

www.aurora-group.eu
www.nbpublish.com

ПРОГРАММНЫЕ СИСТЕМЫ

И

ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ

научный журнал



Выходные данные

Номер подписан в печать: 05-01-2025

Учредитель: Даниленко Василий Иванович, w.danilenko@nbpublish.com

Издатель: ООО <НБ-Медиа>

Главный редактор: Морозов Михаил Николаевич, кандидат технических наук,
mikhail.n.morozov@gmail.com

ISSN: 2454-0714

Контактная информация:

Выпускающий редактор - Зубкова Светлана Вадимовна

E-mail: info@nbpublish.com

тел.+7 (966) 020-34-36

Почтовый адрес редакции: 115114, г. Москва, Павелецкая набережная, дом 6А, офис 211.

Библиотека журнала по адресу: http://www.nbpublish.com/library_tariffs.php

Publisher's imprint

Number of signed prints: 05-01-2025

Founder: Danilenko Vasiliy Ivanovich, w.danilenko@nbpublish.com

Publisher: NB-Media ltd

Main editor: Morozov Mikhail Nikolaevich, kandidat tekhnicheskikh nauk,
mikhail.n.morozov@gmail.com

ISSN: 2454-0714

Contact:

Managing Editor - Zubkova Svetlana Vadimovna

E-mail: info@nbpublish.com

тел.+7 (966) 020-34-36

Address of the editorial board : 115114, Moscow, Paveletskaya nab., 6A, office 211 .

Library Journal at : http://en.nbpublish.com/library_tariffs.php

Редакционный совет

Гельман Виктор Яковлевич – доктор технических наук, профессор, профессор кафедры медицинской информатики и физики ФГБОУ ВО «Северо-Западный государственный медицинский университет им. И.И.Мечникова», 191015, Россия, г. Санкт-Петербург, ул. Кирочная, д.41, gelm@sg2104.spb.edu

Поляков Виктор Павлович – доктор педагогических наук, профессор, Главный научный сотрудник лаборатории психолого-педагогического и учебно-методического обеспечения развития информатизации образования Центра информатизации образования Федерального государственного бюджетного научного учреждения «Институт управления образованием Российской академии образования», 105062, г. Москва, ул. Макаренко, д. 5/16, стр. 1Б, polvikpal@mail.ru

Гармаев Баир Заятуевич – кандидат физико-математических наук, научный сотрудник, Институт физического материаловедения Сибирского Отделения РАН, 670000, Россия, республика Бурятия, г. Улан-Удэ, ул. Сахьяновой, 6, каб. 313

Клименко Анна Борисовна – кандидат технических наук, научный сотрудник Научно-исследовательского института многопроцессорных вычислительных систем имени академика А.В. Каляева Южного федерального университета (НИИ МВС ЮФУ), 347935, Россия, Ростовская область, г. Таганрог, ул. 8 Переулок, 15

Лютикова Лариса Адольфовна – кандидат физико-математических наук, заведующая отделом Нейроинформатики и машинного обучения, Институт прикладной математики и автоматизации Кабардино-Балкарского научного центра РАН – филиал Кабардино-Балкарского научного центра РАН (ИПМА КБНЦ РАН), 360000, Россия, Республика Кабардино-Балкария, г. Нальчик, ул. Шортанова, 89а

Мустафаев Арслан Гасанович – доктор технических наук, профессор, Государственное автономное образовательное учреждение высшего образования "Дагестанский государственный университет народного хозяйства", кафедра «Информационные технологии и информационная безопасность», 367015, Россия, Республика Дагестан, г. Махачкала, ул. Атаева, 5, каб. 4.5

Шестаков Александр Валентинович – кандидат технических наук, доцент Южный Федеральный университет, кафедра вычислительной техники, 347902, Россия, Ростовская область, г. Таганрог, ул. Свободы, 24/2

Сидоркина Ирина Геннадьевна – доктор технических наук, профессор, декан факультета Информатики и вычислительной техники Поволжского государственного технологического университета, Йошкар-Ола, Россия E-mail: dekan_fivt@mail.ru

Екатерина Прасолова-Førland - PhD, Норвежский университет науки и технологии (NTNU), Трондхейм, Норвегия E-mail: Ekaterina.Prasolova-Forland@idi.ntnu.no

Голенков Владимир Васильевич - доктор технических наук, профессор, заведующий кафедрой Интеллектуальных информационных технологий Белорусского государственного университета информатики и радиоэлектроники, г. Минск, Республика Беларусь E-mail: golen@bsuir.by

Домошницкий Александр Исаакович - кандидат физико-математических наук, декан естественно-научного факультета Университетского центра в г.Ариэль, Израиль, Самариа E-mail: adom@ariel.ac.il Department of Mathematics and Computer Sciences, The Ariel

University Center of Samaria, 44837 Ariel, ISRAEL

Коробейников Анатолий Григорьевич - доктор технических наук, профессор «Институт земного магнетизма, ионосферы и распространения радиоволн РАН (ИЗМИРАН)», Санкт-Петербургский филиал E-mail: korobeynikov_a_g@mail.ru

Заболеева-Зотова Алла Викторовна, доктор технических наук, профессор Волгоградского технического университета, Волгоград, Россия E-mail: zabzot@gmail.com

Бенкевич Леонид Владимирович - кандидат физических наук и инженерной физики, научный сотрудник Массачусеттского Технологического Института (MIT), обсерватория Хэйстек, Бостон, США E-mail: ibenkev@gmail.com

Морозов Михаил Николаевич - кандидат технических наук, профессор, руководитель лаборатории мультимедиа, заведующий кафедрой Информатики и системного программирования Поволжского государственного технологического университета, Йошкар-Ола, Россия E-mail: mikhail.n.morozov@gmail.com

Олзоева Сэсэг Ивановна - доктор технических наук, профессор, Восточно-Сибирский государственный университет технологий и управления (г. Улан-Уде) E-mail: sseseg@yandex.ru

Курейчик Владимир Викторович - доктор технических наук, профессор, заведующий кафедрой Систем автоматизации проектирования Технологического института «Южного федерального университета» в г.Таганрог, Россия E-mail: ykur@tsure.ru

Филатова Наталья Николаевна - доктор технических наук, профессор, Тверской государственный технический университет, Тверь, Россия E-mail: nfilatova99@mail.ru

Песошин Валерий Андреевич - член-корреспондент Академии наук Республики Татарстан, заслуженный деятель науки Республики Татарстан и Российской Федерации, доктор технических наук, профессор, заслуженный деятель науки и техники Республики Татарстан. Заведующий кафедрой Компьютерных систем Казанского национальный исследовательский университет им. А.Н. Туполева, Казань, Россия E-mail: pesoshin@evm.kstu-kai.ru

Краснов Сергей Викторович - доктор технических наук, профессор, проректор по научно-исследовательской работе, заведующий кафедрой Информатика и системы управления Волжского университета им. Татищева, Тольятти, Россия E-mail: krasnovtlt@mail.ru

Горохов Алексей Витальевич - доктор технических наук, профессор кафедры Прикладной математики и информационных технологий Поволжского государственного технологического университета, Йошкар-Ола, Россия E-mail: agv64@mail.ru

Галанина Наталья Андреевна - доктор технических наук, профессор, Чувашский государственный университет им. И.Н.Ульянова, Чебоксары, Россия E-mail: galaninacheb@mail.ru

Сюзов Владимир Васильевич - доктор технических наук, профессор, заведующий кафедрой Компьютерные системы и сети Московского государственного технического университета им. Н. Э. Баумана, Москва, Россия E-mail: v.suzev@bmstu.ru

Леухин Анатолий Николаевич - доктор физико-математических наук, профессор, заведующий кафедрой Информационной безопасности Поволжского государственного технологического университета, Йошкар-Ола, Россия E-mail: code@volgatech.net

Гвинианидзе Темур Николаевич - Доктор технических наук, профессор,
Государственный университет им. Ак. Церетели Грузия, г. Кутаиси, пр. Тamar-мепе 59.
П.и 4600. temuri1951@mail.ru

Council of Editors

Gelman Viktor Yakovlevich – Doctor of Technical Sciences, Professor, Professor of the Department of Medical Informatics and Physics of the I.I.Mechnikov Northwestern State Medical University, 41 Kirochnaya Str., St. Petersburg, 191015, Russia, gelm@sg2104.spb.edu

Polyakov Viktor Pavlovich – Doctor of Pedagogical Sciences, Professor, Chief Researcher of the Laboratory of Psychological, Pedagogical and Educational methodological support for the development of Informatization of Education of the Center for Informatization of Education of the Federal State Budgetary Scientific Institution "Institute of Education Management of the Russian Academy of Education", 105062, Moscow, Makarenko str., 5/16, p. 1B, polvikpal@mail.ru

Garmaev Bair Zayatuevich – Candidate of Physical and Mathematical Sciences, Researcher, Institute of Physical Materials Science of the Siberian Branch of the Russian Academy of Sciences, 670000, Russia, Republic of Buryatia, Ulan-Ude, Sakhyanova str., 6, room 313

Klimenko Anna Borisovna – Candidate of Technical Sciences, Researcher at the Research Institute of Multiprocessor Computing Systems named after Academician A.V. Kalyaev of the Southern Federal University (Research Institute of the Ministry of Internal Affairs of the Southern Federal University), 347935, Russia, Rostov region, Taganrog, ul. 8 Lane, 15

Lyutikova Larisa Adolfovna – Candidate of Physical and Mathematical Sciences, Head of the Department of Neuroinformatics and Machine Learning, Institute of Applied Mathematics and Automation of the Kabardino-Balkarian Scientific Center of the Russian Academy of Sciences - branch of the Kabardino-Balkarian Scientific Center of the Russian Academy of Sciences (IPMA KBSC RAS), 360000, Russia, Republic of Kabardino-Balkaria, Nalchik, 89a Shortanova str.

Mustafayev Arslan Hasanovich – Doctor of Technical Sciences, Professor, State Autonomous Educational Institution of Higher Education "Dagestan State University of National Economy", Department of "Information Technologies and Information Security", 367015, Russia, Republic of Dagestan, Makhachkala, Ataeva str., 5, office 4.5

Alexander V. Shestakov – Candidate of Technical Sciences, Associate Professor, Southern Federal University, Department of Computer Engineering, 24/2 Svobody str., Taganrog, Rostov Region, 347902, Russia

Sidorkina Irina Gennadievna - Doctor of Technical Sciences, Professor, Dean of the Faculty of Computer Science and Computer Engineering of the Volga State Technological University, Yoshkar-Ola, Russia E-mail: dekan_fivt@mail.ru

Ekaterina Prasolova-Forland - PhD, Norwegian University of Science and Technology (NTNU), Trondheim, Norway E-mail: Ekaterina.Prasolova-Forland@idi.ntnu.no

Golenkov Vladimir Vasilyevich - Doctor of Technical Sciences, Professor, Head of the Department of Intelligent Information Technologies of the Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus E-mail: golen@bsuir.by

Domoshnitsky Alexander Isaakovich - Candidate of Physical and Mathematical Sciences, Dean of the Faculty of Natural Sciences of the University Center in Ariel, Israel, Samaria E-mail: adom@ariel.ac.il Department of Mathematics and Computer Sciences, The Ariel University Center of Samaria, 44837 Ariel, ISRAEL

Korobeynikov Anatoly Grigorievich - Doctor of Technical Sciences, Professor, Institute of

Terrestrial Magnetism, Ionosphere and Radio Wave Propagation of the Russian Academy of Sciences (IZMIRAN), St. Petersburg Branch E-mail: korobeynikov_a_g@mail.ru

Zaboleeva-Zotova Alla Viktorovna, Doctor of Technical Sciences, Professor of Volgograd Technical University, Volgograd, Russia E-mail: zabzot@gmail.com

Leonid V. Benkevich - Candidate of Physical Sciences and Engineering Physics, Researcher at the Massachusetts Institute of Technology (MIT), Haystack Observatory, Boston, USA E-mail: lbenkev@gmail.com

Mikhail N. Morozov - Candidate of Technical Sciences, Professor, Head of the Multimedia Laboratory, Head of the Department of Computer Science and System Programming of the Volga State Technological University, Yoshkar-Ola, Russia E-mail: mikhail.n.morozov@gmail.com

Olzoeva Seseg Ivanovna - Doctor of Technical Sciences, Professor, East Siberian State University of Technology and Management (Ulan-Ude) E-mail: sseseg@yandex.ru

Kureychik Vladimir Viktorovich - Doctor of Technical Sciences, Professor, Head of the Department of Design Automation Systems of the Technological Institute of the Southern Federal University in Taganrog, Russia E-mail: vkur@tsure.ru

Natalia Filatova - Doctor of Technical Sciences, Professor, Tver State Technical University, Tver, Russia E-mail: nfilatova99@mail.ru

Pesoshin Valery Andreevich - Corresponding member of the Academy of Sciences of the Republic of Tatarstan, Honored Scientist of the Republic of Tatarstan and the Russian Federation, Doctor of Technical Sciences, Professor, Honored Worker of Science and Technology of the Republic of Tatarstan. Head of the Department of Computer Systems of Kazan National Research University named after A.N. Tupolev, Kazan, Russia E-mail: pesoshin@evm.kstu-kai.ru

Krasnov Sergey Viktorovich - Doctor of Technical Sciences, Professor, Vice-Rector for Research, Head of the Department of Computer Science and Control Systems of the Volga State University. Tatishcheva, Togliatti, Russia E-mail: krasnovtlt@mail.ru

Gorokhov Alexey Vitalievich - Doctor of Technical Sciences, Professor of the Department of Applied Mathematics and Information Technologies of the Volga State Technological University, Yoshkar-Ola, Russia E-mail: agv64@mail.ru

Galanina Natalia Andreevna - Doctor of Technical Sciences, Professor, I.N.Ulyanov Chuvash State University, Cheboksary, Russia E-mail: galaninacheb@mail.ru

Vladimir V. Syuzev - Doctor of Technical Sciences, Professor, Head of the Department of Computer Systems and Networks of the Bauman Moscow State Technical University, Moscow, Russia E-mail: v.suzev@bmstu.ru

Leukhin Anatoly Nikolaevich - Doctor of Physical and Mathematical Sciences, Professor, Head of the Department of Information Security of the Volga State Technological University, Yoshkar-Ola, Russia E-mail: code@volgatech.net

Gvinianidze Temur Nikolaevich - Doctor of Technical Sciences, Professor, Ak. Tsereteli State University Georgia, Kutaisi, 59 Tamar-mepe Ave., and 4600. temuri1951@mail.ru

Требования к статьям

Журнал является научным. Направляемые в издательство статьи должны соответствовать тематике журнала (с его рубрикатором можно ознакомиться на сайте издательства), а также требованиям, предъявляемым к научным публикациям.

Рекомендуемый объем от 12000 знаков.

Структура статьи должна соответствовать жанру научно-исследовательской работы. В ее содержании должны обязательно присутствовать и иметь четкие смысловые разграничения такие разделы, как: предмет исследования, методы исследования, апелляция к оппонентам, выводы и научная новизна.

Не приветствуется, когда исследователь, трактуя в статье те или иные научные термины, вступает в заочную дискуссию с авторами учебников, учебных пособий или словарей, которые в узких рамках подобных изданий не могут широко излагать свое научное воззрение и заранее оказываются в проигрышном положении. Будет лучше, если для научной полемики Вы обратитесь к текстам монографий или диссертационных работ оппонентов.

Не превращайте научную статью в публицистическую: не наполняйте ее цитатами из газет и популярных журналов, ссылками на высказывания по телевидению.

Ссылки на научные источники из Интернета допустимы и должны быть соответствующим образом оформлены.

Редакция отвергает материалы, напоминающие реферат. Автору нужно не только продемонстрировать хорошее знание обсуждаемого вопроса, работ ученых, исследовавших его прежде, но и привнести своей публикацией определенную научную новизну.

Не принимаются к публикации избранные части из диссертаций, книг, монографий, поскольку стиль изложения подобных материалов не соответствует журнальному жанру, а также не принимаются материалы, публиковавшиеся ранее в других изданиях.

В случае отправки статьи одновременно в разные издания автор обязан известить об этом редакцию. Если он не сделал этого заблаговременно, рискует репутацией: в дальнейшем его материалы не будут приниматься к рассмотрению.

Уличенные в плагиате попадают в «черный список» издательства и не могут рассчитывать на публикацию. Информация о подобных фактах передается в другие издательства, в ВАК и по месту работы, учебы автора.

Статьи представляются в электронном виде только через сайт издательства <http://www.e-notabene.ru> кнопка "Авторская зона".

Статьи без полной информации об авторе (соавторах) не принимаются к рассмотрению, поэтому автор при регистрации в авторской зоне должен ввести полную и корректную информацию о себе, а при добавлении статьи - о всех своих соавторах.

Не набирайте название статьи прописными (заглавными) буквами, например: «ИСТОРИЯ КУЛЬТУРЫ...» — неправильно, «История культуры...» — правильно.

При добавлении статьи необходимо прикрепить библиографию (минимум 10–15 источников, чем больше, тем лучше).

При добавлении списка использованной литературы, пожалуйста, придерживайтесь следующих стандартов:

- [ГОСТ 7.1-2003 Библиографическая запись. Библиографическое описание. Общие требования и правила составления.](#)
- [ГОСТ 7.0.5-2008 Библиографическая ссылка. Общие требования и правила составления](#)

В каждой ссылке должен быть указан только один диапазон страниц. В теле статьи ссылка на источник из списка литературы должна быть указана в квадратных скобках, например, [1]. Может быть указана ссылка на источник со страницей, например, [1, с. 57], на группу источников, например, [1, 3], [5-7]. Если идет ссылка на один и тот же источник, то в теле статьи нумерация ссылок должна выглядеть так: [1, с. 35]; [2]; [3]; [1, с. 75-78]; [4].... А в библиографии они должны отображаться так:

[1]

[2]

[3]

[4]....

Постраничные ссылки и сноски запрещены. Если вы используете сноску, не содержащую ссылку на источник, например, разъяснение термина, включите сноску в текст статьи.

После процедуры регистрации необходимо прикрепить аннотацию на русском языке, которая должна состоять из трех разделов: Предмет исследования; Метод, методология исследования; Новизна исследования, выводы.

Прикрепить 10 ключевых слов.

Прикрепить саму статью.

Требования к оформлению текста:

- Кавычки даются уголками (« ») и только кавычки в кавычках — лапками (" ").
- Тире между датами дается короткое (Ctrl и минус) и без отбивок.
- Тире во всех остальных случаях дается длинное (Ctrl, Alt и минус).
- Даты в скобках даются без г.: (1932–1933).
- Даты в тексте даются так: 1920 г., 1920-е гг., 1540–1550-е гг.
- Недопустимо: 60-е гг., двадцатые годы двадцатого столетия, двадцатые годы XX столетия, 20-е годы XX столетия.
- Века, король такой-то и т.п. даются римскими цифрами: XIX в., Генрих IV.
- Инициалы и сокращения даются с пробелом: т. е., т. д., М. Н. Иванов. Неправильно: М.Н. Иванов, М.Н. Иванов.

ВСЕ СТАТЬИ ПУБЛИКУЮТСЯ В АВТОРСКОЙ РЕДАКЦИИ.

По вопросам публикации и финансовым вопросам обращайтесь к администратору Зубковой Светлане Вадимовне

E-mail: info@nbpublish.com

или по телефону +7 (966) 020-34-36

Подробные требования к написанию аннотаций:

Аннотация в периодическом издании является источником информации о содержании статьи и изложенных в ней результатах исследований.

Аннотация выполняет следующие функции: дает возможность установить основное

содержание документа, определить его релевантность и решить, следует ли обращаться к полному тексту документа; используется в информационных, в том числе автоматизированных, системах для поиска документов и информации.

Аннотация к статье должна быть:

- информативной (не содержать общих слов);
- оригинальной;
- содержательной (отражать основное содержание статьи и результаты исследований);
- структурированной (следовать логике описания результатов в статье);

Аннотация включает следующие аспекты содержания статьи:

- предмет, цель работы;
- метод или методологию проведения работы;
- результаты работы;
- область применения результатов; новизна;
- выводы.

