базовых алгоритмов также выполнялось на тестовой выборке.

3. Результаты

Для представления результатов были использованы четыре метрики: F-мера, ROUGE-1 [39] и два варианта BERTScore [40]. F-мера рассчитывается на основании показателей точности (precision) и полноты (recall) двух списков ключевых слов: представленного в корпусе текстов и полученного машинным способом. Ключевые слова при этом предварительно нормализуются с помощью библиотеки PyMorphy2 [35]. ROUGE-1 отражает количество совпадающих униграмм в исходном и сгенерированном списках ключевых слов. Метрика BERTScore показывает семантическое сходство списков ключевых слов, оценивая степень близости векторных представлений входящих в них токенов. Векторные представления токенов могут быть получены из предварительно обученных лингвистических моделей. Метрика BERTScore рассчитана в данной работе двумя способами. В первом случае используется многоязычная модель BERT (mBERT) [36], так как данная модель применялась для оценки близости неанглоязычных текстов в оригинальной статье [40]. Во втором варианте близость списков ключевых слов оценивается с помощью ruSciBERT [38], обученной на русскоязычных научных текстах.

[3] https://github.com/igor-shevchenko/rutermextract
[4] https://pypi.org/project/topia.termextract
[5] https://spacy.io/usage/models
[6] https://huggingface.co/ai-forever/ruSciBERT

Table 2. Results, %

Таблица 2. Результаты, %

Метод	Все ключевые слова				Ключевые слова, отсутствующие в тексте	
	F	R	BS	$BS(S)$	BS	$BS(S)$
TopicRank ($N = 1$, $k = 5$)	3.86	7.62	71.65	71.09	68.81	66.26
TopicRank ($N = 1$, $k = 10$)	3.84	8.08	70.89	72.03	66.75	66.29
TopicRank ($N = 1$, $k = 15$)	3.72	7.99	70.49	71.92	65.82	65.94
TopicRank ($N = 3$, $k = 5$)	4.79	6.30	73.48	70.6	69.47	65.88
TopicRank ($N = 3$, $k = 10$)	4.96	6.64	73.52	71.24	68.81	65.85
TopicRank ($N = 3$, $k = 15$)	4.92	6.66	73.48	71.27	68.74	65.77
YAKE! ($N = 1$, $k = 5$)	3.75	7.25	71.47	71.04	68.73	66.29
YAKE! ($N = 1$, $k = 10$)	3.92	8.38	70.87	72.30	66.72	66.54
YAKE! ($N = 1$, $k = 15$)	3.79	8.09	70.45	72.19	65.75	66.13
YAKE! ($N = 3$, $k = 5$)	2.82	5.17	69.30	67.10	66.15	63.09
YAKE! ($N = 3$, $k = 10$)	5.37	6.41	68.27	66.64	64.46	61.50
YAKE! ($N = 3$, $k = 15$)	6.39	6.47	67.01	65.15	62.81	59.64
RuTermExtract ($k = 5$)	9.75	14.42	75.85	74.98	70.73	68.31
RuTermExtract ($k = 10$)	11.02	**15.12**	75.95	**75.80**	70.25	68.16
RuTermExtract ($k = 15$)	10.86	14.87	75.86	75.71	70.04	67.84
KeyBERT ($k = 5$)	5.27	4.24	70.63	68.01	67.08	63.70
KeyBERT ($k = 10$)	6.43	5.31	70.00	67.85	65.70	62.74
KeyBERT ($k = 15$)	6.53	5.65	69.36	66.61	64.85	61.16
mT5	**11.24**	13.10	**76.89**	75.06	**72.85**	**68.95**

Результаты сравнения методов подбора ключевых слов представлены в таблице 2. Для каждого метода были рассчитаны метрики для всех ключевых слов из списка ключевых слов, представленного в корпусе («Все ключевые слова»), а также только для тех ключевых слов из списка, представленного в корпусе, которые не встречаются в исходном тексте в явном виде («Ключевые слова, отсутствующие в тексте»). Во втором случае сравнивался список ключевых слов, представленный в корпусе, за исключением ключевых слов, встречающихся в тексте, и список ключевых слов, полученный моделью. Поскольку методы, выполняющие извлечение ключевых слов из исходного текста, не способны генерировать ключевые слова, отсутствующие в тексте, во втором случае оценка с помощью F-меры и ROUGE не проводилась. Результаты были оценены с позиции семантического сходства (BERTScore). В таблице 2 используются следующие сокращения: F — F-мера, R — ROUGE-1, BS — BERTScore, основанная на mBERT, $BS(S)$ — BERTScore, основанная на ruSciBERT, N — максимальная длина ключевого слова (длина n-граммы), k — количество ключевых слов. Лучший результат в каждом столбце подчеркнут и выделен полужирным шрифтом.