Результаты работы описывают предельно точно и информативно. Приводятся основные теоретические и экспериментальные результаты, фактические данные, обнаруженные взаимосвязи и закономерности. При этом отдается предпочтение новым результатам и данным долгосрочного значения, важным открытиям, выводам, которые опровергают существующие теории, а также данным, которые, по мнению автора, имеют практическое значение.

Выводы могут сопровождаться рекомендациями, оценками, предложениями, гипотезами, описанными в статье.

Сведения, содержащиеся в заглавии статьи, не должны повторяться в тексте аннотации. Следует избегать лишних вводных фраз (например, «автор статьи рассматривает...», «в статье рассматривается...»).

Исторические справки, если они не составляют основное содержание документа, описание ранее опубликованных работ и общеизвестные положения в аннотации не приводятся.

В тексте аннотации следует употреблять синтаксические конструкции, свойственные языку научных и технических документов, избегать сложных грамматических конструкций.

Гонорары за статьи в научных журналах не начисляются.

Цитирование или воспроизведение текста, созданного ChatGPT, в вашей статье

Если вы использовали ChatGPT или другие инструменты искусственного интеллекта в своем исследовании, опишите, как вы использовали этот инструмент, в разделе «Метод» или в аналогичном разделе вашей статьи. Для обзоров литературы или других видов эссе, ответов или рефератов вы можете описать, как вы использовали этот инструмент, во введении. В своем тексте предоставьте prompt - командный вопрос, который вы использовали, а затем любую часть соответствующего текста, который был создан в ответ.

К сожалению, результаты «чата» ChatGPT не могут быть получены другими читателями, и хотя невозстановимые данные или цитаты в статьях APA Style обычно цитируются как личные сообщения, текст, сгенерированный ChatGPT, не является сообщением от человека.

Таким образом, цитирование текста ChatGPT из сеанса чата больше похоже на совместное использование результатов алгоритма; таким образом, сделайте ссылку на автора алгоритма записи в списке литературы и приведите соответствующую цитату в тексте.

Пример:

На вопрос «Является ли деление правого полушария левого полушария реальным или метафорой?» текст, сгенерированный ChatGPT, показал, что, хотя два полушария мозга в некоторой степени специализированы, «обозначение, что люди могут быть охарактеризованы как «левополушарные» или «правополушарные», считается чрезмерным упрощением и популярным мифом» (OpenAI, 2023).

Ссылка в списке литературы

OpenAI. (2023). ChatGPT (версия от 14 марта) [большая языковая модель].
<https://chat.openai.com/chat>

Вы также можете поместить полный текст длинных ответов от ChatGPT в приложение к своей статье или в дополнительные онлайн-материалы, чтобы читатели имели доступ к точному тексту, который был сгенерирован. Особенно важно задокументировать точный созданный текст, потому что ChatGPT будет генерировать уникальный ответ в каждом сеансе чата, даже если будет предоставлен один и тот же командный вопрос. Если вы создаете приложения или дополнительные материалы, помните, что каждое из них должно быть упомянуто по крайней мере один раз в тексте вашей статьи в стиле APA.

Пример:

При получении дополнительной подсказки «Какое представление является более точным?» в тексте, сгенерированном ChatGPT, указано, что «разные области мозга работают вместе, чтобы поддерживать различные когнитивные процессы» и «функциональная специализация разных областей может меняться в зависимости от опыта и факторов окружающей среды» (OpenAI, 2023; см. Приложение А для полной расшифровки). .

Ссылка в списке литературы

OpenAI. (2023). ChatGPT (версия от 14 марта) [большая языковая модель].
<https://chat.openai.com/chat> Создание ссылки на ChatGPT или другие модели и программное обеспечение ИИ

Приведенные выше цитаты и ссылки в тексте адаптированы из шаблона ссылок на программное обеспечение в разделе 10.10 Руководства по публикациям (Американская психологическая ассоциация, 2020 г., глава 10). Хотя здесь мы фокусируемся на ChatGPT, поскольку эти рекомендации основаны на шаблоне программного обеспечения, их можно адаптировать для учета использования других больших языковых моделей (например, Bard), алгоритмов и аналогичного программного обеспечения.

Ссылки и цитаты в тексте для ChatGPT форматируются следующим образом:

OpenAI. (2023). ChatGPT (версия от 14 марта) [большая языковая модель].
<https://chat.openai.com/chat>

Цитата в скобках: (OpenAI, 2023)

Описательная цитата: OpenAI (2023)

Давайте разберем эту ссылку и посмотрим на четыре элемента (автор, дата, название и

источник):

Автор: Автор модели OpenAI.

Дата: Дата — это год версии, которую вы использовали. Следуя шаблону из Раздела 10.10, вам нужно указать только год, а не точную дату. Номер версии предоставляет конкретную информацию о дате, которая может понадобиться читателю.

Заголовок. Название модели — «ChatGPT», поэтому оно служит заголовком и выделено курсивом в ссылке, как показано в шаблоне. Хотя OpenAI маркирует уникальные итерации (например, ChatGPT-3, ChatGPT-4), они используют «ChatGPT» в качестве общего названия модели, а обновления обозначаются номерами версий.

Номер версии указан после названия в круглых скобках. Формат номера версии в справочниках ChatGPT включает дату, поскольку именно так OpenAI маркирует версии. Различные большие языковые модели или программное обеспечение могут использовать различную нумерацию версий; используйте номер версии в формате, предоставленном автором или издателем, который может представлять собой систему нумерации (например, Версия 2.0) или другие методы.

Текст в квадратных скобках используется в ссылках для дополнительных описаний, когда они необходимы, чтобы помочь читателю понять, что цитируется. Ссылки на ряд общих источников, таких как журнальные статьи и книги, не включают описания в квадратных скобках, но часто включают в себя вещи, не входящие в типичную рецензируемую систему. В случае ссылки на ChatGPT укажите дескриптор «Большая языковая модель» в квадратных скобках. OpenAI описывает ChatGPT-4 как «большую мультимодальную модель», поэтому вместо этого может быть предоставлено это описание, если вы используете ChatGPT-4. Для более поздних версий и программного обеспечения или моделей других компаний могут потребоваться другие описания в зависимости от того, как издатели описывают модель. Цель текста в квадратных скобках — кратко описать тип модели вашему читателю.

Источник: если имя издателя и имя автора совпадают, не повторяйте имя издателя в исходном элементе ссылки и переходите непосредственно к URL-адресу. Это относится к ChatGPT. URL-адрес ChatGPT: <https://chat.openai.com/chat>. Для других моделей или продуктов, для которых вы можете создать ссылку, используйте URL-адрес, который ведет как можно более напрямую к источнику (т. е. к странице, на которой вы можете получить доступ к модели, а не к домашней странице издателя).

Другие вопросы о цитировании ChatGPT

Вы могли заметить, с какой уверенностью ChatGPT описал идеи латерализации мозга и то, как работает мозг, не ссылаясь ни на какие источники. Я попросил список источников, подтверждающих эти утверждения, и ChatGPT предоставил пять ссылок, четыре из которых мне удалось найти в Интернете. Пятая, похоже, не настоящая статья; идентификатор цифрового объекта, указанный для этой ссылки, принадлежит другой статье, и мне не удалось найти ни одной статьи с указанием авторов, даты, названия и сведений об источнике, предоставленных ChatGPT. Авторам, использующим ChatGPT или аналогичные инструменты искусственного интеллекта для исследований, следует подумать о том, чтобы сделать эту проверку первоисточников стандартным процессом. Если источники являются реальными, точными и актуальными, может быть лучше прочитать эти первоисточники, чтобы извлечь уроки из этого исследования, и перефразировать или процитировать эти статьи, если применимо, чем использовать их интерпретацию модели.

Материалы журналов включены:

- в систему Российского индекса научного цитирования;
- отображаются в крупнейшей международной базе данных периодических изданий Ulrich's Periodicals Directory, что гарантирует значительное увеличение цитируемости;
- Всем статьям присваивается уникальный идентификационный номер Международного регистрационного агентства DOI Registration Agency. Мы формируем и присваиваем всем статьям и книгам, в печатном, либо электронном виде, оригинальный цифровой код. Префикс и суффикс, будучи прописанными вместе, образуют определяемый, цитируемый и индексируемый в поисковых системах, цифровой идентификатор объекта — digital object identifier (DOI).

[Отправить статью в редакцию](#)

Этапы рассмотрения научной статьи в издательстве NOTA BENE.



Содержание

Лизнева Ю.С., Костюкович А.Е., Кокорева Е.В. Анализ возможностей определения местоположения в сети Wi-Fi с использованием алгоритмов нейронных сетей	1
Дагаев А.Е., Попов Д.И. Сравнение автоматического обобщения текстов на русском языке	13
Булгаков В.Д., Гвоздецкий И.Н. Модель и алгоритм консенсуса Proof of Performance	23
Димитриченко Д.П. Анализ целесообразного поведения различных типов автоматов в условиях игры в размещения	49
Захаров А.А. Метод обнаружения объектов на изображениях на основе нейронных сетей на графах и небольшого количества обучающих примеров	66
Князев М.А., Шаброва А.С., Крючков А.А. Подход к выбору механизмов защиты устройств персонального Интернета вещей на основе математической модели с двумя критериями	76
Хлесткин А.Ю., Райков А.В., Казанцев А.А., Емелин Д.П., Ларин Д.В. Роль операционных систем и оболочек в облачных вычислениях: анализ ОС и оболочек, используемых в облачных платформах и их влияние на облачную инфраструктуру	93
Смирнов А.А., Подольский Е.А., Черенков А.В., Государев И.Б. Сравнительный анализ производительности сред исполнения JavaScript кода: Node.js, Deno и Bun	109
Копышева Т.Н., Митрофанова Т.В., Смирнова Т.Н., Христофорова А.В. Применение платформ Neural Network Wizard и Loginom Community для решения задач аппроксимации и классификации в образовательном процессе	124
Ишанхонов А.Ю., Пшиченко Д.В., Можаровский Е.А., Алуев А.С. Роль больших языковых моделей в интегрированных средах разработки нового поколения	140
Шейнман В., Стариков Д.Д., Тюменцев Д.В., Вавилов Г.Д. Повышение эффективности процессов разработки программного обеспечения: контейнерные технологии	151
Пекунов В.В. Объектно-транзакционные модели программ на алгоритмических языках	162
Хейфиц А.Е., Янчус В.Э., Борович Е.В. Методика проведения экспериментального исследования по восприятию визуальной информации в области периферийного зрения человека	170
Дамдинова Т.Ц., Дамдинов З.Ш., Прудова Л.Ю., Бубеев И.Т. Моделирование пор нерегулярной формы по яркости пикселей цифрового изображения	182
Золотухина Д.Ю. Эффективность распределённых кэширующих платформ в современных backend-архитектурах: сравнительный анализ Redis и Hazelcast	192
Англоязычные метаданные	205

Contents

Lizneva Y.S., Kostyukovich A.E., Kokoreva E.V. Analysis of the possibilities of determining location in a Wi-Fi network using neural network algorithms	1
Dagaev A.E., Popov D.I. Comparison of automatic summarization of texts in Russian	13
Bulgakov V.D., Gvozdevsky I.N. Proof of Performance Consensus Model and Algorithm	23
Dimitrichenko D.P. Analysis of the appropriate behavior of various types of automata in the conditions of the placement game	49
Zakharov A.A. A method for detecting objects in images based on neural networks on graphs and a small number of training examples	66
Knyazev M.A., Shabrova A.S., Kryuchkov A.A. An approach to choosing protection mechanisms for personal Internet of Things devices based on a mathematical model with two criteria	76
Khlestkin A.Y., Raikov A.V., Kazantsev A.A., Emelin D.P., Larin D.V. The role of operating systems and shells in cloud computing: analysis of OS and shells used in cloud platforms and their impact on cloud infrastructure	93
Smirnov .A., Podolskiy E.A., Cherenkov A.V., Gosudarev I.B. A comparative analysis of the performance of JavaScript code execution environments: Node.js, Deno and Bun	109
Kopysheva T.N., Mitrofanova T.V., Smirnova T.N., Khristoforova A.V. The use of artificial intelligence systems for data processing in the educational process	124
Ishankhonov A.Y., Pshychenko D.V., Mozharovskii E.A., Aluev A.S. The Role of LLM in Next-Generation Integrated Development Environments	140
Sheinman V., Starikov D.D., Tiumentsev D.V., Vavilov G.D. Improving the Efficiency of Software Development Processes: Container Technologies	151
Pekunov V.V. Object-transactional models of programs in algorithmic languages	162
Kheyfits A.E., Yanchus V.E., Borevich E.V. Methodology for conducting an experimental study on the perception of visual information in the field of human peripheral vision	170
Damdinova T.T., Damdinov Z.S., Prudova L.Y., Bubeev I.T. Modeling irregular-shaped pores based on the brightness of pixels in a digital image	182
Zolotukhina D. The Efficiency of Distributed Caching Platforms in Modern Backend Architectures: A Comparative Analysis of Redis and Hazelcast	192
Metadata in english	205

Программные системы и вычислительные методы

Правильная ссылка на статью:

Лизнева Ю.С., Костюкович А.Е., Кокорева Е.В. Анализ возможностей определения местоположения в сети Wi-Fi с использованием алгоритмов нейронных сетей // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.72107 EDN: CSDXDU URL: https://nbpublish.com/library_read_article.php?id=72107

Анализ возможностей определения местоположения в сети Wi-Fi с использованием алгоритмов нейронных сетей

Лизнева Юлия Сергеевна

ORCID: 0000-0001-9746-7413

кандидат технических наук

доцент, Институт телекоммуникаций; Сибирский государственный университет телекоммуникаций и информатики

630102, Россия, Новосибирская область, г. Новосибирск, ул. Кирова, 86

✉ ktm5r@rambler.ru



Костюкович Анатолий Егорович

кандидат технических наук

доцент, кафедра автоматической электросвязи; Сибирский государственный университет телекоммуникаций и информатики
Руководитель, Научно-образовательный центр для наукоемких промышленных предприятий г. Новосибирска

630102, Россия, Новосибирская область, г. Новосибирск, ул. Кирова, 86, оф. 404

✉ aek1954@gmail.com



Кокорева Елена Викторовна

ORCID: 0000-0002-4437-7251

кандидат технических наук

доцент, Сибирский государственный университет телекоммуникаций и информатики

630102, Россия, Новосибирская область, г. Новосибирск, ул. Кирова, 86, каб. 605

✉ elen.vik@gmail.com



[Статья из рубрики "Математическое и программное обеспечение новых информационных технологий"](#)

DOI:

10.7256/2454-0714.2024.4.72107

EDN:

CSDXDU

Дата направления статьи в редакцию:

28-10-2024

Дата публикации:

06-11-2024

Аннотация: Позиционирование внутри помещений в сети Wi-Fi относится к классу задач, в которых зависимость выходных характеристик от входных переменных подвержена влиянию многих параметров и внешних факторов. При решении таких задач необходимо учитывать, что в определении местоположения существенный интерес представляет не только определение статических координат объекта, но и прогнозирование вектора его перемещений. В случае, когда местоположение объекта определяется только по уровню мощности сигнала, принимаемого от нескольких точек доступа в сети Wi-Fi, использование моделей затухания сигнала, учитывающих условия распространения радиоволн внутри помещений, затруднено из-за необходимости в достоверных сведениях о материале перекрытий, пола и потолка, наличии фиксированных и мобильных затеняющих объектов и т.д. Поскольку электромагнитная обстановка внутри помещения меняется в зависимости от многих факторов, вышеупомянутые модели приходится подстраивать под эти изменения. Так как нахождение закономерностей в большом объёме данных требует нестандартных алгоритмов, для решения задачи позиционирования можно использовать искусственные нейронные сети. Важно выбрать архитектуру нейронной сети, способную учитывать изменения уровня сигнала, принимаемого мобильным устройством от точек доступа сети Wi-Fi. Перед обучением нейронной сети проводится предобработка статистических данных. Например, из набора данных для машинного обучения исключаются аномальные случаи, когда в одной измерительной точке устройство фиксирует сигнал менее чем от трех точек доступа. В результате анализа статистических данных было установлено, что одинаковое расстояние между измерительными точками приводит к тому, что нейронная сеть неверно определяет местоположение объекта. В работе показано, что в целях повышения точности позиционирования местоположения в условиях сложной радиообстановки при составлении радиокарт необходимо определить оптимальные варьирующиеся расстояния между измерительными точками. Проведённые экспериментальные исследования с учётом предложенного подхода к оптимизации расстояний между измерительными точками доказывают, что точность определения местоположения в подавляющем большинстве измерительных точек достигает 100%.

Ключевые слова:

Wi-Fi, позиционирование, измерительная точка, RSSI, нейронная сеть, мощность сигнала, скрытый слой, машинное обучение, обучающая выборка, набор обучающих данных

1. Введение

Большинство существующих Wi-Fi сетей планировались исходя из требований к качеству доставки широкополосного контента в конкретные помещения внутри зданий, при этом задача оптимизации частотно-территориального планирования при размещении точек доступа Wi-Fi для целей определения местоположения не ставилась.

Как следствие, в существующих сетях Wi-Fi передача данных в конкретные помещения внутри зданий выполняется вполне удовлетворительно, но реализовать возможность позиционирования с приемлемой точностью (до 5...7 м) в таких сетях невозможно.

Целью данной статьи являлась оценка потенциальных возможностей использования нейронных сетей для повышения точности определения местоположения в существующих неоптимально построенных Wi-Fi сетях.

Для проведения экспериментальных исследований была выбрана типичная для университетов сеть Wi-Fi, в которой точки доступа размещались владельцами помещений интуитивно, без какого-либо частотно-территориального планирования.

Характерной особенностью таких сетей Wi-Fi является отсутствие данных об уровне мощности передатчика по каждой точке доступа, отсутствие сведений о материалах стен, что не позволяет применять соответствующие модели распространения радиоволн при оценке расстояния от мобильных терминалов до этих точек доступа, например, при использовании алгоритмов трилатерации [\[1\]](#).

В этих условиях единственно доступным способом позиционирования может быть способ, основанный на обработке результатов натурных измерений (метод построения образцовых карт) [\[2, 3\]](#). Учитывая большой разброс результатов измерений, задача данного исследования состояла в том, чтобы даже в этих неблагоприятных условиях оценить потенциально достижимую точность позиционирования с привлечением методов статистической обработки и элементов искусственного интеллекта в виде некоторых нейронных сетей [\[4, 5\]](#).

2. Анализ исходных данных

Экспериментальная часть состояла в проведении измерений уровня мощности сигнала RSSI по всем этажам одного из корпусов университета. На каждом этаже были отмечены измерительные точки (ИТ), расстояние между которыми изначально было выбрано равным 8 м.

Согласно [\[6, 7\]](#), метод позиционирования объекта по радиокарте состоит из двух этапов:

- построение радиокарты по результатам измерений;
- определение местоположения объекта по результатам статистической обработки экспериментальных данных.

В каждой измерительной точке были проведены замеры уровней RSSI от всех видимых точек доступа. Видимой считалась точка доступа, значение RSSI от которой фиксировалось на уровне не ниже -95 дБм.

Уровень сигнала в каждой измерительной точке фиксировался пятью мобильными устройствами, при этом каждым мобильным устройством выполнялось 10 измерений. Таким образом, в каждой измерительной точке фиксировалось 50 результатов измерений RSSI.

По множеству причин разброс значений RSSI был достаточно большим, поэтому для последующего использования экспериментальных данных в процессе определения местоположения, была проведена статистическая обработка экспериментальных данных, определены средние значения RSSI для каждой измерительной точки, а также отклонения от средних значений. На рис. 1 показано изменение среднего значения RSSI

при удалении от точки доступа №1 как для отдельных мобильных устройств, так и усредненные результаты по всем мобильным устройствам.