На рисунке 1 показана зависимость результатов модели mT5 от величины параметра repetition penalty, влияющего на размер ошибки модели в случае генерации повторяющихся токенов. График построен на основе результатов моделей на полной тестовой выборке. Лучший результат для каждой метрики выделен пятиугольным маркером. Поскольку для трех метрик из четырех лучшие результаты были получены при значении repetition penalty, равном 1.4, в таблице 2 указаны результаты этой модели.

Результаты, полученные с помощью RuTermExtract и mT5, в целом выше, чем результаты TopicRank, YAKE! и KeyBERT. При этом наиболее высокий показатель по F-мере демонстрирует mT5 (11.24 %), превосходя RuTermExtract на 0.22 %. RuTermExtract показывает наиболее высокие по-

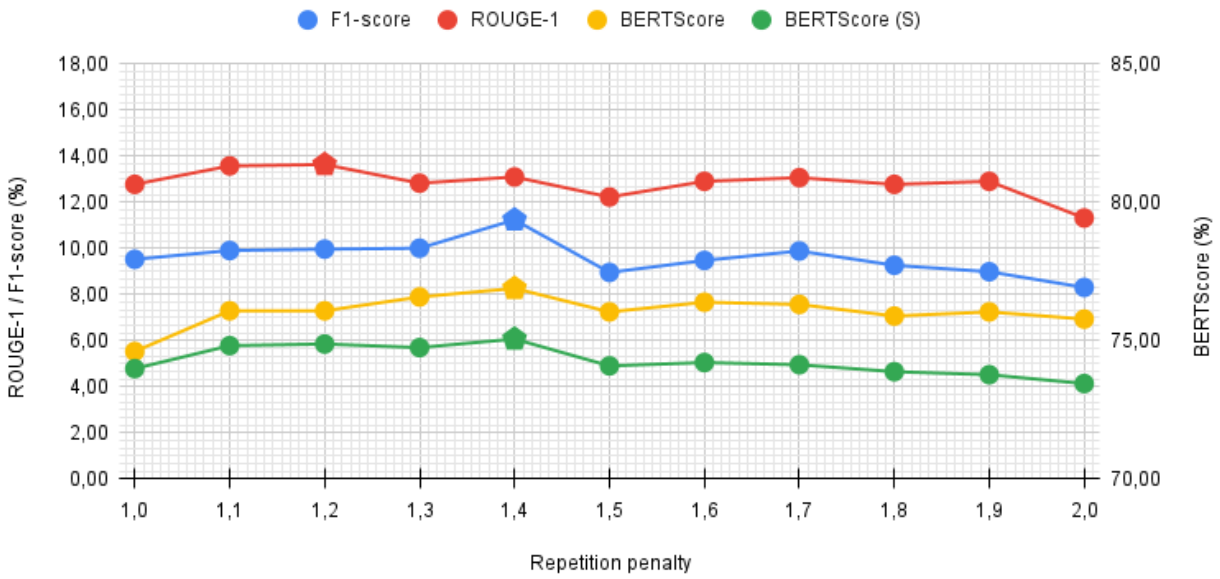


Fig. 1. The influence of the repetition penalty parameter on keyphrase generation performance

Рис. 1. Влияние параметра repetition penalty на качество генерации ключевых слов

казатели по метрике ROUGE-1 (15.12 %). По BERTScore лучшие результаты также получены с помощью данных методов: mT5 — 76.89 % (BS), RuTermExtract — 75.8 % ($BS(S)$). mT5 демонстрирует лучшие результаты при генерации ключевых слов, отсутствующих в исходном тексте. Генерация таких ключевых слов является технически более сложной задачей, чем извлечение ключевых слов, поскольку требует от модели «понимания» семантики текста и способности к реферированию и обобщению. Мы дополнительно рассчитали показатели F-меры и полноты (recall) для генерации ключевых слов, отсутствующих в тексте, и получили показатели 1.33 % и 1.7 % соответственно.

В таблице 3 представлены примеры ключевых слов, полученных с помощью разных алгоритмов. Для удобства сравнения приведены только результаты при $k = 5$. Приведенные примеры показывают, что ключевые слова, сгенерированные с помощью mT5, носят ожидаемо более общий характер, чем ключевые слова, выделенные другими методами. Количество ключевых слов, сгенерированных mT5, невелико и в целом близко среднему количеству ключевых слов для текста в используемом корпусе. В приведенных примерах mT5 генерирует грамматически корректные словосочетания и не требует проведения дополнительной нормализации, что является преимуществом данного подхода.