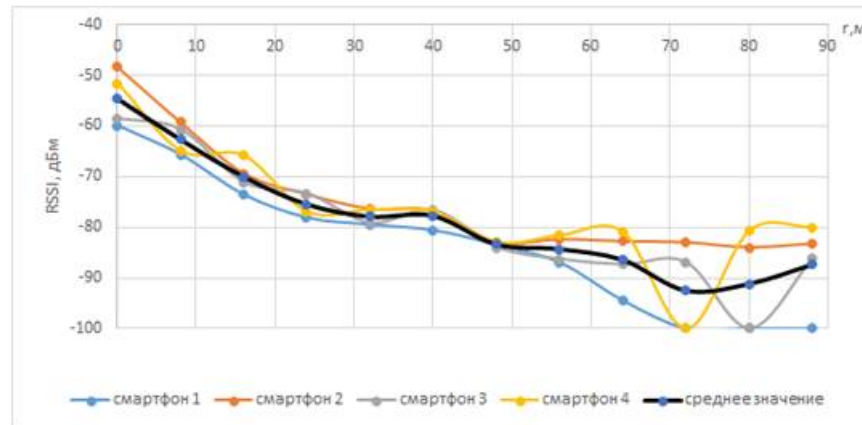


Рис. 1. Зависимость среднего RSSI (точка доступа №1), дБм

Анализ рисунка показал, что разброс значений уровня сигнала в каждой измерительной точке, в зависимости от смартфона, может достигать 20 дБм. Кроме того, на рис. 1 видно, что при удалении от точки доступа наблюдается плавное снижение величины размаха, а затем резкий рост. Поэтому, при построении радиокарты были использованы результаты измерений несколькими мобильными устройствами от разных вендоров. На основании того, что разброс уровней RSSI в каждой измерительной точке значительный, необходимо учитывать не только математическое ожидание, но и среднее квадратическое отклонение.

Для исключения из экспериментальных данных аномальных значений были построены графики доверительных интервалов.

В частности, на рис. 2 показаны границы доверительного интервала, по результатам статистической обработки экспериментальных данных, принятых от точки доступа №1 в разных измерительных точках.

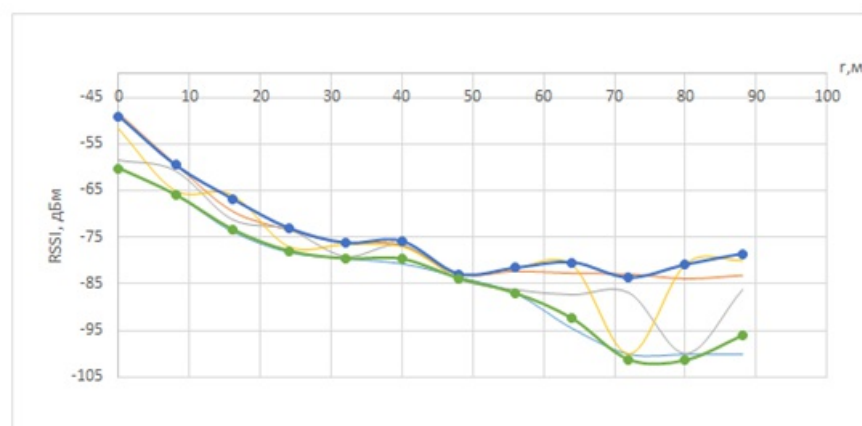


Рис. 2. Доверительный интервал (точка доступа №1), дБм

Анализ рисунка показал, что все данные, полученные в результате измерения уровня RSSI, находятся внутри доверительного интервала. То есть из дальнейшего анализа исключены аномальные значения, учет которых может влиять на определение местоположения объекта.

Аналогично был проведен статический анализ данных, полученных от остальных точек

доступа.

Уровень мощности сигнала в условиях сложной радиобстановки, когда помехи создают точки доступа из близлежащих строений и клиентские устройства (смартфоны, в которых активирована функция точки доступа), может сильно отклоняться от среднего значения. Следовательно, одной из задач, решаемых в процессе составления радиокарты, является выбор координат измерительных точек.

Для определения оптимального размещения измерительных точек был проведен анализ радиобстановки с взаимным влиянием точек доступа друг на друга. На рис. 3 показаны границы доверительного интервала, полученного при обработке статистических данных, принятых от точки доступа №2, расстояние между измерительными точками 8 метров.

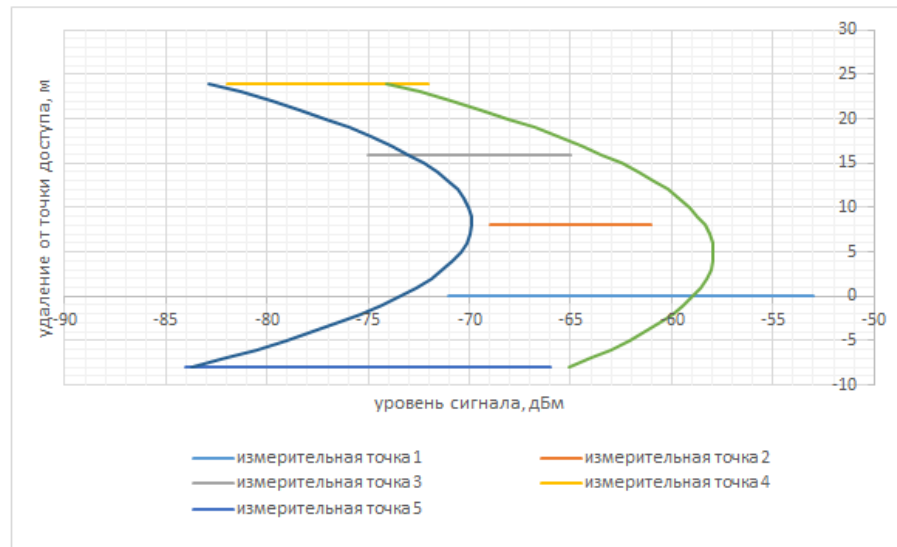


Рис. 3. Доверительный интервал (точка доступа №2), дБм (так как точка доступа находится не в одной из крайних комнат здания и измерения проведены последовательно по всему коридору, плюс на графике означает приближение к точке доступа, минус – удаление от точки доступа, что важно для решения задач трекинга)

Так как в каждой измерительной точке смартфон получает сигнал от нескольких точек доступа, аналогичные графики доверительных интервалов были построены для всех точек доступа.

Анализ рис. 3 показывает, что выбранное расстояние между измерительными точками, равное 8 метрам, приводит к более частому «перекрытию» уровня принимаемых сигналов от разных точек доступа, что может влиять на точность позиционирования.

Принимая во внимание, что все изыскания проводятся в существующей инфраструктуре, изменение которой не рассматривалось в рамках данного исследования, дальнейшая задача заключается в оптимизации выбора расстояния между измерительными точками.

3. Оптимизация выбора расстояния между измерительными точками

Для выбора расстояния между измерительными точками были получены зависимости средних значений RSSI, полученных от каждой точки доступа отдельно. На рис. 4 показано, что для аппроксимации статистических данных, полученных от точки доступа №3, наиболее подходящей является полиномиальная функция (коэффициент детерминации равен 0,913) [8–10]. То есть, изменяя величину X в уравнении тренда, можно получить такое расстояние между измерительными точками, при котором

«перекрывание» уровня сигналов будет минимальным или исчезнет.

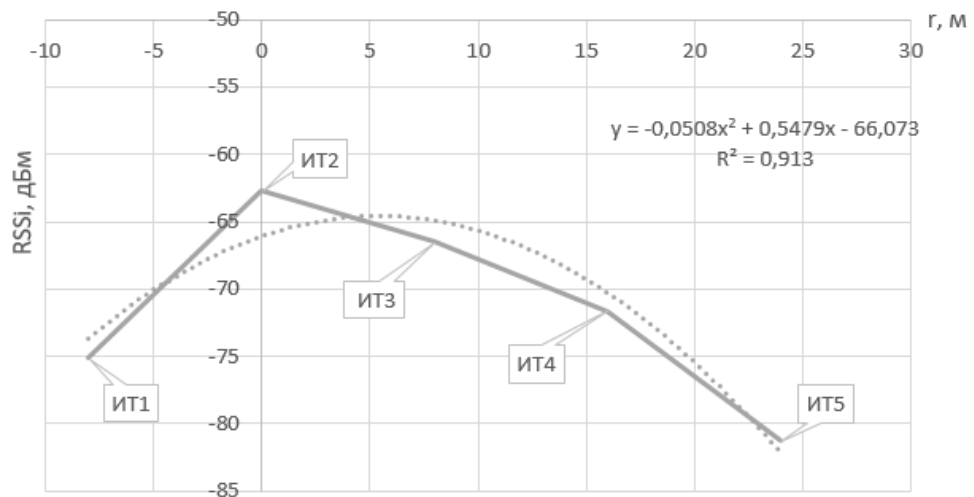


Рис. 4. Зависимость среднего RSSI (точка доступа №3), дБм

Далее были рассчитаны вероятности того, что уровень мощности не выходит за пределы заданного диапазона в рассматриваемых измерительных точках (табл. 1).

Таблица 1. Вероятность нахождения уровня мощности в заданном диапазоне

уровень мощности, дБм		измерительная точка				
		5	4	3	2	1
-88	-91	0,206				
-85	-88	0,147				
-82	-85	0,088	0,030			
-79	-82	0,147	0,061	0,032		
-76	-79	0,118	0,121	0,065	0,029	0,364
-73	-76	0,059	0,182	0,032	0,000	0,273
-70	-73	0,235	0,273	0,226	0,088	0,227
-67	-70		0,182	0,129	0,059	0,091
-64	-67		0,121	0,194	0,353	0,000
-61	-64		0,030	0,258	0,118	0,000
-58	-61			0,065	0,059	0,000
-55	-58				0,206	0,045
-52	-55				0,088	

Проведем попарное сравнение уровня мощности в соседних измерительных точках. Так, в табл. 1 видно, что наиболее вероятное «перекрывание» мощности для измерительных точек 5 и 4 начинается с -73 дБм. То есть расстояние между измерительными точками необходимо уменьшить до 4 метров (см. рис. 4). Аналогично попарно сравнивая уровни мощности для остальных точек доступа были получены новые расстояния: ИТ4-ИТ3 – 10 м, ИТ3-ИТ2 – 7,5 м, ИТ2-ИТ1 – 9 м.

Далее с помощью нейронной сети была подтверждена оптимальность выбора расстояния между измерительными точками.

4. Решение задачи определения местоположения с помощью нейронных сетей

Перед началом обучения нейронной сети экспериментальные данные формируются в блок обучающей выборки. В табл. 2 приведен фрагмент обучающей выборки, полученной для фиксированного расстояния между измерительными точками (8 метров).

Если сигнал от точки доступа слабый (приемное устройство не зафиксировало уровень мощности), то в базу данных был занесен уровень -100 дБм.

Таблица 2. Фрагмент обучающей выборки

		Точка доступа			
		3	2	1	4
измерительная точка	4	-76	-76	-78	-100
	4	-76	-76	-78	-100
	3	-77	-72	-71	-100
	3	-68	-89	-74	-100
	2	-59	-100	-60	-80
	2	-56	-100	-61	-92
	1	-64	-100	-59	-81

После того, как сформирована обучающая выборка, необходимо выбрать архитектуру нейронной сети, задать число нейронов в скрытом слое, присвоить весовым коэффициентам некоторые значения, задать допустимую величину ошибки и нормировать значения обучающего вектора.

Так как нейронные сети в основном работают с данными, представленными числами из интервала [\[-1, 1\]](#), то необходимо провести нормировку обучающих данных. Кроме того, если значения сосредоточены в относительно небольшой области единичного интервала, информационное содержание таких входных данных невелико. В пределе нулевой энтропии, когда все данные совпадают, эти входные данные не несут никакой полезной информации. Напротив, если значения данных распределены в единичном интервале по нормальному закону, информация таких данных максимальна [11–13].

После того как входные данные приведены к интервалу [\[-1, 1\]](#), был проведен сравнительный анализ основных алгоритмов обучения для двух архитектур нейронной сети [\[14, 15\]](#). Для эксперимента была использована многослойная нейронная сеть с одним скрытым слоем (количество нейронов в этом слое определяется экспериментально). При выборе архитектуры нейронной сети обычно опробуется несколько конфигураций, поэтому в представленной работе был проведен сравнительный анализ двух вариантов: сеть прямой передачи и каскадная направленная сеть [\[16\]](#).

Перед использованием нейронной сети ее необходимо обучить. Так как соответствие между входами и выходами носит явно выраженный нелинейный характер, то была использована нейронная сеть с нелинейными функциями активации [\[17, 18\]](#).

Для выходного слоя была использована линейная функция, так как линейный уровень вывода позволяет сети производить значения вне диапазона [\[-1, 1\]](#).

Алгоритм обратного распространения эффективен в ситуациях, когда отношения между входом и выходом нелинейны, а количество обучающих данных велико [\[19\]](#). Недостатком классического алгоритма обратного распространения является большое число итераций для достижения минимума функции ошибки. Время, требуемое для вычисления производных от ошибки по весам на заданном тренировочном примере, пропорционально размерам сети, поскольку объем вычислений пропорционален количеству весов. Однако с увеличением размеров сети требуется большее количество тренировочных примеров, поэтому приходится модифицировать веса множество раз. Следовательно, время обучения растет значительно быстрее, чем размеры сети. С

другой стороны, большая скорость обучения приводит к неустойчивости процесса [20].

В данном исследовании был выбран алгоритм обучения Левенберга-Марквардта, который показал наилучший результат в условии сложной радиообстановки.

На рис. 5 наглядно показаны результаты работы нейронной сети по определению местоположения на одном этаже здания.



Рис. 5. Вероятность определения местоположения на одном этаже

На пересечении строк и столбцов цветом и цифрами указаны вероятности правильного определения местоположения для каждой измерительной точки через 8 метров.

В частности – зеленый цвет означает, что вероятность правильного определения равна 100%, красный цвет – вероятность ошибочного определения не равна 0%, что означает, что в этих местах зона неуверенного покрытия, поэтому точность определения местоположения будет более 8 метров, а цифрами указывается вероятность того, что мобильный терминал может находиться некоторой вероятностью в смежных и даже более удаленных точках какого-либо этажа.

Учитывая тот факт, что данные исследования проводились в сети Wi-Fi, построенной без частотно-территориального планирования, можно признать, что в этих условиях нейронная сеть показывает достаточно неплохие результаты – точность определения местоположения от 8 до 16 метров по большинству этажей, где радиопокрытие достаточное.

Далее был проведен анализ точности позиционирования с помощью обучающей выборки, полученной после оптимизации расстояний между измерительными точками (рис. 6).

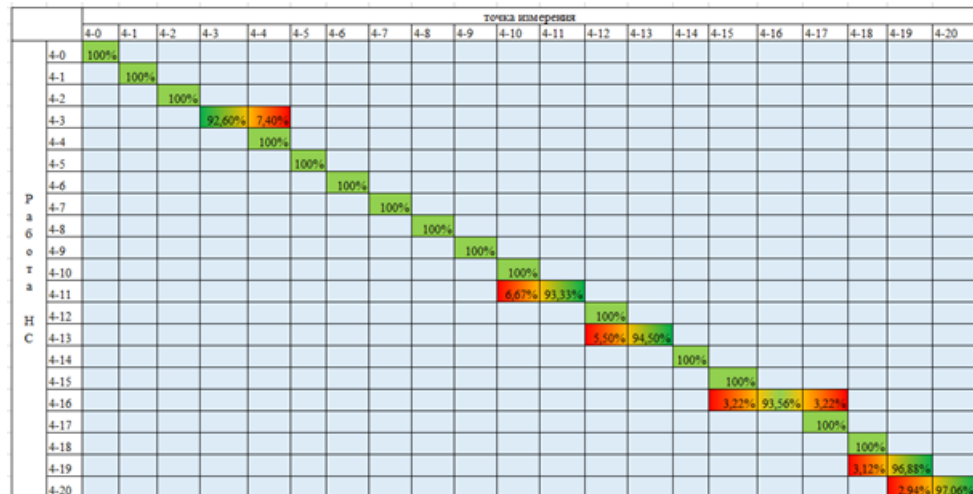


Рис. 6. Вероятность определения местоположения после изменения координат ИТ

На рис. 6 показаны результаты работы нейронной сети после оптимизации расстояния между измерительными точками. На данном рисунке видно, что вероятность правильного определения местоположения практически во всех измерительных точках достигает 100%, что можно считать очень хорошим результатом для неоптимально построенной Wi-Fi сети.

Анализ результатов, представленных на данном рисунке, показал, что в условиях сложной радиообстановки, оптимизация расстояния между измерительными точками приводит к повышению точности позиционирования.

5. Заключение

В данной работе были исследованы потенциальные возможности повышения точности определения местоположения на основе применения алгоритмов нейронных сетей в радиосетях Wi-Fi, организованных без учета частотно-территориального планирования.

Полученные в процессе научной работы результаты показали, что применение нейронной сети прямой передачи (алгоритм обучения Левенберга-Марквардта) позволяет даже в условиях очень плохого радиопокрытия определять местоположение мобильного терминала с вполне приемлемой для конечного пользователя точностью и достоверностью.

Можно сделать вывод, что если строить любые радиосети с учетом размещения точек доступа (базовых станций) не только для качественной доставки интернет-контента абонентам мобильной сети, но и с целью оказания услуг определения местоположения, то использование алгоритмов нейронных сетей позволит добиться желаемых показателей точности при меньшем количестве точек доступа (базовых станций).

Библиография

1. Андреев Р.А., Остроумов С.И., Федоров А.С. Методы позиционирования в сетях Wi-Fi // Экономика и качество систем связи. 2021. №3 (21). С 50-63
2. Кокорева Е.В., Костюкович А.Е., Дошинский И.В. Оценка погрешности измерений местонахождения абонента в сети Wi-Fi // Программные системы и вычислительные методы. 2019. № 4. С.30-38. DOI: 10.7256/2454-0714.2019.4.31316 URL: https://e-notabene.ru/itmag/article_31316.html
3. Kokoreva, E.V., Shurygina, K.I. Bragin, A.S. Impact of Wi-Fi network coverage planning on the logistics objects location accuracy // XV International Scientific Conference on Precision

- Agriculture and Agricultural Machinery Industry "State and Prospects for the Development of Agribusiness-INTERAGROMASH 2022". 2022. Vol. 363.
4. Kokoreva, E.V. & Shurygina, K.I. An Assessment of the Local Positioning System Effectiveness // Lecture Notes in Networks and Systems. 2022. Vol. 246. Pp. 436–443.
 5. Kokoreva E., Kostyukovich A., Shurygina K., Doshchinsky I. Experimental Study of the Positioning System in the Centralized Wi-Fi Network // Lecture Notes on Data Engineering and Communications Technologies. 2022. Vol. 107. Pp. 346–357.
 6. Киреев А.В., Фокин Г.А. Оценка точности локального позиционирования мобильных устройств с помощью радиокарт и инерциальной навигационной системы // Труды учебных заведений связи. 2017. №4. С. 54-62.
 7. Кучин И.Ю., Иксанов Ш.Ш., Рождественский С.К., Коряков А.Н. Разработка системы позиционирования и контроля объектов с помощью беспроводной технологии Wi-Fi // Научный вестник Новосибирского государственного технического университета. 2015. №3 (60). С. 130-146.
 8. Садовникова Н.А., Шмойлова Р.А. Анализ временных рядов и прогнозирование / М.: Евразийский открытый институт, 2024.
 9. Ахметханов Р.С., Дубинин Е.Ф., Куксова В.И. Анализ временных рядов в диагностике технических систем // Машиностроение и инженерное образование. 2013. № 2. С. 11–20.
 10. Chen W., Hussain W., Cauteruccio F., Zhang X. Deep Learning for Financial Time Series Prediction: A State-of-the-Art Review of Standalone and Hybrid Models // Computer Modeling in Engineering & Sciences. 2024. Vol. 139. № 1. Pp. 187–224.
 11. Ежов А.А., Шумский С.А. Нейрокомпьютинг и его применения в экономике и бизнесе / М.: ИНТУИТ, 2016.
 12. Karakida, R., Takase, T. Optimal layer selection for latent data augmentation // Neural Networks. 2024. Vol. 181.
 13. Sandnes, A.T., Grimstad, B., & Kolbjørnsen, O. Multi-task neural networks by learned contextual inputs // Neural Networks. 2024. Vol. 179.
 14. Хайкин С. Нейронные сети. Полный курс / М.: Вильямс, 2019.
 15. Zhou, X., You, Zh., Sun, W., Zhao, D., Yan, Sh. Fractional-order stochastic gradient descent method with momentum and energy for deep neural networks // Neural Networks. 2024. Vol. 181.
 16. Лизнева Ю.С. Исследование трафика ОКС N 7 и разработка методики его прогнозирования / Новосибирск: СибГУТИ, 2008.
 17. Marshoodulla S.Z, Saha G. A survey of data mining methodologies in the environment of IoT and its variants // Journal of Network and Computer Applications. 2024. Vol. 228.
 18. Zhou X., Du H., Xue Sh., Ma Zh. Recent advances in data mining and machine learning for enhanced building energy management // Energy. 2024. Vol. 307.
 19. Иванько А.Ф., Иванько М.А., Сизова Ю.А. нейронные сети: общие технологические характеристики // Научное обозрение. Технические науки. 2019. № 2. С. 17-23.
 20. Sevilla-Salcedo C., Gallardo-Antolín A., Gómez-Verdejo V., Parrado-Hernández E. Bayesian learning of feature spaces for multitask regression // Neural Networks. 2024. Vol. 179.