Заключение

В данной работе задача подбора ключевых слов рассматривается как задача автоматического реферирования текста на естественном языке. Мы описываем результаты экспериментов по генерации списка ключевых слов в виде одной строки для аннотаций научных текстов на русском языке с помощью предобученной лингвистической модели mT5. Результаты сравниваются с результатами ряда широко используемых методов извлечения ключевых слов.

Среди преимуществ генерации ключевых слов с помощью предобученной лингвистической модели можно назвать отсутствие необходимости проводить нормализацию и задавать ограничения на количество и длину ключевых слов, возможность генерировать ключевые слова, которые не упомянуты в исходном тексте в явном виде. С другой стороны, указанные свойства могут быть также ограничениями указанного подхода. Дообучение рассмотренной модели требует наличия обуча-

Table 3. The examples of keyphrases **Таблица 3.** Примеры ключевых слов

Метод	Ключевые слова
Пример 1. Рассмотрено современное состояние разработок газовых сенсоров на основе оксидов металлов, как наиболее перспективных. Для исследования сенсоров разработана информационная система, обладающая широкими возможностями по обеспечению их эффективного функционирования, а также передачи информации с использованием сетевых технологий.	
Список ключевых слов, представленный в корпусе	газовый анализ, информационная система, электронный нос, нейронные сети
TopicRank ($N = 1$, $k = 5$)	сенсор, возможность, система, обеспечение, исследование
TopicRank ($N = 3$, $k = 5$)	обеспечению, широкими возможностями, информационная система, исследования сенсоров, эффективного функционирования
YAKE! ($N = 1$, $k = 5$)	состояние, технология, сенсор, разработка, основа
YAKE! ($N = 3$, $k = 5$)	рассмотрено современное состояние, современное состояние разработок, состояние разработок газовых, основе оксидов металлов, разработок газовых сенсоров
RuTermExtract ($k = 5$)	современное состояние разработок, основа оксидов металлов, обладающая широкие возможности, эффективное функционирование, сетевые технологии
KeyBERT ($k = 5$)	рассмотрено современное, разработана информационная, система обладающая, эффективного функционирования, современное состояние
mT5	оксиды металлов, информационная система, обработка информации
Пример 2. Впервые выявляются общие и особенные черты методики аннотирования машиночитаемых документов. Проанализирована практика аннотирования электронных документов локального доступа и ресурсов удаленного доступа. Выделены источники сведений об электронных документах, их идентификационные признаки, значимые для методики аннотирования. Названы причины, которые осложняют процесс аннотирования электронных информационных ресурсов удаленного доступа. Показаны пути создания работоспособных методик аннотирования машиночитаемых документов.	
Список ключевых слов, представленный в корпусе	аннотирование, электронные документы, аспектная схема
TopicRank ($N = 1$, $k = 5$)	аннотирование, документ, доступ, методика, ресурс
TopicRank ($N = 3$, $k = 5$)	доступа, значимые, идентификационные признаки, методики аннотирования, электронных документах
YAKE! ($N = 1$, $k = 5$)	аннотирование, документ, методика, доступ, ресурс
YAKE! ($N = 3$, $k = 5$)	впервые выявляются общие, особенные черты методики, впервые выявляются, выявляются общие, особенные черты
RuTermExtract ($k = 5$)	электронные документы, удалённый доступ, машиночитаемые документы, особенные черты методики аннотирования, электронные информационные ресурсы
KeyBERT ($k = 5$)	аннотирования машиночитаемых, машиночитаемых документов, методик аннотирования, методики аннотирования, машиночитаемых
mT5	машиночитаемый документ, удаленный доступ, аннотирование, методы аннотирования

ющей выборки и, вероятно, дообученная модель ограниченно пригодна для генерации ключевых слов для текстов других предметных областей. Кроме того, эффективность предложенного подхода и значения метрик зависят от специфики корпуса текстов, используемого для экспериментов. В рассмотренном корпусе доля ключевых слов, не встречающих в тексте в явном виде, составляет 53.17 % и 54.8 % для обучающей и тестовой выборок соответственно. Поскольку подходы, осуществляющие извлечение, а не генерацию ключевых слов, не способны генерировать ключевые слова данного типа, модели генерации текста, подобные mT5, имеют преимущество на таких корпусах.