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

В статье рассматривается проблема повышения точности определения местоположения в существующих Wi-Fi сетях с помощью алгоритмов нейронных сетей. Объектом

исследования является типичная университетская сеть Wi-Fi, построенная без частотно-территориального планирования. Цель работы — оценка потенциальных возможностей использования нейронных сетей для улучшения точности позиционирования в подобных сетях.

Методология исследования включает проведение экспериментальных измерений уровня мощности сигнала (RSSI) по всей территории здания, использование статистической обработки данных и применение алгоритмов нейронных сетей для повышения точности определения местоположения. Экспериментальные данные собирались с помощью нескольких мобильных устройств в различных точках, после чего проводилась нормировка данных и обучение нейронной сети с использованием алгоритма Левенберга-Марквардта.

Тема исследования актуальна, так как с развитием технологий Wi-Fi возникает потребность в более точных и экономичных методах определения местоположения в беспроводных сетях. Применение нейронных сетей в этой области позволяет эффективно использовать существующую инфраструктуру без значительных дополнительных затрат, что делает данный подход особенно ценным для общественных учреждений, таких как университеты.

Научная новизна статьи заключается в применении нейронных сетей для повышения точности позиционирования в неоптимально построенных Wi-Fi сетях. Исследование показало, что даже в условиях отсутствия частотно-территориального планирования можно добиться удовлетворительных результатов по точности определения местоположения, что открывает новые возможности для использования существующей инфраструктуры.

Статья структурирована логично и последовательно. Введение содержит обоснование актуальности проблемы и цель исследования. Описание методологии детализировано и предоставляет достаточно информации для повторения экспериментов. Анализ данных и выводы представлены наглядно, с использованием графиков и таблиц, что способствует лучшему восприятию результатов. Стил изложения научный, но доступный, что делает статью интересной как для специалистов, так и для широкой читательской аудитории.

Авторы делают вывод, что использование нейронных сетей позволяет повысить точность определения местоположения в Wi-Fi сетях, построенных без учета частотно-территориального планирования. Полученные результаты показывают перспективность применения подобных методов в реальных условиях. Работа будет интересна специалистам в области беспроводных технологий, разработчикам систем позиционирования, а также исследователям, занимающимся применением методов машинного обучения в задачах обработки сигналов.

Статья выполнена на высоком уровне и имеет существенную научную ценность. Материал изложен последовательно, результаты подтверждаются экспериментальными данными. Авторы представили детальный анализ, подтвержденный статистическими методами и визуализированный в виде графиков и таблиц, что подчеркивает достоверность выводов. Работа демонстрирует как теоретическую значимость, так и практическую применимость предложенного подхода, особенно в контексте оптимизации существующих сетей Wi-Fi для задач позиционирования.

Дальнейшее развитие данного направления может включать исследование других алгоритмов машинного обучения, их сравнительный анализ с используемыми методами, а также возможное применение гибридных моделей для повышения точности и устойчивости позиционирования. Кроме того, исследование может быть расширено для различных типов сетей и других сценариев использования, что позволит еще глубже оценить потенциал нейронных сетей в задачах позиционирования.

Рекомендую принять статью к публикации, так как она вносит значительный вклад в

развитие технологий позиционирования в беспроводных сетях и демонстрирует перспективность применения нейронных сетей для решения подобных задач.

Программные системы и вычислительные методы

Правильная ссылка на статью:

Дагаев А.Е., Попов Д.И. Сравнение автоматического обобщения текстов на русском языке // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.69474 EDN: CSFMFC URL: https://nbpublish.com/library_read_article.php?id=69474

Сравнение автоматического обобщения текстов на русском языке

Дагаев Александр Евгеньевич

аспирант, кафедра «Информатика и информационные технологии», Московский политехнический университет

107023, Россия, г. Москва, ул. Большая Семёновская, 38

✉ alejaandro@bk.ru



Попов Дмитрий Иванович

доктор технических наук

Сочинский государственный университет, профессор кафедры Информационных технологий и математики

354000, Россия, Краснодарский край, г. Сочи, ул. Пластунская, 94

✉ damitry.popov@gmail.com



[Статья из рубрики "Системный анализ, поиск, анализ и фильтрация информации"](#)

DOI:

10.7256/2454-0714.2024.4.69474

EDN:

CSFMFC

Дата направления статьи в редакцию:

29-12-2023

Дата публикации:

07-11-2024

Аннотация: Предметом исследования в данной статье является обобщение текстов на русском языке с использованием моделей искусственного интеллекта. В частности, авторы сравнивают популярные модели GigaChat, YaGPT2, ChatGPT-3.5, ChatGPT-4, Bard, Bing AI и YouChat и проводят сравнительное исследование их работы на текстах

русского языка. В качестве исходных материалов для последующего обобщения в статье берутся наборы данных для русского языка, такие как Gazeta, XL-Sum и WikiLingua, а также для сравнения эффективности обобщения были взяты дополнительные наборы данных на английском языке CNN Dailymail и XSum. В статье применяются показатели: ROUGE, BLEU score, BERTScore, METEOR и BLEURT для оценки обобщения текстов. данной статье в качестве метода исследования используется сравнительный анализ данных, полученных в ходе автоматического обобщения с помощью моделей искусственного интеллекта. Научная новизна исследования заключается в проведении сравнительного анализа качества автоматического обобщения текстов на русском и английском языках с использованием различных нейросетевых моделей обработки естественного языка. Авторы исследования привлекли внимание к новым моделям GigaChat, YaGPT2, ChatGPT-3.5, ChatGPT-4, Bard, Bing AI и YouChat, рассматривая и анализируя их эффективность в задаче обобщения текста. Итоги обобщения на русском языке показывают, что YouChat демонстрирует самые высокие результаты по совокупности оценок, подчеркивая эффективность модели в обработке и генерации текста с более точным воспроизведением ключевых элементов содержания. В отличие от YouChat, модель Bard показала наихудшие результаты, представляя собой модель с наименьшей способностью к генерации связного и релевантного текста.

Ключевые слова:

обработка естественного языка, суммаризация текста, GigaChat, YaGPT2, ChatGPT-3, ChatGPT-4, Bard, Bing AI, YouChat, сжатие текста

Введение

Реферирование текста – это важное направление в области обработки естественного языка, которое имеет немалое значение для большого количества задач. Для обобщения текста модель искусственного интеллекта должна быть способна создавать связный и релевантный контент, одновременно сжимая главную информацию в более короткую форму вне зависимости от предметных областей. С помощью обобщения текстов можно определить и сопоставить качество работы нейросетей. В статье будет проведено сравнительное исследование популярных моделей искусственного интеллекта на текстах русского языка.

Связанные работы

Оценкой качества языковых моделей в последнее время занимались достаточно активно, но в основном исследования направлены на обработку английского языка, как международного. С русским языком, как основным, сравнения между GigaChat, YaGPT2, ChatGPT-3.5, ChatGPT-4, Bard, Bing AI и YouChat найдено не было.

В работе [\[6\]](#) было исследовано рекурсивное обобщение через GPT-3.5, а также методы отбора существенного контента для обобщения. В работе [\[7\]](#) указывают, что модели GPT с трудом идентифицирует важную информацию и более подвержены ошибкам при обобщении длинных текстовых форм. Исследование качества генерации с применением моделей GPT и дальнейший анализ показали, что качество для языков с высокой лингвистической частотностью выше, чем с низкой [\[8\]](#). А в работе [\[9\]](#) отмечена слабая производительность GPT при работе с русским языком в мультязычном наборе данных. В недавних исследованиях [\[1\]\[2\]](#) показывается, что качество обобщения новостей с

помощью больших языковых моделей находится на сопоставимом с созданным человеком уровне.

Наборы данных

В качестве наборов данных для русского языка использованы:

Gazeta [\[3\]](#). Набор данных содержит 63435 новостей, размещенных на сайте gazeta.ru.

XL-Sum [\[12\]](#). В наборе представлены 1,35 миллионов аннотированных пар статей BBC на разных языках, в том числе на русском – 77803.

WikiLingua [\[16\]](#). Многоязычный набор данных, созданный для оценки задачи обобщения. Материалы включают в себя статьи на 18 языках из WikiHow. На русском языке собраны 52928 статей.

Для сравнения эффективности обобщения были взяты дополнительные наборы данных на английском языке:

CNN Dailymail [\[11\]](#). В набор включены новостные статьи CNN за период с апреля 2007 по апрель 2015 года и Daily Mail с июня 2010 по апрель 2015 года. Суммарное количество составляет 311672 статьи.

XSum [\[10\]](#). Набор состоит из 226711 статей BBC за период с 2010 по 2017 год.

Из всех перечисленных наборов случайным образом выделены 100 оригинальных текстов, которые были унифицированы по длине в 1024 токена.

Показатели оценки

В данной работе использовались показатели оценки качества текстов ROUGE [\[4\]](#), BLEU score [\[5\]](#), BERTScore [\[13\]](#), METEOR [\[14\]](#) и BLEURT [\[15\]](#). ROUGE используется для оценки качества текстов, созданных машинами. Он анализирует сходство между искусственно созданным текстом и эталонным. ROUGE определяет точность и полноту информации. Он может использоваться для оценки различных задач, включая обобщение текста и машинный перевод.

Среди разных вариантов ROUGE в работе были использованы:

ROUGE-1. Этот показатель вычисляет перекрытие униграмм (отдельных слов) между машинным и эталонным текстом. Это помогает оценить точность машинного перевода или обобщения текста.

ROUGE-2, который работает с биграммами, то есть с парами слов.

ROUGE-L, который анализирует длинные фразы в тексте и в результате измеряет сходство между сгенерированным текстом и ссылочным текстом с учетом последовательностей слов, то есть оценивает длину самой длинной общей подпоследовательности

Каждый из этих вариантов ROUGE позволяет по-разному оценить качество сгенерированного машиной текста. ROUGE-1 и ROUGE-2 фокусируются на перекрытии на уровне слов и биграмм, в то время как ROUGE-L рассматривает структуру и порядок слов в текстах.

BLEU score [\[5\]](#) представляет собой показатель, используемый для измерения качества машинного текста, в частности, в контексте машинного перевода и обобщения текста. Первоначально он был создан для оценки качества машинного перевода, однако сейчас применяется для многих других задач в области NLP. BLEU score оценивает сходство между машинно-созданным текстом с одним или более эталонными текстами, написанными людьми. Это достигается путем сравнения n-грамм в машинном тексте с n-граммами в эталонных текстах.

BERTScore метрика, которая оценивает схожесть между двумя текстами, используя векторные представления, полученные с использованием модели BERT. Оценка BERTScore хорошо коррелируется с суждениями человека и обеспечивает более высокую эффективность выбора модели, чем существующие показатели, кроме этого она более устойчива к сложным примерам по сравнению с существующими метриками [\[13\]](#). В статье используется F1 Score, которая рассчитывается как среднее гармоническое значение точности и запоминания. Это обеспечивает сбалансированный показатель, учитывающий как ложноположительные, так и ложноотрицательные результаты.

METEOR [\[14\]](#). Метрика METEOR часто используется для оценки качества машинного перевода и предоставляет более подробную информацию, чем показатель BLEU. При этом учитывается не только точность и запоминаемость отдельных слов, но также рассматриваются основы слов, синонимы и порядок слов. Всё это сделано в целях обеспечения более целостной оценки качества.

BLEURT [\[15\]](#) представляет собой обученную метрику, основанную на BERT и RemBERT. В качестве входных данных используются пары текстов: кандидат-эталонный текст, и выводится оценка, показывающая, насколько хорошо кандидат владеет языком и передает основной смысл текста.

Результаты

Результаты по вышеуказанным показателям для набора данных Gazeta представлены в таблице 1.

Таблица 1 – Результаты на наборе Gazeta

	ROUGE-1	ROUGE-2	ROUGE-L	BLEU	BERTScore	METEOR	BLEURT
GigaChat	0,17	0,09	0,17	6,71	0,71	0,16	0,14
YaGPT2	0,11	0,04	0,11	6,05	0,70	0,09	0,00
ChatGPT-3.5	0,29	0,12	0,28	5,88	0,72	0,19	0,14
ChatGPT-4	0,27	0,10	0,25	6,12	0,71	0,16	0,04
Bard	0,33	0,18	0,32	4,43	0,71	0,26	-0,06
Bing AI	0,33	0,15	0,31	5,09	0,72	0,23	0,03
YouChat	0,33	0,19	0,32	9,60	0,72	0,24	0,22

Результаты для набора XL-Sum приведены в таблице 2.

Таблица 2 – Результаты на наборе XL-Sum

	ROUGE-1	ROUGE-2	ROUGE-L	BLEU	BERTScore	METEOR	BLEURT
GigaChat	0,15	0,05	0,14	5,69	0,68	0,09	-0,03
YaGPT2	0,10	0,02	0,10	5,98	0,66	0,04	-0,07
ChatGPT-	0,24	0,10	0,23	6,60	0,70	0,15	0,20

Model	0,24	0,10	0,24	4,36	0,69	0,20	-0,09
ChatGPT-4	0,24	0,10	0,24	4,36	0,69	0,20	-0,09
Bard	0,24	0,10	0,23	4,48	0,69	0,20	-0,13
Bing AI	0,32	0,17	0,30	4,41	0,71	0,21	-0,09
YouChat	0,38	0,23	0,36	5,94	0,73	0,26	0,12

По набору WikiLingua результаты приведены в таблице 3.

Таблица 3 – Результаты на наборе WikiLingua

	ROUGE-1	ROUGE-2	ROUGE-L	BLEU	BERTScore	METEOR	BLEURT
GigaChat	0,33	0,16	0,32	5,30	0,73	0,23	0,16
YaGPT2	0,20	0,05	0,19	5,07	0,72	0,09	0,12
ChatGPT-3.5	0,27	0,09	0,26	5,11	0,71	0,17	0,00
ChatGPT-4	0,23	0,05	0,21	4,54	0,70	0,14	0,04
Bard	0,34	0,17	0,33	5,04	0,75	0,24	0,08
Bing AI	0,41	0,24	0,39	4,51	0,75	0,29	0,09
YouChat	0,56	0,36	0,54	4,70	0,83	0,47	0,14

Для подсчета общей оценки каждому показателю был обозначен индивидуальный вес. Веса были распределены исходя из специфики задачи обобщения текстов, где больший вес выделен на семантическое сходство, комбинацию семантического и структурного сходства, а также на степень сжатия. Формула расчета:

$$S = 0,05 * M_1 + 0,05 * M_2 + 0,05 * M_3 + 0,1 * M_4 + 0,20 * M_5 + 0,20 * M_6 + 0,20 * M_7 + 0,15 * C \quad (1)$$

Где S – общая оценка,

M_1 – ROUGE-1;

M_2 – ROUGE-2;

M_3 – ROUGE-L;

M_4 – BLEU;

M_5 – BERTScore;

M_6 – METEOR;

M_7 – BLEURT;

C – Степень сжатие текста (%).

На рисунке 1 изображена диаграмма общих оценок для всех использованных наборов данных русского языка.

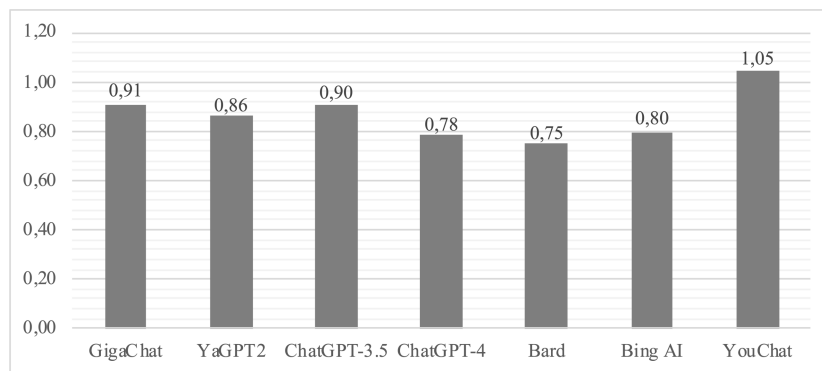


Рисунок 1 – Диаграмма общих оценок для наборов данных русского языка

YouChat показал самые высокие результаты на всех наборах данных по совокупности показателей. Тем самым подчеркивается эффективность модели в обработке и генерации текста, а также ее способность к точному воспроизведению ключевых элементов содержания.

Bard, в отличие от других моделей, выполняет генерацию связного и контекстуально релевантного текста худшего качества, что приводит к неудовлетворительным результатам при оценке сходства, обобщения и других задач обработки естественного языка. Также более низкие показатели могут указывать на трудности в восприятии тонкостей языка, что приводит к таким проблемам, как нерелевантная информация, отсутствие смысловой последовательности и неточности в воспроизведении основного контекста.

GigaChat лучше подошел к задаче обобщения по сравнению с ChatGPT-3.5, но в целом результаты находятся на сопоставимом по качеству уровне.

GigaChat показал воспроизведение контекста точнее YaGPT2 и продемонстрировал более осмысленную генерацию текста с общей повышенной способностью к обобщению.

Bard показал наименьшую итоговую оценку на наборах русского языка.

Результаты сжатия между входными и выходными данными отражены на рисунке 2.

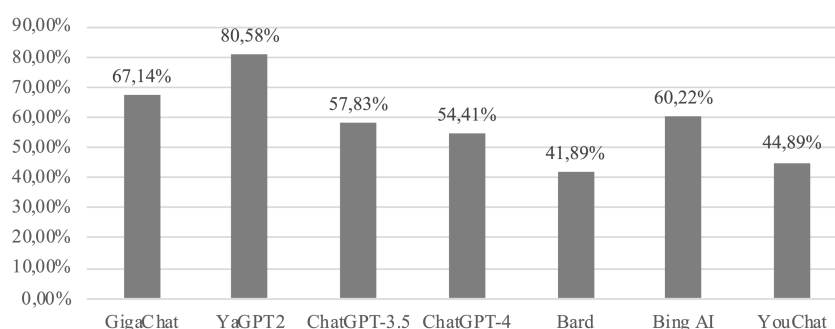


Рисунок 2 – Обобщенный график сжатия на наборах данных русского языка

Максимальное сжатие показал YaGPT2 – 80,58%, далее GigaChat – 67,14%, Bing AI – 60,22%, ChatGPT-3.5 – 57,83%, ChatGPT-4 – 54,41%, YouChat – 44,89%, а наиболее низкие результаты по сжатию получились у Bard – 41,89%.

Проведя обобщение на английском языке для набора данных CNN Dailymail были зафиксированы показатели, которые указаны в таблице 4. Следует отметить, что YaGPT2 не работает с текстами на английском языке, по этой причине он был исключен из

списка моделей для последующего анализа.

Таблица 4 – Результаты на наборе CNN Dailymail

	ROUGE-1	ROUGE-2	ROUGE-L	BLEU	BERTScore	METEOR	BLEURT
GigaChat	0,32	0,16	0,30	6,27	0,72	0,16	-0,29
ChatGPT-3.5	0,28	0,11	0,25	5,35	0,71	0,13	-0,34
ChatGPT-4	0,30	0,09	0,27	5,02	0,71	0,16	-0,37
Bard	0,32	0,17	0,30	7,04	0,72	0,16	-0,55
Bing AI	0,33	0,15	0,31	5,63	0,72	0,16	-0,38
YouChat	0,39	0,19	0,36	5,48	0,73	0,22	-0,28

Результаты по набору XSum отражены в таблице 5

Таблица 5 – Результаты на наборе XSum

	ROUGE-1	ROUGE-2	ROUGE-L	BLEU	BERTScore	METEOR	BLEURT
GigaChat	0,38	0,22	0,36	6,01	0,74	0,19	-0,27
ChatGPT-3.5	0,34	0,14	0,31	5,09	0,72	0,18	-0,28
ChatGPT-4	0,34	0,11	0,31	4,58	0,71	0,18	-0,36
Bard	0,36	0,20	0,35	4,31	0,73	0,15	-0,36
Bing AI	0,42	0,22	0,40	4,45	0,74	0,25	-0,36
YouChat	0,42	0,21	0,39	5,47	0,73	0,25	-0,17

Общая оценка на данных английского языка приведена на рисунке 3. Результаты показывают, что качество выполненного обобщения имеет зависимость от исходного языка.

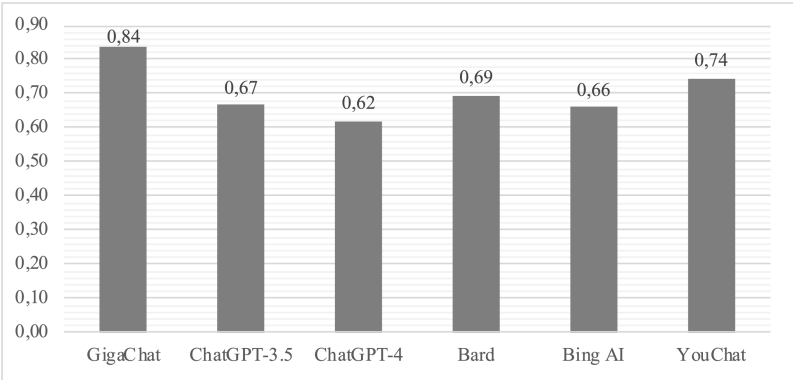


Рисунок 3 – Диаграмма общих оценок для наборов данных английского языка

GigaChat показал самый высокий балл среди других рассмотренных моделей.

ChatGPT-4 в наборах на английском языке демонстрируют самое низкое качество, а также сжимает текст меньше остальных моделей.

Результаты сжатия между входными и выходными данными представлены на рисунке 4.

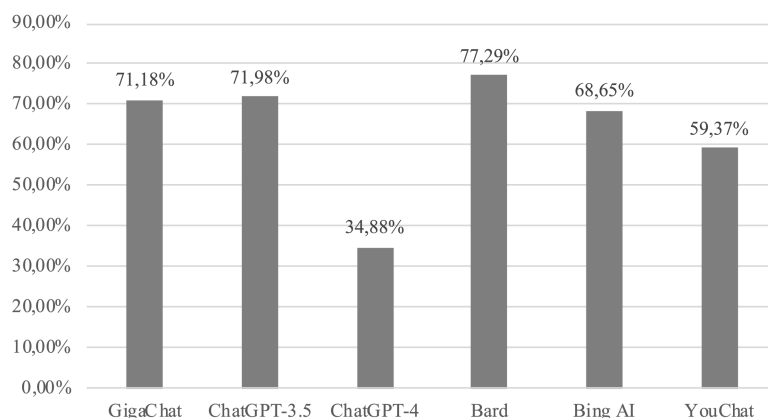


Рисунок 4 – Обобщенный график сжатия на наборах данных английского языка

Максимальное сжатие на англоязычных наборах данных показал Bard – 77,29%, ChatGPT-3.5 – 71,98%, GigaChat – 71,18%, Bing AI – 68,65%, YouChat – 59,37%, самые низкие показатели у ChatGPT-4 – 34,88%

В целом, все представленные модели показали приемлемые результаты по выбранным метрикам. Это свидетельствует о том, что их можно применять для решения задач обобщения текста. Однако стоит отметить, что исходя из разных возможностей моделей могут быть отличия в определенных сценариях использования, связанных с размером модели, типом задачи или языковыми особенностями, что требует дополнительного исследования.

Заключение

В данной работе было проведено сравнение качества автоматического обобщения текстов с помощью различных нейросетей в части обработки естественного языка, таких как GigaChat, YaGPT2, ChatGPT-3.5, ChatGPT-4, Bard, Bing AI и YouChat. Для этого был взят и предобработан набор данных, содержащий тексты на русском и для сравнения на английском языках. Затем выполнено обобщение одинакового списка текстов на каждой модели. После этого получены результаты по показателям ROUGE^[4], BLEU score^[5], BERTScore^[13], METEOR^[14] и BLEURT^[15], которые сравнивали оригинальные и сгенерированные в ходе автоматического реферирования тексты. Были также получены результаты общей оценки между всеми показателями, где каждому показателю был выделен вес, исходя из важности для задачи обобщения текста.

Данные, полученные в ходе сравнения, будут способствовать более глубокому пониманию рассматриваемых моделей, помогая делать выбор при применении искусственного интеллекта для задач обобщения текстов в качестве основы для будущих разработок.

В дальнейшем планируется исследовать работу по обработке текстов между моделями с различными параметрами настроек.

Библиография

1. Goyal T., Li J. J., Durrett G. News summarization and evaluation in the era of gpt-3 //arXiv preprint arXiv:2209.12356. – 2022.
2. Zhang T. et al. Benchmarking large language models for news summarization //arXiv preprint arXiv:2301.13848. – 2023.
3. Gusev I. Dataset for automatic summarization of Russian news //Artificial Intelligence

- and Natural Language: 9th Conference, AINL 2020, Helsinki, Finland, October 7–9, 2020, Proceedings 9. – Springer International Publishing, 2020. – С. 122-134.
4. Lin C. Y. Rouge: A package for automatic evaluation of summaries //Text summarization branches out. – 2004. – С. 74-81.
5. Post M. A call for clarity in reporting BLEU scores //arXiv preprint arXiv:1804.08771. – 2018.
6. Bhaskar A., Fabbri A., Durrett G. Prompted opinion summarization with GPT-3.5 //Findings of the Association for Computational Linguistics: ACL 2023. – 2023. – С. 9282-9300.
7. Tang L. et al. Evaluating large language models on medical evidence summarization //npj Digital Medicine. – 2023. – Т. 6. – №. 1. – С. 158.
8. Hendy A. et al. How good are gpt models at machine translation? a comprehensive evaluation //arXiv preprint arXiv:2302.09210. – 2023.
9. Jiao W. et al. Is ChatGPT a good translator? Yes with GPT-4 as the engine //arXiv preprint arXiv:2301.08745. – 2023.
10. Narayan S., Cohen S. B., Lapata M. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization //arXiv preprint arXiv:1808.08745. – 2018.
11. Nallapati R. et al. Abstractive text summarization using sequence-to-sequence rnns and beyond //arXiv preprint arXiv:1602.06023. – 2016.
12. Hasan T. et al. XL-sum: Large-scale multilingual abstractive summarization for 44 languages //arXiv preprint arXiv:2106.13822. – 2021.
13. Zhang T. et al. Bertscore: Evaluating text generation with bert //arXiv preprint arXiv:1904.09675. – 2019.
14. Banerjee S., Lavie A. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments //Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization. – 2005. – С. 65-72.
15. Sellam T., Das D., Parikh A. P. BLEURT: Learning robust metrics for text generation //arXiv preprint arXiv:2004.04696. – 2020.
16. Ladhak F. et al. WikiLingua: A new benchmark dataset for cross-lingual abstractive summarization //arXiv preprint arXiv:2010.03093. – 2020.

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Предметом исследования в данной статье является сравнение качества автоматического обобщения (реферирования) текстов на русском языке с использованием различных моделей искусственного интеллекта. Методология исследования включает отбор и предобработку наборов текстовых данных на русском и английском языках, генерацию рефератов этих текстов различными моделями ИИ, а также оценку качества полученных рефератов с помощью стандартных метрик ROUGE, BLEU, BERTscore, METEOR и BLEURT. Используемые в работе метрики (ROUGE, BLEU, BERTscore, METEOR и BLEURT) формируют комплексный подход к оценке качества автоматического реферирования текстов, учитывающий различные аспекты: точность передачи отдельных слов и словосочетаний (ROUGE, BLEU), семантическое сходство и порядок слов в предложениях (BERTscore, METEOR), общую передачу смысла исходного текста (BLEURT). Каждая метрика имеет свои преимущества и недостатки. В целом их совместное использование позволяет получить наиболее объективную оценку и сравнить

эффективность разных моделей автоматического реферирования текстов. При этом результаты отдельных метрик могут несколько расходиться, что связано с учетом ими различных лингвистических факторов. Тема является актуальной, поскольку задача автоматического реферирования текстов активно исследуется в области обработки естественного языка и имеет множество практических применений. Сравнение эффективности различных подходов для русского языка ранее не проводилось. Научная новизна работы заключается в том, что впервые проведено сравнительное исследование качества обобщения текстов на русском языке с использованием ряда популярных моделей искусственного интеллекта. Стиль изложения научный, текст структурирован, основные разделы соответствуют логике исследования. Содержание достаточно полно раскрывает заявленную тему. Библиография актуальна, охватывает последние работы в данной предметной области. Результаты исследования представляют интерес для специалистов в области компьютерной лингвистики и обработки естественного языка. Могут найти применение при выборе оптимальных моделей ИИ для решения задач автоматического реферирования текстов. Таким образом, статья актуальна, обладает научной новизной и может быть рекомендована к публикации. Рекомендации для дальнейших исследований:

1. Расширение списка сравниваемых моделей автоматического реферирования за счет наиболее передовых и популярных архитектур.
2. Более подробный анализ влияния конфигурации моделей (размер, объем обучающих данных и т.д.) на качество реферирования.
3. Исследование особенностей применения моделей для текстов из разных предметных областей и на разных языках.
4. Разработка комбинированных подходов с использованием нескольких моделей на разных этапах процесса реферирования.
5. Сравнение с рефератами, составленными экспертами, для выявления недостатков существующих алгоритмов.

Проведение таких исследований позволит лучше понять возможности и ограничения современных моделей автоматического реферирования текстов.

Программные системы и вычислительные методы

Правильная ссылка на статью:

Булгаков В.Д., Гвоздевский И.Н. Модель и алгоритм консенсуса Proof of Performance // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.71119 EDN: NAGMFW URL: https://nbpublish.com/library_read_article.php?id=71119

Модель и алгоритм консенсуса Proof of Performance

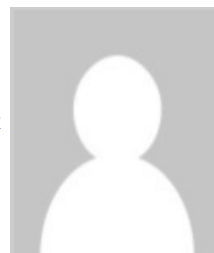
Булгаков Владислав Дмитриевич

ORCID: 0009-0006-2056-6169

аспирант; кафедра Программное обеспечение вычислительной техники и автоматизированных систем; Белгородский государственный технологический университет им. В.Г. Шухова

308012, Россия, Белгородская область, г. Белгород, ул. Костюкова, 46

✉ bulgakowlad@yandex.ru



Гвоздевский Игорь Николаевич

ORCID: 0000-0002-3235-3869

кандидат технических наук

доцент; кафедра Программное обеспечение вычислительной техники и автоматизированных систем; Белгородский государственный технологический университет им. В.Г. Шухова

308012, Россия, Белгородская область, г. Белгород, ул. Костюкова, 36, оф. 421

✉ Gvozdevskiy.in@bstu.ru



[Статья из рубрики "Models and methods of information security management"](#)

DOI:

10.7256/2454-0714.2024.4.71119

EDN:

NAGMFW

Дата направления статьи в редакцию:

25-06-2024

Дата публикации:

30-11-2024

Аннотация: В статье исследуется принцип работы модели Proof of Performance (PoP), основанной на алгоритме консенсуса, поддерживающем функции горизонтального шардинга. Модель PoP вносит изменения в традиционную структуру блоков,

используемую в алгоритмах Proof of Stake и сетях на базе ядра Tendermint. Горизонтальный шардинг позволяет распределять транзакции между несколькими узлами (шардами), что значительно увеличивает пропускную способность сети. Основная цель исследования — изучение способов повышения эффективности и масштабируемости блокчейн-сетей через динамическое распределение транзакций и адаптивное управление узлами. Важным аспектом является определение параметров и изменяемых характеристик узлов, таких как производительность и надежность для равномерного и справедливого распределения нагрузки внутри сети. Это обеспечивает адаптацию системы к изменяющимся условиям нагрузки. В работе используются аналитические и формальные методы для описания структуры блока, механизма распределения транзакций и системы наказаний и поощрений для шардов. Исследование представляет собой инновационный подход к управлению блокчейн-сетями, акцентируя внимание на производительности узлов. Модель PoP с горизонтальным шардингом обеспечивает более высокую пропускную способность и масштабируемость сети по сравнению с традиционными алгоритмами консенсуса. Предложена система динамического распределения нагрузки и адаптивного изменения весов узлов на основе их производительности, что способствует повышению эффективности и надежности сети. В результате исследования доказано, что модель Proof of Performance значительно увеличивает скорость обработки транзакций и общую производительность блокчейн-сети. Примеры применения подтверждают эффективность модели в различных типах сетей, таких как DeFi-платформы, системы управления цепочками поставок и IoT-сети. Модель PoP стимулирует узлы к поддержанию высокой производительности, обеспечивая справедливое распределение нагрузки и повышая общую устойчивость сети.

Ключевые слова:

Модель консенсуса, Алгоритм консенсуса, Шард, Блок, Доказательство производительности, Доказательство ставки, Горизонтальный шардинг, Производительность, Распределение нагрузки, Блокчейн

Введение

Актуальность проблемы повышения производительности и масштабируемости блокчейн-сетей обусловлена растущими требованиями к обработке транзакций в высоконагруженных системах, таких как финансовые технологии, системы Интернета вещей (IoT) и децентрализованные приложения (dApps). Существующие алгоритмы консенсуса, включая Proof of Stake (PoS) на базе ядра Tendermint демонстрируют ограниченную пропускную способность, обусловленную высокой нагрузкой и ограниченными ресурсами производительности узлов-валидаторов сети. Эти ограничения создают узкие места в функционировании сетей и снижают их устойчивость к изменениям нагрузки^[1].

Цель данного исследования — разработка и анализ модели Proof of Performance (PoP), которая направлена на повышение эффективности и масштабируемости блокчейн-сетей посредством внедрения технологии горизонтального шардинга. В основе PoP лежит идея равновесного распределения транзакций между узлами, что позволяет снизить нагрузку на отдельные узлы-валидаторы и повысить общую производительность сети.

Для достижения поставленной цели решаются следующие задачи:

- Разработка модифицированной структуры блока, обеспечивающей поддержку шардинга.
- Построение математической модели распределения нагрузки и расчёта производительности узлов.
- Экспериментальная проверка корректности работы модели PoP в развернутой тестовой сети.

Научная новизна работы заключается в предложении и описании новейшего алгоритма Proof of Performance, сочетающего адаптивное распределение нагрузки и учет производительности узлов, что позволяет существенно увеличить пропускную способность сети и её устойчивость к изменениям нагрузки.

Представленная работа закладывает основу для дальнейших исследований, направленных на интеграцию описанного алгоритма консенсуса в существующие блокчейн-системы и развитие технологий децентрализованного управления нагрузкой.

Описание модели Proof of Performance

В алгоритме консенсуса Proof of Stake узел, предлагающий блок (Block proposer) получает из пула транзакций (Transaction pool/mempool) определенное количество транзакций для обработки, чтобы в последующем опубликовать их в блоке, а, соответственно, в сети. Такой подход ограничивает количество транзакций, которые могут быть обработаны сетью за определенный момент времени, а также повышает нагрузку на узел, который является предлагающим в данной итерации создания блока. Более того, подход PoS к определению участника сети, который получит возможность добавить блок в сеть определяется количеством монет участника, что само по себе пагубно воздействует на философию децентрализации блокчейн-сети [\[2\]](#).

Внедрение технологии горизонтального шардинга позволит снять нагрузку с узла, предлагающего блок и распределить ее равномерно на все узлы, являющимися шардами, то есть на узлы, предоставляющие свои вычислительные мощности для процесса обработки транзакций в сети [\[3\]](#).

Далее представлена схема, которая описывает процесс обработки транзакций в алгоритмах консенсуса Proof of Stake (Рисунок 1) и Proof of Performance (Рисунок 2).

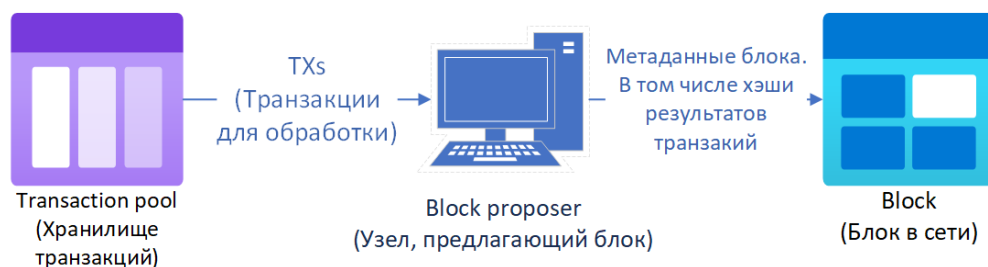


Рис.1 – Процесс обработки транзакций в алгоритме Proof of Stake

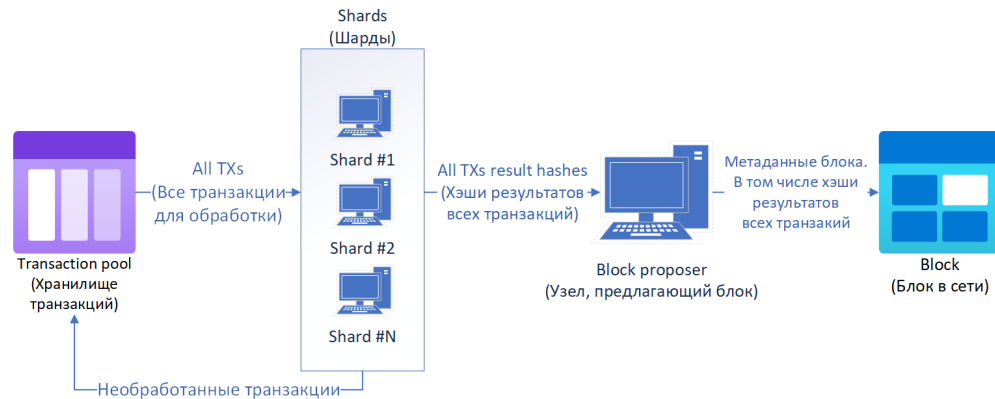


Рис.2 – Процесс обработки транзакций в алгоритме Proof of Performance

Основными преимуществами данного подхода являются:

- Единовременная обработка всех транзакций, находящихся в хранилище. В случае Proof of Stake, алгоритм не может прислать на обработку узлу количество транзакций, превышающее определенный порог, так как это пагубно скажется на производительности узла, предлагающего блок и может вызвать сбой в сети, в случае выхода этого узла из строя. В модели Proof of Performance алгоритм «достаёт» из хранилища все транзакции и распределяет их равномерно между шардами [\[4\]](#).
- Распределение нагрузки. Отсутствие диверсификации в процессе обработки транзакций является собой большую проблему в сфере блокчейна [\[5\]](#). Выход из строя узла, на котором лежит задача создания нового блока увеличивает время его создания, а в случае, если все узлы сети не справились с этой задачей, вызывает блокировку сети. Введение понятия шардов, описание их алгоритма работы и внедрение их в существующую систему позволит узлу, занятому процессом создания блоков не тратить свои вычислительные мощности на обработку транзакций. Ему предстоит лишь получить эти данные от шардов, проверить их и опубликовать в блоке.