В дальнейшей работе будут исследованы другие лингвистические модели и тексты, относящиеся к различным предметным областям (например, новостные). Также будет изучена возможность генерации заданного пользователем количества ключевых слов и введения других ограничений на параметры списка ключевых слов, генерируемого моделью. Отдельным направлением для дальнейших исследований является генерация ключевых слов, в явном виде отсутствующих в исходном тексте. Вероятно, полученные в работе показатели F-меры и полноты при генерации ключевых слов, отсутствующих в тексте, были бы выше, если бы дообучение модели проводилось не на списках всех ключевых слов, а только на ключевых словах, отсутствующих в тексте. Также может быть проведена типизация таких ключевых слов (синонимы, гиперонимы и так далее), и для генерации ключевых слов каждого типа может быть проведена отдельная оценка эффективности подходов.

References

[1] N. S. Lagutina, K. V. Lagutina, A. S. Adrianov, and I. V. Paramonov, “Russian language thesauri: Automated construction and application for natural language processing tasks”, *Modeling and Analysis of Information Systems*, vol. 25, no. 4, pp. 435–458, 2018, in Russian.

[2] S. Beliga, *Keyword extraction: A review of methods and approaches*, https://api.semanticscholar.org/CorpusID:6834431, 2014.

[3] E. Çano and O. Bojar, “Keyphrase generation: A multi-aspect survey”, in *25th Conference of Open Innovations Association (FRUCT)*, IEEE, 2019, pp. 85–94.

[4] R. Campos, V. Mangaravite, A. Pasquali, A. Jorge, C. Nunes, and A. Jatowt, “YAKE! keyword extraction from single documents using multiple local features”, *Information Sciences*, vol. 509, pp. 257–289, 2020.

[5] S. R. El-Beltagy and A. Rafea, “KP-Miner: A keyphrase extraction system for English and Arabic documents”, *Information systems*, vol. 34, no. 1, pp. 132–144, 2009.

[6] A. Bougouin, F. Boudin, and B. Daille, “TopicRank: Graph-based topic ranking for keyphrase extraction”, in *International joint conference on natural language processing (IJCNLP)*, 2013, pp. 543–551.

[7] R. Mihalcea and P. Tarau, “TextRank: Bringing order into text”, in *Proceedings of the 2004 conference on empirical methods in natural language processing*, 2004, pp. 404–411.

[8] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, “KEA: Practical automatic keyphrase extraction”, in *Proceedings of the fourth ACM conference on Digital libraries*, 1999, pp. 254–255.

[9] M. Grootendorst, *KeyBERT: Minimal keyword extraction with BERT*, version v0.3.0, 2020. DOI: 10.5281/zenodo.4461265. [Online]. Available: https://github.com/MaartenGr/KeyBERT.

[10] F. Boudin and Y. Gallina, “Redefining absent keyphrases and their effect on retrieval effectiveness”, in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Association for Computational Linguistics, 2021, pp. 4185–4193.

[11] R. Meng, S. Zhao, S. Han, D. He, P. Brusilovsky, and Y. Chi, "Deep keyphrase generation", in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 582–592.

[12] E. Cano and O. Bojar, "Keyphrase generation: A text summarization struggle", in *Proceedings of NAACL-HLT*, 2019, pp. 666–672.

[13] J. Zhao and Y. Zhang, "Incorporating linguistic constraints into keyphrase generation", in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019, pp. 5224–5233.

[14] R. Liu, Z. Lin, and W. Wang, *Keyphrase prediction with pre-trained language model*, 2020. arXiv: 2004.10462 [cs.CL].

[15] M. Kulkarni, D. Mahata, R. Arora, and R. Bhowmik, "Learning rich representation of keyphrases from text", in *Findings of the Association for Computational Linguistics: NAACL 2022*, 2022, pp. 891–906.

[16] A. Vaswani *et al.*, "Attention is all you need", in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 2017, pp. 6000–6010.

[17] M. F. M. Chowdhury, G. Rossiello, M. Glass, N. Mihindukulasooriya, and A. Gliozzo, *Applying a generic sequence-to-sequence model for simple and effective keyphrase generation*, 2022. arXiv: 2201.05302 [cs.CL].

[18] A. Glazkova and D. Morozov, "Applying transformer-based text summarization for keyphrase generation", *Lobachevskii Journal of Mathematics*, vol. 44, no. 1, pp. 123–136, 2023.

[19] A. Glazkova and D. Morozov, "Multi-task fine-tuning for generating keyphrases in a scientific domain", in *IX International Conference on Information Technology and Nanotechnology (ITNT)*, IEEE, 2023, pp. 1–5.