Структура блока в модели Proof of Performance

Так как процесс учета и контроля шардинга представляет собой новую задачу, для децентрализованного хранения результатов ее решения необходимо модифицировать имеющуюся структуру блока. На данный момент, в сетях Proof of Stake преобладает следующая структура, состоящая из 15-ти полей:

- Block #N: Номер блока. Это уникальный идентификатор блока в блокчейне, указывающий на его позицию в цепочке.
- Version: Версия программного обеспечения или протокола, используемого для создания этого блока.
- ChainID: Идентификатор сети, к которой относится данный блок.
- Height: Высота блока. Это порядковый номер блока в блокчейне, начиная с генезис-блока (нулевого блока).
- Time: Временная метка, указывающая время создания блока.
- LastBlockID: Идентификатор предыдущего блока в цепочке. Это значение используется для связывания блоков в единую цепь.
- LastCommitHash: Хэш последнего подтвержденного блока. Это значение используется

для проверки корректности предыдущего блока.

- **DataHash:** Хэш данных блока. Это значение представляет собой криптографический хэш всех транзакций, включенных в блок.
- **ValidatorsHash:** Хэш валидаторов. Это значение представляет собой криптографический хэш списка валидаторов, участвующих в валидации блока.
- **NextValidatorsHash:** Хэш следующих валидаторов. Это значение представляет собой криптографический хэш списка валидаторов, которые будут участвовать в валидации следующего блока.
- **ConsensusHash:** Хэш консенсуса. Это значение представляет собой криптографический хэш параметров консенсусного алгоритма, используемого для достижения соглашения между валидаторами.
- **AppHash:** Хэш приложения. Это значение представляет собой криптографический хэш состояния приложения после выполнения всех транзакций в блоке.
- **LastResultsHash:** Хэш результатов последнего выполнения. Это значение представляет собой криптографический хэш результатов выполнения транзакций в предыдущем блоке.
- **EvidenceHash:** Хэш доказательств. Это значение представляет собой криптографический хэш всех доказательств, включенных в блок, которые могут быть использованы для выявления злонамеренных валидаторов.
- **ProposerAddress:** Адрес предлагающего блока. Это уникальный идентификатор участника сети, который предложил данный блок для добавления в цепочку.

Данная структура в виде таблицы, составляющей блок, продемонстрирована на рисунке 3.

Block #N
Version
ChainID
Heigh
Time
LastBlockID
LastCommitHash
DataHash
ValidatorsHash
NextValidatorsHash
ConsensusHash
AppHash
LastResultsHash
EvidenceHash
ProposerAddress

Рис.3 – Структура блока в алгоритме Proof of Stake

Так как данная структура нуждается в доработке и добавлении параметров для учета процесса шардинга, то структура блока в модели Proof of Performance будет включать в себя следующие дополнительные параметры:

- ShardsHash: Хэш списка шардов. Криптографический хэш, представляющий список всех узлов, которые являются шардами в сети.
- CurrentShardsHash: Хэш текущих шардов. Криптографический хэш, представляющий список шардов, участвовавших в обработке транзакций в данном блоке.
- NextShardsHash: Хэш следующих шардов. Криптографический хэш состояния шардов, которые будут участвовать в процессе обработки транзакций в следующем блоке.
- ShardsEvidenceHash: Хэш доказательств шардов. Криптографический хэш всех доказательств, включенных в блок, которые могут быть использованы для выявления злонамеренных шардов.
- DataAccordanceShardsHash: Хэш соответствия данных шардов. Криптографический хэш, обеспечивающий согласованность данных между разными шардами. Данное поле содержит хэш соответствия между шардами и обработанными ими транзакциями.

Согласно обновленной структуре, модель блока в алгоритме Proof of Performance, реализующим процесс горизонтального шардинга, будет выглядеть следующим образом:

Block #N
Version
ChainID
Heigh
Time
LastBlockID
LastCommitHash
DataHash
ShardsHash
CurrentShardsHash
NextShardsHash
ShardsEvidenceHash
DataAccordanceShardsHash
ValidatorsHash
NextValidatorsHash
ConsensusHash
AppHash
LastResultsHash
EvidenceHash
ProposerAddress

Рис.4 – Структура блока в алгоритме Proof of Performance

Эти нововведения позволяют сети учитывать вклад каждого шарда в обработку транзакций, обеспечивать безопасность данных и распределять нагрузку между узлами пропорционально их производительности.

Необходимость таких изменений обусловлена следующими факторами:

1. Проблемы текущей структуры блоков:

- Отсутствие поддержки горизонтального шардинга.
- Концентрация нагрузки на узле-валидаторе.
- Повышенная вероятность сбоя из-за перегрузки одного узла.

2. Преимущества модифицированной структуры:

- Равномерное распределение транзакций между узлами.
- Возможность адаптации к изменениям нагрузки.
- Повышение устойчивости сети за счёт децентрализации обработки данных.

Таким образом, предложенные изменения в структуре блока обеспечивают основу для эффективной реализации модели PoP и её дальнейшей интеграции в блокчейн-системы.

Распределение весов

Для того, чтобы обеспечить справедливую оценку и распределение нагрузки между шардами, алгоритм Proof of Performance берет на себя роль арбитра и обеспечивает процесс управления процедурами назначения, поощрения и наказания шардов в зависимости от их производительности.

В основе алгоритма Proof of Performance лежит система оценки производительности (Performance) каждого шарда.

Для определения производительности используются следующие параметры, влияющие на конечный результат вычислений веса шарда:

- Performance (P): Производительность шарда в обработке транзакций. Оценивается на основе успешности выполнения транзакций в текущем и предыдущих блоках.
- Weight (W): Вес шарда, определяющий уровень его конкурентной способности обрабатывать транзакции в следующем блоке.
- Penalty (Penalty): Штраф за несвоевременную или неуспешную обработку транзакций
- Reward (Reward): Поощрение за успешную обработку транзакций.

Оценка производительности P_i происходит по следующей формуле:

$$P_i = T_i / T_j$$

где T_i – общее количество транзакций, назначенных шарду, T_j – количество транзакций, успешно обработанных шардом;

Определение веса шарда является одной из главных задач алгоритма, так как неверно определенный вес может пагубно отразиться на производительности сети (например, если шард, обладающий небольшими вычислительными мощностями получит большой вес). Для определения веса (WS_{next}) используется следующая формула:

$$WS_{next} = WS_{current} * 1 + Reward_S - Penalty_S$$

где $WS_{current}$ – вес шарда в данном блоке, $Reward_S$ – коэффициент поощрения для шарда, а $Penalty_S$ – коэффициент штрафа (наказания) для шарда.

Коэффициенты поощрения и штрафа определяются следующим образом:

$$Reward_S = k_r * P_S$$

$$Penalty_S = k_p * (1 - P_S)$$

где k_r и k_p – константы, задающие влияние поощрений и штрафов на вес шарда.

После подсчета весов для следующей итерации, их необходимо нормализовать, чтобы сумма весов всех шардов не превышала 1, то есть 100%. Для этого используются следующие формулы:

$$W_{total} = \sum W^{next}$$

$$W_{snorm} = W_s / \theta,$$

где θ – сумма всех ненормализованных весов шардов, а W – нормализованный вес для шарда s .

Для распределения транзакций между шардами в следующем блоке используется взвешенный случайный выбор на основе обновленных нормализованных весов W_{snorm} . Общее количество транзакций делится между шардами пропорционально их весам на основе следующей формулы:

$$T_s^{next} = T * W_s^{norm}$$

где T_s^{next} – количество транзакций, назначаемых шарду в следующем блоке.

Работоспособность алгоритма

Алгоритм работы Proof of Performance будет состоять из следующих шагов:

1. Инициализация. На этом шаге устанавливаются начальные веса всех шардов. Данный шаг выполняется только на первой итерации работы алгоритма, то есть при создании генезис (нулевого) блока;
2. Сбор транзакций. Все входящие от пользователей сети транзакции собираются в общий пул (хранилище);
3. Распределение транзакций. Транзакции распределяются между шардами на основе текущих (или первоначальных) весов;
4. Обработка транзакций. Каждый шард обрабатывает назначенные ему транзакции;
5. Оценка производительности. Производительность каждого шарда оценивается по формуле $P_{[6]}$.
6. Обновление весов. Веса шардов обновляются с учетом поощрений и штрафов.

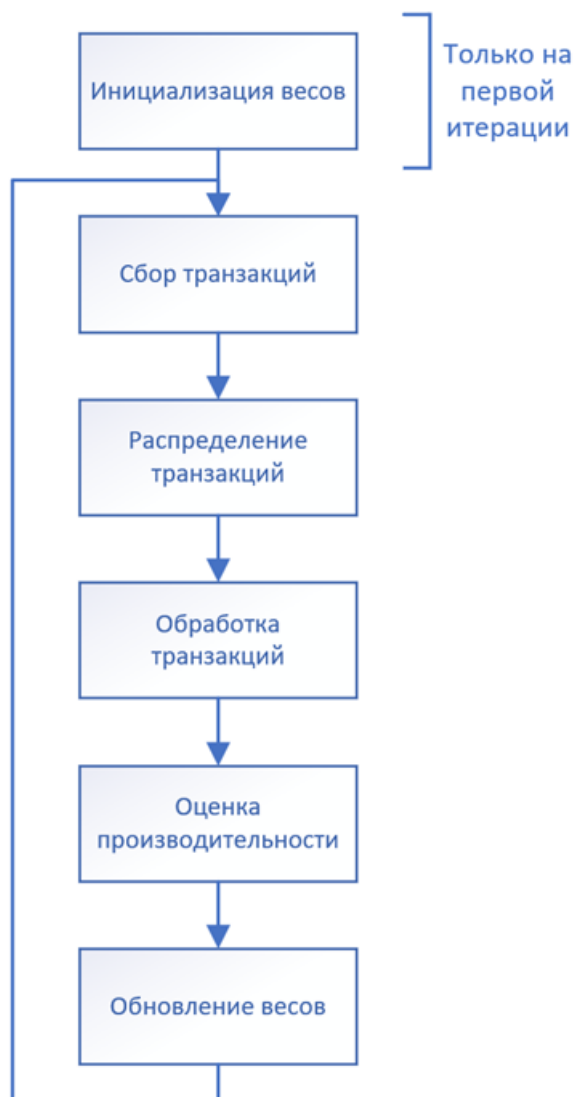


Рис.5 – Алгоритм обработки транзакций в модели Proof of Performance

Доказательство корректности работы алгоритма

Допустим, в сети имеются три шарда 3 с текущими весами $Ws1=0.3, Ws2=0.4, Ws3=0.3$. Допустим, что значения коэффициентов поощрения и наказания $kr=0.1$ и $kp=0.2$.

После создания блока заметим, что:

- 1 обработал 90% назначенных ему транзакций (P);
- 2 обработал 70% (P);
- 3 обработал 80% (P).

Тогда:

$$Rewards1=0.1*0.9=0.09;$$

$$Penaltys1=0.2*(1-0.9)=0.02;$$

$$Ws1next=0.3*1+0.09-0.02=0.3*1.07=0.321;$$

Аналогично для 3 :

$Rewards2=0.1*0.7=0.07$;

$Penaltys2=0.2*(1-0.7)=0.06$;

$Ws2next=0.4*1+0.07-0.06=0.4*1.01=0.404$.

$Rewards3=0.1*0.8=0.08$;

$Penaltys3=0.2*(1-0.8)=0.04$;

$Ws3next=0.3*1+0.08-0.04=0.3*1.04=0.312$.

Нормализация весов:

o ;

$Ws1norm=0.3211.037=0.31$,

$Ws2norm=0.4041.037=0.39$,

$Ws3norm=0.3121.037=0.30$.

Тогда новая сумма весов всех шардов: = , а
новый вес каждого шарда:

= ;

= ;

= .

Номер шарда	Вес «до»	Процент успеха обработки транзакций	Вес «после»
1	0.3	90	0.309546677
2	0.4	70	0.38958534
3	0.3	80	0.30086789

После обновления весов, транзакции распределяются пропорционально новым весам, обеспечивая справедливую и эффективную обработку транзакций в следующем блоке.

Допустим, публикации в сеть ожидает 1000 транзакций. Тогда, на основе новых весов, каждый шард получит:

$Ts1next=1000*0.309546677\approx310$,

$Ts2next=1000*0.38958534\approx390$,

$Ts3next=1000*0.30086789\approx300$.

Сравнение скорости обработки транзакций в сетях Proof of Stake и Proof of Performance

Рассмотрим пример, в котором в двух сетях (А и В) существует по 5 узлов, которые поддерживают их работоспособность.

Сеть **A** работает на алгоритме консенсуса Proof of Stake. Условимся, что каждый валидатор (узел) имеет техническое оснащение, позволяющее обработать 1000 транзакций за 1 блок. Тогда, при предложении нового блока, валидатор обрабатывает 1000 транзакций и помещает их в блок, то есть скорость обработки: 1000транз/блок или 167 транз/сек (при среднем времени блока 6 секунд).

В то же время в сети **B**, которая поддерживается алгоритмом консенсуса Proof of Performance при возникновении задачи по генерации нового блока каждый шард берет на себя задачу по обработке 1000 транзакций. При условии, что шардов в сети 5, то скорость обработки транзакций будет: 5000транз/блок или же 833транз/сек. То есть производительность сети на основе алгоритма консенсуса Proof of Performance в X раз больше, чем в сетях на основе Proof of Stake, где X – количество шардов в сети.



Применение алгоритма консенсуса Proof of Performance

Алгоритм PoP может быть внедрен и оказывать положительное влияние на производительность тех блокчейн-сетей, в которых важна высокая пропускная способность при обработке транзакций, а также масштабируемость.

Алгоритм может быть полезен в сетях следующих типов:

- Сети с большим количеством децентрализованных приложений (dApps)^[7];
- Системы IoT (Интернет вещей)^[8];
- Глобальные платежные сети^[9];
- Системы управления цепочками поставок^[10].

Моделирование работы алгоритма Proof of Performance и анализ его влияния на производительность

Для оценки влияния алгоритма Proof of Performance (PoP) на производительность блокчейн-сети был проведен эксперимент, в ходе которого копия сети Polkadot, развернутая на локальном оборудовании, подвергалась нагрузке и замерам метрик производительности сети. После замеров, сеть была обновлена и в нее был внедрен алгоритм Proof of Performance на замену Proof of Stake, который использовался изначально.

Polkadot использует модель парачейнов, где несколько независимых блокчейнов подключаются к основной сети (релейной цепочке) и используют ее вычислительные

мощности для функционирования. При текущей архитектуре нагрузка от парачейнов концентрируется на одном узле в момент создания блока, что может приводить к узким местам и задержкам при обработке транзакций.

Эксперимент состоял из следующих этапов:

1. Создание копии сети Polkadot и запуск парачейнов:

- Развёртывание сети: запущена тестовая сеть, основанная на исходном коде Polkadot, с использованием алгоритма консенсуса PoS.
- Парачейны: развёрнуто 5 парачейнов, имитирующих различные децентрализованные приложения (dApps) с высокой активностью транзакций. Для имитации органической нагрузки на сеть использовались генераторы транзакций, которые публиковали в сеть определенное количество транзакций, создавая нагрузку на узлы сети.
- Конфигурация узлов: сеть состояла из 10 валидаторов для имитации реальной распределённой сети.

2. Измерение производительности сети на PoS:

- Пропускная способность транзакций: по итогам генерации сетью 50 блоков (примерно 5 минут) было обработано 15934 события (транзакции). Средняя пропускная способность составила 53 транзакции в секунду.
- Отказоустойчивость: при отключении 20% валидаторов производительность сети снизилась до 34 транзакций в секунду в течении следующих 50-ти блоков.
- Загрузка узлов: нагрузка центральных процессоров вычислительных машин, на которых располагались узлы-валидаторы находилась на уровне 90-95%, в случае, если валидатор, который располагался на данной машине, был выбран в качестве предлагающего данный блок и 10-15% в случае ожидания очереди на генерацию блока.

3. Миграция сети на алгоритм Proof of Performance:

- Сохранение текущего состояния: создан хеш текущего состояния сети для обеспечения целостности данных после перехода на новую модель консенсуса.
- Внесение предложения об обновлении: инициировано предложение об обновлении алгоритма консенсуса, которое было одобрено участниками сети.
- Обновление программного обеспечения: в исходный код ядра внесены изменения, реализующие минимальные возможности работы сети на алгоритме Proof of Performance. Обновлен алгоритм выбора шардов для обработки транзакций, а также механизм наказаний и поощрений шардов.
- Запуск обновленной сети: проведен запуск сети на основе PoP с перенесенными парачейнами.

4. Измерение производительности сети на PoP:

- Пропускная способность транзакций: по итогам генерации 50-ти блоков пропускная способность сети увеличилась до 259 транзакций в секунду, что является приростом на 487%.
- Отказоустойчивость: при отключении 20% шардов производительность снизилась примерно на 20%.
- Загрузка узлов: нагрузка центральных процессоров вычислительных машин, на которых располагались узлы-валидаторы находилась на уровне 40-60% в течении всего времени замера.

5. Сравнительный анализ результатов:

Алгоритм консенсуса	Proof of Stake	Proof of Performance
Параметр сети		
Пропускная способность (транзакций в секунду)	53	259
Время создания блока (секунд)	6	6
Процент снижения производительности при отказе 20% узлов	30	20
Загрузка узлов в нагрузке (процентов ЦП)	90-95	40-60

6. Выводы:

- Повышение производительности: внедрение PoP значительно повышает эффективность сети, делая ее более подходящей для приложений с высокой нагрузкой.
- Масштабируемость: PoP обеспечивает лучшую масштабируемость без необходимости увеличения вычислительных ресурсов отдельных узлов.
- Улучшение пользовательского опыта: снижение задержек и увеличение скорости подтверждения транзакций повышает удовлетворенность пользователей.

Сценарии внедрения алгоритма Proof of Performance

1. Децентрализованные приложения (dApps)

Сценарий внедрения:

В высоконагруженных dApps (децентрализованных приложениях), таких как децентрализованные биржи, игры или социальные сети, количество транзакций может быть очень высоким, что создает значительную нагрузку на сеть. Узлы сети могут не справляться с обработкой большого объема данных, особенно если они обладают небольшими вычислительными ресурсами.

Рекомендации по применению:

- Использование алгоритма PoP целесообразно в ситуациях, когда dApps генерируют большое количество транзакций, и требуется высокая пропускная способность сети. Обычно, в подобных ситуациях при применении алгоритма PoS индикатором того, что сеть не справляется с нагрузкой, может служить большое время ожидания подтверждения транзакций или увеличение времени генерации блока.
- Если узлы сети имеют слабые вычислительные мощности, PoP поможет распределить нагрузку и обеспечить стабильную работу приложения.

Потенциальные проблемы и вызовы:

- Синхронизация данных между шардами: возможны сложности с обеспечением согласованности данных.
- Сложность настройки: требуется настройка сети, что включает в себя изменение программного кода ядра и других сетевых параметров для поддержки шардинга и PoP.

Возможные решения:

- Внедрение эффективных алгоритмов консенсуса между шардами для обеспечения согласованности данных.
- Предоставление инструментов и документации для упрощения процесса настройки и интеграции PoP.

2. Системы Интернета вещей (IoT)

Сценарий внедрения:

В IoT-сетях множество устройств генерируют данные, которые необходимо обрабатывать в реальном времени. Ограниченные вычислительные ресурсы устройств и высокая скорость генерации данных требуют эффективного решения для обработки транзакций.

Рекомендации по применению:

- Применение PoP в крупных IoT-сетях с высокой плотностью устройств и большими объемами данных.
- Использование PoP для распределения обработки данных между узлами с более высокими вычислительными мощностями.

Потенциальные проблемы и вызовы:

- Ограниченные ресурсы устройств: IoT-устройства могут не поддерживать необходимую вычислительную нагрузку.
- Безопасность данных: обеспечение безопасности и конфиденциальности данных в распределенной сети.

Возможные решения:

- Использование промежуточных узлов или шлюзов с достаточными ресурсами для обработки данных от IoT-устройств.
- Внедрение протоколов шифрования и аутентификации для защиты данных.

3. Глобальные платежные сети

Сценарий внедрения:

Международные платежные системы требуют высокой скорости обработки транзакций и способности масштабироваться под растущие объемы операций. Задержки и низкая пропускная способность могут негативно влиять на пользовательский опыт и доверие к системе.

Рекомендации по применению:

- Внедрение PoP для ускорения обработки транзакций и увеличения пропускной способности сети.
- Использование PoP в системах с большим количеством микроплатежей, где скорость подтверждения транзакции критична.

Потенциальные проблемы и вызовы:

- Регуляторные требования: необходимо соблюдать правила KYC (знай своего клиента) и AML (противодействие отмыванию денег).
- Безопасность транзакций: обеспечение защиты от мошеннических действий и атак.

Возможные решения:

- Интеграция механизмов идентификации и верификации пользователей в систему.
- Реализация дополнительных уровней безопасности и мониторинга транзакций.

Рекомендации по интеграции алгоритма Proof of Performance в существующие блокчейн-инфраструктуры

Рекомендации по применению алгоритма PoP.

Для повышения производительности и устранения проблем, вызванных недостатком мощности отдельных узлов, следует рассмотреть для интеграции алгоритм Proof of Performance в следующих случаях:

- В сетях с высоким объемом транзакций и требованием к высокой пропускной способности.
- В случае, если узлы сети обладают небольшими вычислительными ресурсами и не справляются с нагрузкой.
- При необходимости повышения масштабируемости и отказоустойчивости сети.

Использование алгоритма может быть необязательным:

- В небольших сетях с низким объемом транзакций, где стандартные алгоритмы консенсуса, такие как PoS, удовлетворяют требованиям.
- В случаях, если требования к производительности и скорости обработки не являются критичными.

Внедрение алгоритма Proof of Performance (PoP) в существующую сеть на Proof of Stake (PoS) требует поэтапного подхода и взаимодействия с участниками сети. Ниже представлен общий алгоритм внедрения PoP:

1. Анализ и подготовка.

- Изучите текущую инфраструктуру PoS, определите цели перехода на PoP (повышение производительности, масштабируемость).
- Выявите возможные узкие места и требования к ресурсам.
- Информировать валидаторов, разработчиков и пользователей о планах.

2. Разработка и тестирование PoP.

- Разработайте версию PoP, совместимую с архитектурой и смарт-контрактами сети.
- Создайте тестовую сеть для испытаний PoP без воздействия на основную сеть.
- Проведите нагрузочные тесты и аудиты безопасности.

3. Сохранение состояния сети.

- Создайте резервную копию текущего состояния сети.
- Сгенерируйте контрольный хеш для проверки целостности после миграции.

4. Предложение обновления консенсуса.

- Используйте механизмы голосования сети для принятия решения об обновлении сети.
- Обеспечьте достижение необходимого кворума.

5. Обновление узлов и программного обеспечения.

- Подготовьте обновленные версии ПО узлов с интегрированным PoP.
- Обеспечьте обратную совместимость для плавного перехода.

- Проведите бета-тестирование с участием ограниченного числа узлов.

6. Развёртывание обновлений

- Определите последовательность обновления узлов.
- Установите мониторинг для отслеживания процесса и быстрого реагирования.
- Предоставьте подробные руководства по обновлению.

7. Запуск обновлённой сети на PoP

- После обновления достаточного числа узлов и получения консенсуса, убедитесь в корректной работе сети.
- Сравните текущий хеш состояния сети с контрольным для подтверждения целостности данных.

8. Мониторинг:

- Обеспечьте мониторинг ключевых метрик сети: пропускную способность, время блоков, нагрузку на узлы.

Внедрение алгоритма Proof of Performance в сеть, работающую на Proof of Stake, представляет собой сложный, но управляемый процесс, который при должной подготовке и реализации может привести к значительным улучшениям производительности и масштабируемости сети.

Анализ преимуществ и недостатков алгоритма Proof of Performance

Преимущества алгоритма Proof of Performance по сравнению с другими алгоритмами консенсуса:

1. Улучшенная масштабируемость и пропускная способность

- Горизонтальный шардинг: использование PoP позволяет эффективно распределять обработку транзакций между шардами, что значительно повышает масштабируемость сети. В отличие от PoS, где нагрузка сосредоточена на узлах-валидаторах, PoP распределяет нагрузку равномерно, позволяя обрабатывать больше транзакций в единицу времени.
- Параллельная обработка: благодаря параллельной обработке транзакций в шардах, сеть может достигать высокой пропускной способности без необходимости увеличения вычислительных ресурсов отдельных узлов.

2. Повышенная отказоустойчивость и надежность сети

- Распределение нагрузки: в PoP выход из строя отдельного шарда не приводит к значительному снижению производительности сети, поскольку другие шарды могут компенсировать нагрузку.
- Снижение рисков перегрузки узлов: отсутствие концентрации обработки на отдельных узлах снижает риски перегрева и сбоев, что повышает общую стабильность сети.

3. Энергоэффективность

- Отказ от ресурсоемких вычислений: в отличие от Proof of Work (PoW), PoP не требует решения сложных криптографических задач, что снижает энергопотребление и делает сеть более экологичной и экономичной.
- Оптимизация ресурсов: узлы используют вычислительные мощности более эффективно, что снижает затраты на поддержание сети.

4. Стимулирование высокой производительности узлов

- Механизмы поощрения и штрафов: PoP мотивирует узлы поддерживать высокую производительность через систему вознаграждений и штрафов, основанных на их эффективности.

5. Гибкость и адаптивность

- Динамическое распределение нагрузки: PoP позволяет динамически изменять распределение транзакций в зависимости от текущей производительности шардов.
- Адаптация к изменяющимся условиям: сеть может быстро реагировать на изменения нагрузки и ресурсов, поддерживая оптимальную производительность.

Недостатки алгоритма Proof of Performance:

1. Сложность реализации и интеграции

- Технические сложности: Внедрение PoP требует значительных изменений в архитектуре сети и глубокого понимания механизмов шардинга.
- Необходимость разработки новых инструментов: требуется создание специальных инструментов и протоколов для обеспечения эффективной работы PoP.

2. Потенциальные риски безопасности

- Уязвимости в шардинге: распределение данных между шардами может создать дополнительные векторы атаки, если не обеспечена надлежащая безопасность.
- Сложности в согласовании данных: обеспечение консистентности и целостности данных между шардами требует надежных механизмов консенсуса.

3. Требования к ресурсам узлов

- Разнородность узлов: узлы с различными вычислительными возможностями могут испытывать сложности в поддержании необходимой производительности.
- Необходимость постоянного мониторинга: для поддержания оптимальной работы сети требуется постоянный мониторинг и управление ресурсами.

4. Сопротивление со стороны сообщества

- Изменение устоявшихся процессов: переход на PoP может встретить сопротивление от участников сети, привыкших к текущим алгоритмам.
- Обучение и адаптация: требуется время и ресурсы для обучения разработчиков и операторов узлов новым методам работы.

Сравнение с другими алгоритмами консенсуса:

1. Proof of Work (PoW):

Преимущества PoP над PoW:

- Более высокая энергоэффективность.
- Лучшая масштабируемость и пропускная способность.
- Отсутствие необходимости в дорогостоящем оборудовании для майнинга.

Недостатки PoP по сравнению с PoW:

- Более сложная система достижения консенсуса.

2. Proof of Stake (PoS):

Преимущества PoP над PoS:

- Улучшенная масштабируемость за счет горизонтального шардинга.
- Более равномерное распределение нагрузки между узлами.
- Меньшая зависимость от стейка (количества токенов) для участия в консенсусе.

Недостатки PoP по сравнению с PoS:

- Сложность внедрения в существующие сети.
- Требуется более сложных механизмов для обеспечения безопасности и согласованности данных.

3. Delegated Proof of Stake (DPoS):

Преимущества PoP над DPoS:

- Более децентрализованный подход без концентрации власти у делегатов.
- Стимулирование производительности каждого узла, а не только избранных.

Недостатки PoP по сравнению с DPoS:

- Более сложная система управления узлами и нагрузкой.
- Возможные трудности в достижении консенсуса между большим количеством шардов.

Проблемы рассмотренных алгоритмов, решение, предложенное алгоритмом PoP, доказательство решения и выводы представлены в следующей таблице:

Проблема	Решение в PoP	Доказательство	Вывод
Перегрузка узлов в PoS	Горизонтальный шардинг и адаптивное распределение весов	Математический и аппарат распределения весов	Динамическое распределение нагрузки между узлами
Снижение отказоустойчивости сети при сбоях	Распределение нагрузки между шардами	Проведенный эксперимент замера производительности сети при отключении 20% узлов	Устойчивость сети сохраняется даже при отключении 20% узлов
Ограниченная пропускная способность	Параллельная обработка транзакций	Проведенный эксперимент сравнения скорости обработки транзакций в сетях PoS и PoP	Увеличение производительности сети до 5000 транзакций/блок
Задержки при обработке транзакций в сети PoS	Распределение транзакций пропорционально производительности шардов	Проведенный эксперимент замера производительности обработки транзакций	Снижение времени ожидания обработки транзакций за счёт равномерной загрузки шардов
		Математический	Система штрафов и

Риск злоупотреблений со стороны узлов	Введение механизма оценки производительности узлов и санкций за неэффективность	аппарат распределения весов, основанный на системе штрафов и поощрений	поощрений мотивирует узлы к поддержанию высокой производительности
Сложность интеграции новых узлов в сеть	Механизм нормализации весов обеспечивает справедливое распределение транзакций в ходе подключения новых узлов к сети	Математический аппарат распределения весов и его справедливое назначение новых весов	Динамическое распределение нагрузки автоматически адаптируется к изменению числа узлов
Сложности в обеспечении согласованности данных	Введение дополнительных хэш-параметров для проверки согласованности	Измененная структура блока включает в себя параметры для учета согласованности данных всеми узлами сети	Хэш соответствия данных между шардами минимизирует риски расхождений в данных

Сравнение взглядов отечественных и зарубежных исследователей на проблему масштабирования блокчейн-сетей

Научные исследования в области алгоритмов консенсуса и масштабируемости блокчейн-сетей активно развиваются как в зарубежной, так и в отечественной литературе. Однако подходы к решению данных вопросов существенно различаются.

1. Зарубежные исследования.

Зарубежные работы, такие как исследование «Sharding-Based Proof-of-Stake Blockchain Protocols: Key Components & Probabilistic Security Analysis» акцентируют внимание на безопасности и вероятностной оценке устойчивости алгоритмов консенсуса. Основное внимание уделяется их применимости в условиях высокой нагрузки на сеть и возможности работы с большим количеством активных узлов. В их исследовании модели PoS процесс увеличения производительности сети рассматривается через внедрение шардинга [\[11\]](#).

Другим примером изучения процесса масштабирования блокчейн-сетей является работа Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer и Robbert van Renesse «Bitcoin-NG: A Scalable Blockchain Protocol», в которой авторы представляют протокол Bitcoin-NG. Он предлагает деление блокчейна на микро- и макроблоки для ускорения обработки транзакций. Однако в нём отсутствуют механизмы динамического распределения нагрузки между узлами, что делает его уязвимым к перегрузкам [\[12\]](#).

2. Отечественные исследования.

В работе Абдулжалилова А.З. «Методы и стратегии масштабируемости блокчейн-технологий: анализ, сравнение и перспективы» автор рассматривает несколько методов, которые обеспечивают увеличение ключевых параметров производительности сети через распределенные системы управления списками, использования шардинга, применение

sidechains и внедрение алгоритмов консенсуса, дружественным к процессу масштабирования [\[13\]](#).

Большое внимание поиску решения проблемы масштабируемости в блокчейн сетях выделяет Грепан В.Н. в своей работе «Практические проблемы использования блокчейн-технологий». В своей работе он рассматривает возможность увеличения размера блоков, использование алгоритмов консенсуса с повышенной производительностью, шардирование, развитие межблокчейных решений и оптимизацию протоколов и алгоритмов как способы решения проблемы масштабирования блокчейн-сетей [\[14\]](#).

3. Основные различия.

Аспект	Зарубежный подход	Отечественный подход
Фокус исследований	Безопасность и вероятность устойчивости алгоритмов консенсуса	Практическая реализация решений масштабируемости
Методы масштабирования	Внедрение шардинга в PoS для повышения производительности; использование микро- и макроблоков	Расширение размеров блоков, использование sidechains, межблокчейновых решений и шардирования
Решение проблемы нагрузки	Отсутствие механизмов динамического распределения нагрузки между узлами	Динамическое распределение нагрузки через адаптацию алгоритмов консенсуса и оптимизацию протоколов
Основной фокус масштабируемости	Работа с высокой нагрузкой на сеть и большим количеством активных узлов	Увеличение производительности сети с учётом ограниченных вычислительных ресурсов
Инструменты решения проблем	Оптимизация структуры блоков и их деление на микро- и макроблоки	Адаптация алгоритмов консенсуса, использование распределённых систем управления и развитие протоколов
Практическое применение	Глобальные сети с большими пулом узлов и высокими нагрузками	Локальные сети с ограниченными ресурсами и потребностью в эффективной интеграции (решения для бизнеса и госструктур)

Согласно рассмотренным взглядам на решение проблемы масштабируемости блокчейн-сетей, становится понятно, что имеет смысл поиск метода, который бы удовлетворял сразу нескольким задачам при построении процесса масштабируемости.

Модель Proof of Performance (PoP) предлагает несколько нововведений, которые вносят значительный вклад в развитие алгоритмов консенсуса и масштабировании блокчейн-сетей:

1. Интеграция горизонтального шардинга:

- В отличие от большинства существующих решений, PoP сочетает шардинг с адаптивным управлением узлами, что позволяет одновременно увеличивать пропускную способность и сохранять устойчивость сети.

2. Система динамического распределения нагрузки:

- PoP вводит механизм расчёта производительности узлов и адаптивного обновления их весов, чего нет в традиционных подходах.

3. Устойчивость к сбоям:

- Эксперименты показывают, что PoP сохраняет производительность сети даже при выходе из строя до 20% узлов, что превосходит показатели других алгоритмов, таких как PoS.

4. Практическая применимость:

- Модель PoP разработана с учётом ограниченных вычислительных ресурсов, что делает её особенно актуальной для внедрения в Российской Федерации, включая государственные и коммерческие проекты.

Таким образом, сравнение отечественных и зарубежных исследований показало, что модель PoP не только решает многие известные проблемы алгоритмов консенсуса, но и предлагает новые подходы, которые могут быть полезны в условиях реальных ограничений. Прирост научного знания заключается в разработке гибкой системы управления узлами, повышающей производительность и устойчивость блокчейн-сетей.

Заключение

В статье рассмотрен принцип работы модели Proof of Performance, учтены и описаны ее особенности, приведен разбор и пример работы алгоритма. Проведен анализ скорости обработки транзакций и определены возможные пути ее применения.

Представленная и описанная модель позволит внедрить адаптивный подход к потреблению ресурсов, а также найти компромисс между потребностями в производительности сети и затратами на поддержание ее работоспособности.

Библиография

1. Бауэр В.П., Побываев С.А., Кузнецов Н.В. Потенциал использования технологии распределенного реестра (блокчейн) в системах государственного управления // Фундаментальные исследования. 2019. № 12 (часть 2). С. 247–252.
2. Борискевич И.А. Алгоритмы консенсуса в блокчейн сетях // Белорусский государственный университет информатики и радиоэлектроники. 6-я Научная Конференция Аспирантов, Магистрантов и Студентов БГУИР. Минск, 2020. С. 116–117.
3. Luu, L., Narayanan, V., Zheng, C., Baweja, K., Gilbert, S., & Saxena, P. (2016). A Secure

- Sharding Protocol for Open Blockchains. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (pp. 17–30).
4. Kokoris-Kogias, E., Jovanovic, P., Gasser, L., Gailly, N., Syta, E., & Ford, B. (2017). OmniLedger: A Secure, Scale-Out, Decentralized Ledger via Sharding. IACR Cryptology ePrint Archive, 2017(406).
5. Zamani, M., Movahedi, M., & Raykova, M. (2018). RapidChain: Scaling Blockchain via Full Sharding. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (pp. 931–948).
6. King, S., & Nadal, S. (2012). PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake. Retrieved from <https://bitcoin.peryaudo.org/vendor/peercoin-paper.pdf>
7. Swan, M. (2015). Blockchain 2.0: Contracts. In Blockchain: Blueprint for a New Economy (pp. 9–10). O'Reilly Media, Inc.
8. Шаламов Г. А., Петухов А. С. Слияние технологий IoT и блокчейн: от теории до реального времени // Прогрессивная экономика. 2023. № 9. С. 32. DOI: 10.54861/27131211_2021_9_32.
9. Goswami, S. (2017). Scalability analysis of blockchains through blockchain simulation. Thesis, Master of Science in Computer Science, University of Nevada, Las Vegas.
10. Zamyatin, A., Harz, D., Lind, J., Gudgeon, L., Werner, S., & Knottenbelt, W. J. (2019). XCLAIM: Trustless, Interoperable, Cryptocurrency-Backed Assets. In IEEE Symposium on Security and Privacy (pp. 193–210).
11. Makrakis, D., & Senhaji, A. (2023). Sharding-Based Proof-of-Stake Blockchain Protocols: Key Components & Probabilistic Security Analysis. Sensors, 23(5), 2819. DOI: 10.3390/s23052819.
12. Eyal, I., Gencer, A. E., Sirer, E. G., & van Renesse, R. (2016). Bitcoin-NG: A Scalable Blockchain Protocol. In Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI '16) (pp. 45–59).
13. Абдулжалилов А.З. Методы и стратегии масштабируемости блокчейн-технологий: анализ, сравнение и перспективы // Международный научный журнал «Вестник науки.» 2023. № 11 (68). Т. 4. С. 625–634.
14. Гречан В.Н. Практические проблемы использования блокчейн-технологий // Научный сетевой журнал «Столыпинский вестник.» 2024. № 9.

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Статья посвящена исследованию и описанию модели и алгоритма консенсуса Proof of Performance (PoP) для блокчейн-сетей. В работе подробно рассматриваются основные элементы модели, преимущества предложенного алгоритма и его потенциал для повышения производительности и масштабируемости блокчейн-систем.

Методология исследования базируется на сравнительном анализе существующих алгоритмов консенсуса, таких как Proof of Stake (PoS), и предложенного алгоритма Proof of Performance. В статье представлены теоретические расчеты, алгоритмические схемы и примеры применения PoP, а также результаты моделирования, подтверждающие эффективность предложенного подхода.

Исследование актуально в свете растущих требований к производительности и масштабируемости блокчейн-сетей. Традиционные алгоритмы, такие как PoS, имеют ограничения по пропускной способности и устойчивости к нагрузкам. Внедрение новых моделей, таких как PoP, может существенно повысить эффективность обработки

транзакций и устойчивость сети к сбоям.