[20] D. Wu, W. U. Ahmad, and K.-W. Chang, *Pre-trained language models for keyphrase generation: A thorough empirical study*, 2022. arXiv: 2212.10233 [cs.CL].

[21] E. G. Sokolova and O. Mitrofanova, "Automatic keyphrase extraction by applying KEA to Russian texts", in *Computational linguistics and computing ontologies*, in Russian, 2017, pp. 157–165.

[22] M. V. Sandul and E. G. Mikhailova, "Keyword extraction from single Russian document", in *Proceedings of the Third Conference on Software Engineering and Information Management*, 2018, pp. 30–36.

[23] E. Sokolova, A. Moskvina, and O. Mitrofanova, "Keyphrase extraction from the Russian corpus on linguistics by means of KEA and RAKE algorithms", in *Data analytics and management in data-intensive domains*, 2018, pp. 369–372.

[24] O. A. Mitrofanova and D. A. Gavrilic, "Experiments on automatic keyphrase extraction in stylistically heterogeneous corpus of Russian texts", *Terra Linguistica*, vol. 50, no. 4, pp. 22–40, 2022, in Russian.

[25] D. Morozov, A. Glazkova, M. Tyutyulnikov, and B. Iomdin, "Keyphrase generation for abstracts of the Russian-language scientific articles", *NSU Vestnik. Series: Linguistics and Intercultural Communication*, vol. 21, no. 1, pp. 54–66, 2023, in Russian.

[26] B. Koloski, S. Pollak, B. Škrlj, and M. Martinc, "Extending neural keyword extraction with TF-IDF tagset matching", in *Proceedings of the EACL Hackashop on News Media Content Analysis and Automated Report Generation*, 2021, pp. 22–29.

[27] D. Morozov and A. Glazkova, *Keyphrases CS&Math Russian*, version v1, 2022. DOI: 10.17632/dv3j9wc59v.1. [Online]. Available: https://data.mendeley.com/datasets/dv3j9wc59v/1.

[28] L. Xue *et al.*, "mT5: A massively multilingual pre-trained text-to-text transformer", in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021, pp. 483–498.

[29] K. Grashchenkov, A. Grabovoy, and I. Khabutdinov, "A method of multilingual summarization for scientific documents", in *Ivannikov Ispras Open Conference (ISPRAS)*, IEEE, 2022, pp. 24–30.

[30] A. Gryaznov, R. Rybka, I. Moloshnikov, A. Selivanov, and A. Sboev, "Influence of the duration of training a deep neural network model on the quality of text summarization task", *AIP Conference Proceedings*, vol. 2849, no. 1, p. 400 006, 2023.

[31] A. A. Pechnikov, "Comparative analysis of scientometrics indicators of journals Math-Net.ru and Elibrary.ru", *Vestnik Tomskogo gosudarstvennogo universiteta*, no. 56, pp. 112–121, 2021, in Russian.

[32] Y. Kuratov and M. Arkhipov, "Adaptation of deep bidirectional multilingual transformers for Russian language", in *Komp'juternaja Lingvistika i Intellektual'nye Tehnologii*, in Russian, 2019, pp. 333–339.

[33] C. Raffel *et al.*, "Exploring the limits of transfer learning with a unified text-to-text transformer", *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 5485–5551, 2020.

[34] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank citation ranking: Bringing order to the web: Stanford InfoLab", 1999, p. 1 508 503.

[35] M. Korobov, "Morphological analyzer and generator for Russian and Ukrainian languages", in *Analysis of Images, Social Networks and Texts: 4th International Conference, AIST 2015, Yekaterinburg, Russia, April 9–11, 2015, Revised Selected Papers 4*, Springer, 2015, pp. 320–332.

[36] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding", in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 4171–4186.

[37] F. Boudin, "PKE: An open source python-based keyphrase extraction toolkit", in *Proceedings of COLING 2016, the 26th international conference on computational linguistics: system demonstrations*, 2016, pp. 69–73.

[38] N. A. Gerasimenko, A. S. Chernyavsky, and M. Nikiforova, "ruSciBERT: A transformer language model for obtaining semantic embeddings of scientific texts in Russian", in *Doklady Mathematics*, Springer, vol. 106, 2022, S95–S96.

[39] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries", in *Text summarization branches out*, 2004, pp. 74–81.

[40] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi, *BERTScore: Evaluating text generation with BERT*, 2020. arXiv: 1904.09675 [cs.CL].