Научная новизна работы заключается в предложении и разработке нового алгоритма консенсуса, который использует горизонтальный шардинг для равномерного распределения нагрузки между узлами сети. Это позволяет повысить пропускную способность сети и уменьшить вероятность возникновения сбоев из-за перегрузки отдельных узлов.

Статья написана в научно-техническом стиле, характеризуется логичной структурой и последовательным изложением материала. Введение содержит обоснование актуальности темы, основная часть включает описание алгоритма, его преимущества и примеры применения. Заключение подводит итоги и намечает направления дальнейших исследований.

Авторы делают обоснованные выводы о преимуществах модели Proof of Performance, таких как увеличенная скорость обработки транзакций и улучшенная масштабируемость сети. Результаты моделирования подтверждают эффективность предложенного подхода. Статья будет интересна широкому кругу специалистов в области блокчейн-технологий, включая разработчиков, исследователей и инженеров, занимающихся вопросами повышения производительности и масштабируемости блокчейн-сетей.

Рекомендации по доработке:

Для улучшения восприятия материала и повышения его практической ценности, рекомендуется дополнить работу следующими элементами:

1. Примеры практического применения:

- Добавить конкретные примеры внедрения алгоритма Proof of Performance в реальные блокчейн-сети. Это может включать описание пилотных проектов или сценариев использования, которые помогут читателям лучше понять практическую значимость предложенного решения.

- Привести данные о потенциальных улучшениях производительности и пропускной способности сети при переходе от традиционных алгоритмов консенсуса к PoP на основе реальных кейсов.

2. Сценарии внедрения:

- Описать возможные сценарии внедрения алгоритма Proof of Performance в различных типах блокчейн-сетей, таких как децентрализованные приложения (dApps), системы Интернета вещей (IoT) и глобальные платежные сети.

- Рассмотреть потенциальные проблемы и вызовы, которые могут возникнуть при внедрении PoP, а также предложить возможные решения для их преодоления.

3. Включить практические рекомендации для разработчиков и инженеров по интеграции алгоритма PoP в существующие блокчейн-инфраструктуры. Это может включать советы по настройке сети, оптимизации вычислительных мощностей и управлению нагрузкой.

4. Провести более детальный анализ преимуществ и недостатков алгоритма Proof of Performance по сравнению с другими алгоритмами консенсуса. Это поможет читателям объективно оценить целесообразность использования PoP в различных контекстах.

Внесение указанных дополнений сделает статью более полной и информативной, а также увеличит её ценность для практического применения.

В целом, работа представляет собой значительный вклад в область исследований блокчейн-технологий и заслуживает публикации в журнале "Программные системы и вычислительные методы".

Результаты процедуры повторного рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Предмет исследования. С учётом выбранного заголовка статья должна быть посвящена вопросам использования модели и алгоритма консенсуса Proof of Performance. Содержание статьи соответствует заявленной теме, однако полностью её не раскрывает, т.к. не содержится обоснований авторским суждениям.

Методология исследования базируется на анализе и синтезе данных. Было бы также интересно графически представить предлагаемые рекомендации. Ценно, что при обосновании доказательств корректности работы алгоритма автор применяет математический аппарат, обеспечивающий позитивное впечатление от рецензируемой статьи. Однако важно также полученные выводы в процессе применения математического аппарата сопоставить с обозначенными проблемами использования алгоритма и путями их решения. Например, это может быть представлено в виде схемы.

Актуальность исследования обусловлена важностью изучения вопросов обработки транзакций в алгоритме Proof of Performance как для конкретных экономических агентов, так и для обеспечения цифровой трансформации в Российской Федерации. При этом потенциальную читательскую аудиторию интересуют конкретные обоснованные авторские рекомендации по решению выявленных в процессе исследования проблем.

Научная новизна в представленном на рецензирование материале частично содержится. В частности, она связана с рекомендациями по интеграции алгоритма Proof of Performance в существующие блокчейн-инфраструктуры.

Стиль, структура, содержание. Стиль изложения является научным. Текст структурирован. Содержание статьи представляет собой развёрнутый план, приведённые тезисы следует наполнить конкретными авторскими обоснованиями и дополнениями. В содержании статьи автор привёл результаты анализа преимуществ и недостатков алгоритма Proof of Performance, однако приведённые тезисы нуждаются в обосновании, а также в рекомендациях по устранению выявленных недостатков. При проведении доработки статьи было бы интересно сравнить с другими алгоритмами.

Библиография. Библиографический список состоит из 11 наименований. Ценно, что автор проработал зарубежные источники, однако абсолютно не уделил внимание отечественным научным публикациям, что говорит о недостаточной проработке методологической базы. При проведении доработки статьи данное замечание следует обязательно устранить.

Апелляция к оппонентам. Несмотря на сформированный список научных публикаций, какой-либо научной дискуссии не осуществлено. Было бы интересно показать отличия отечественной и зарубежной мысли по данному вопросу. При проведении доработки статьи необходимо обязательно полученные результаты обсудить с теми, кто были получены другими авторами. Также важно показать ответ на вопрос: "В чём состоит прирост научного знания по сравнению с уже имеющимся в литературе?"

Выводы, интерес читательской аудитории. С учётом вышеизложенного заключаем о необходимости доработки статьи с учётом обозначенных замечаний. После проведения доработки и проведения повторного рецензирования может быть решён вопрос о целесообразности опубликования данной статьи. При качественной доработке она будет востребована у лиц, занимающихся данными вопросами, в т.ч. в рамках учебного

процесса в высших учебных заведениях.

Результаты процедуры окончательного рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Представленная статья на тему «Модель и алгоритм консенсуса Proof of Performance» соответствует тематике журнала «Программные системы и вычислительные методы» и посвящена актуальной проблеме повышения производительности и масштабируемости блокчейн-сетей обусловлена растущими требованиями к обработке транзакций в высоконагруженных системах, таких как финансовые технологии, системы Интернета вещей (IoT) и децентрализованные приложения (dApps). Как указывают авторы в статье, существующие алгоритмы консенсуса, включая Proof of Stake (PoS) на базе ядра Tendermint демонстрируют ограниченную пропускную способность, обусловленную высокой нагрузкой и ограниченными ресурсами производительности узлов-валидаторов сети. Эти ограничения создают узкие места в функционировании сетей и снижают их устойчивость к изменениям нагрузки.

В статье представлен достаточно широкий анализ литературных российских и зарубежных источников по теме исследования.

В качестве новизны исследования авторы указывают предложение и описание новейшего алгоритма Proof of Performance, сочетающего адаптивное распределение нагрузки и учет производительности узлов, что позволяет существенно увеличить пропускную способность сети и её устойчивость к изменениям нагрузки. Представленная работа закладывает основу для дальнейших исследований, направленных на интеграцию описанного алгоритма консенсуса в существующие блокчейн-системы и развитие технологий децентрализованного управления нагрузкой.

Стиль и язык изложения материала является научным и доступным для широкого круга читателей. Статья по объему соответствует рекомендуемому объему от 12 000 знаков.

Статья достаточно структурирована - в наличии введение, заключение, внутреннее членение основной части (Описание модели Proof of Performance, Структура блока в модели Proof of Performance, Распределение весов, Доказательство корректности работы алгоритма, Сравнение скорости обработки транзакций в сетях Proof of Stake и Proof of Performance, Применение алгоритма консенсуса Proof of Performance, Моделирование работы алгоритма Proof of Performance и анализ его влияния на производительность, Сценарии внедрения алгоритма Proof of Performance, Рекомендации по интеграции алгоритма Proof of Performance в существующие блокчейн-инфраструктуры, Анализ преимуществ и недостатков алгоритма Proof of Performance, Сравнение взглядов отечественных и зарубежных исследователей на проблему масштабирования блокчейн-сетей).

Авторами рассмотрен принцип работы модели Proof of Performance, учтены и описаны ее особенности, приведен разбор и пример работы алгоритма; проведен анализ скорости обработки транзакций и определены возможные пути ее применения.

Практическая значимость статьи четко обоснована. Представленная и описанная модель позволит внедрить адаптивный подход к потреблению ресурсов, а также найти компромисс между потребностями в производительности сети и затратами на поддержание ее работоспособности.

Статья «Модель и алгоритм консенсуса Proof of Performance» может быть рекомендована к публикации в журнале «Программные системы и вычислительные методы».

Программные системы и вычислительные методы

Правильная ссылка на статью:

Димитриченко Д.П. Анализ целесообразного поведения различных типов автоматов в условиях игры в размещения // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.72488 EDN: SQCVFI URL: https://nbpublish.com/library_read_article.php?id=72488

Анализ целесообразного поведения различных типов автоматов в условиях игры в размещения

Димитриченко Дмитрий Петрович

кандидат технических наук

научный сотрудник, Институт прикладной математики и автоматизации

360000, Россия, г. Нальчик, ул. Шортанова, 89а

✉ dimdp@rambler.ru



[Статья из рубрики "Теория, программное обеспечение и языки параллельных вычислений"](#)

DOI:

10.7256/2454-0714.2024.4.72488

EDN:

SQCVFI

Дата направления статьи в редакцию:

27-11-2024

Дата публикации:

05-12-2024

Аннотация: Объектом исследования настоящей работы являются однородные коллективы автоматов, обладающие свойством целенаправленного поведения. Предметом настоящего исследования является сравнение различных конструкций таких автоматов при реализации условий игры в размещения. Целью настоящего исследования является количественная оценка влияния инерционных свойств автоматов и значений глубины памяти на эффективность функционирования коллектива автоматов в рамках игры в размещения. В рассматриваемых коллективах автоматы выполняют действия в заданной среде обитания с различной степенью результативности. Автоматы, в соответствии со своей конструкцией, реагируют на поданный входной сигнал очередным действием. Оценка эффективности автомата определяется, как сумма положительных сигналов, или отрицательных сигналов, полученных автоматом за

рассматриваемый промежуток времени. Указанная характеристика зависит, как от заявленной конструкции автомата, так и от глубины его памяти. Требуется определить наиболее простые конструкции автоматов, позволяющие достигать оптимума эффективности в заданной окружающей среде наискорейшим образом. Формализация, как свойство окружающей среды, так и действий автоматов, а также обработка полученных результатов производится при помощи аппарата теории игр. В этом случае значения эффективности функционирования автоматов представляются, как совокупные суммы выигрышей и проигрышей игроков-автоматов. Новизной проведенного исследования является построение интегральной оценки эффективности поведения коллектива автоматов, позволяющей проследить не только стремление коллектива к максимизации суммы поощрений, но и к минимизации внутриколлективной конкуренции. Полученный результат позволяет проследить влияние инерционных качеств автоматов, реализованных в виде соответствующих конструкций, на эффективность функционирования в заданной окружающей среде, формализованной в виде игры в размещения. Автомат с линейной тактикой и автомат Крылова образуют две предельные реализации автоматной стратегии приближения к оптимуму. Первые за счет высокой скорости смены действий, вторые за счет длительного пребывания в состояниях, близких к оптимальным. Областью применения полученных результатов является дальнейшее исследование более сложных динамических сред при помощи наиболее простых конструкций автоматов, так как синхронные коллективы автоматов в процессе вычислительной реализации плохо поддаются распараллеливанию, что приводит к значительному росту временных и вычислительных затрат при усложнении структуры динамических сред или при увеличении данных оптимизационных задач.

Ключевые слова:

однородный коллектив автоматов, целесообразное поведение, поощрения, штрафы, обучение с подкреплением, автомат, глубина памяти, теория игр, оптимальная стратегия, игра в размещения

Введение

Организация сложного поведения децентрализованных систем, построенных на базе простых элементов находит свое отражение в таком разделе машинного обучения, как «Коллективное поведение автоматов» [\[1\]](#), имеющее широкое прикладное значение от решения задач автоматной оптимизации, интеллектуального управления, построение робототехнических систем [\[2\]](#), и до вопросов психологии [\[3\]](#).

При этом, связь между нейросетевым и автоматным подходами является достаточно тесной [\[4, 5\]](#). В связи с ростом вычислительных возможностей многопользовательских и многопоточных систем возникает объективная потребность в анализе и управлении [\[6, 7\]](#) совокупностью агентов действующих децентрализованно, в соответствии с собственным целеполаганием и взаимодействующих друг с другом и заданной окружающей средой.

При этом часто возникают ситуации, когда одно и тоже действие, выбранное несколькими агентами, может быть физически выполнено в конкретный момент времени только одним из них.

Например, только один из потоков, обрабатывающих актуальные данные, может производить операцию чтения (или записи). Всем остальным претендентам доступ к этим

данным блокируется.

Только один робот на перекрестке, из двух подъехавших по разным путям, может пересекать противоположную трассу. Оба робота не могут одновременно двигаться (будет столкновение), или одновременно ожидать прохождения другого робота, тем самым создав ситуацию взаимоблокировки прохождения перекрестка на неопределенный срок не только текущими, но и следующими за ними роботами.

В биологической интерпретации, совместно выбираемым действиям агентов соотносятся, разделяемые несколькими животными, определенные участки экологической ниши. Тогда количество актуального ресурса на участке делится (не обязательно поровну) между всеми, присутствующими на нем животными.

Аналогичная ситуация возникает при активном использовании интернет-трафика несколькими приложениями.

Это простейшие случаи, иллюстрирующие правила поведения в рамках игры в размещения, когда действие, выбранное несколькими агентами, в отдельно взятый момент времени может выполнить (в нашем случае получить поощрение) только один из них (в нашем случае один из автоматов, выбравших это действие).

Все выше перечисленное позволяет привлечь к анализу поведения мультиагентных систем (конечного множества агентов в искусственной среде) теорию вероятности, теорию игр, теорию автоматов для получения теоретически обоснованных и подтверждаемых методами имитационного моделирования результатов.

Примененный в настоящей работе автоматный подход предоставляет в распоряжение исследователя формализованный (в рамках дискретной математики) метод, позволяющий выполнить формальную постановку задачи и проанализировать поведение агентов и самой заданной среды обитания в терминах автоматной модели (коллектива автоматов), входных, выходных и внутренних алфавитов, а также правил перехода и результатов выбора.

Целью настоящей работы является получение количественных оценок влияния инерционных свойств (различным образом проявляемых) определенных типов автоматов и различных значений глубины памяти на эффективность функционирования коллектива автоматов в стохастической среде, организованной по правилам игры в размещения.

Новизной проведенного исследования является следующее:

1. Обоснование (теоретическое доказательство соответствующего утверждения и его подтверждение при выполнении вычислительных экспериментов) о том, что в стохастической среде игры в размещения в качестве доверительного интервала времени функционирования коллектива автоматов достаточно принять время функционирования отдельного (эталонного) автомата. Это позволило минимизировать время проведения вычислительных экспериментов для исследуемых типов автоматов, что особенно актуально при росте значений глубины памяти и, как следствие, росте количества времени вычислений.
2. Такая оценка доверительного интервала оказывается инвариантной относительно заданного типа автоматов. При этом, чем больше коллектив автоматов, тем точнее оказывается полученная таким образом оценка актуального интервала времени.
3. Применение для оценки эффективности коллектива автоматов интегральной

величины математических ожиданий всех автоматов, которая в условиях не равнозначных по цене действий, является особенно информативной и позволяет точно интерпретировать приближение коллектива автоматов к статистически оптимальному состоянию, так как в силу стохастичности окружающей среды автоматы фактически всегда находятся в движении.

4. Удалось пронаблюдать, что полученная таким образом интегральная величина эффективности функционирования коллектива автоматов, не только количественно стремится к максимизации совокупности поощрений от среды, но и информирует о стремлении коллектива к минимизации уровня внутриколлективной конкуренции.

Одним из традиционных методов анализа поведения автоматов является применение аппарата теории игр [8]. Такой выбор обусловлен тем, что теория игр позволяет легко создавать формализуемые системы правил (среда) и стратегий поведения (доступные действия), определяющие характер взаимодействия отдельного автомата (или совокупности автоматов) и окружающей среды [9].

Это обстоятельство позволяет сформировать замкнутый контур Среда-Автомат, в котором действия автомата определяют реакцию окружающей среды, которая в свою очередь, через обратную связь оказывает влияние на выбор последующего действия автомата.

Первые интересные модели такого типа были созданы Михаилом Львовичем Цетлиным [9]. Он выступил создателем целого направления исследований, получившего название «коллективное поведение автоматов» [9-12]. Им были сформулированы основные принципы, лежащие в основе подобных моделей, и способы их реализации.

Формализация понятия целенаправленного поведения позволила М. Л. Цетлину сформулировать базовый принцип организации сложного поведения совокупности децентрализованных систем, образующих коллективы автоматов.

Основные положения этого подхода состоят в следующем:

- 1) принцип суперпозиции;
- 2) принцип соизмеримости;
- 3) принцип универсальности структуры автомата.

Принцип суперпозиции. Любое достаточно сложное поведение складывается из совокупности простых поведенческих актов. Совместная реализация таких простых поведенческих актов и простейшее взаимодействие приводят в результате к весьма сложным поведенческим процессам.

Принцип соизмеримости. Степень целесообразного поведения анализируемой кибернетической системы рассматривается, как величина математического ожидания совокупности штрафов и поощрений, полученных этой системой от окружающей среды за наблюдаемый промежуток времени t . Эта величина находится в интервале со следующими границами:

Левая граница – это значение математического ожидания количества штрафов и поощрений полученных простейшей системой, реализующей стратегию случайного выбора доступных в данной среде действий из множества $D=d_1, \dots, d_m, m \geq 2$.

Правая граница – это величина математического ожидания штрафов и поощрений,

полученных системой, которая в любой момент времени t , $t > 1$, всегда реализует заведомо оптимальное в данной среде действие $d_i = d^*$, $i = 1, \dots, m$.

Очевидно, что чем ближе величина математического ожидания поощрений анализируемой системы к левой границе определенного таким образом интервала, тем точнее ее поведение соответствует стратегии «случайного выбора». А чем ближе эта величина к правой границе интервала, тем ближе характер поведения «к системе, осведомленной о наилучшей возможной стратегии».

Такое определение целесообразного поведения не зависит от структуры анализируемой системы, а опирается только на статистический результат взаимодействия со средой.

Принцип универсальности. Конструкция анализируемой системы не должна содержать эмпирических данных об оптимальных (или неоптимальных) действиях в данной среде, т. е. такие действия система выявляет в процессе своего функционирования в течение некоторого времени t .

Первая автоматная реализация такой системы была предложена М. Л. Цетлиным.

Автоматная реализация

Приведем формальное описание автомата:

$X = x_1, \dots, x_k$ – множество входных сигналов, поступающих от окружающей среды на вход автомата (входной алфавит).

$D = d_1, \dots, d_m$ – конечное множество доступных автомату действий, (выходной алфавит).

$S = s_1, \dots, s_n$, $2 \leq m \leq n$ – конечное множество внутренних состояний автомата (внутренний алфавит).

Правила функционирования автомата в дискретные моменты времени t однозначным образом задаются двумя функциями:

Функцией переходов внутренних состояний: $st+1 = F(st, xt)$, и начальным внутренним состоянием в нулевой момент времени $t=t_0$: $s_0 = S(t_0)$.

Функцией зависимости выходных сигналов (действий) от внутренних состояний: $dt = G(st)$.

Общая постановка задачи

Пусть задана некоторая среда E , в которой будут функционировать автоматы исследуемых типов.

Пусть также имеются типы автоматов, для которых возможно установить степень целесообразного поведения в среде E определенным выше образом.

Требуется:

- определить степень влияния глубины памяти q каждого из рассматриваемых типов автоматов на изменение степени целесообразности поведения в указанной среде E ;
- на основе полученных результатов определить тип конструкции автомата, достигающей заданного уровня целесообразности с минимально возможной глубиной памяти $q=q^*$;
- в случае наличия нескольких эквивалентных конструкций, указать наиболее простую в

реализации конструкцию.

Биологические предпосылки

Результаты из экспериментальной биологии потребовали сформулировать кибернетический «белый ящик», обладающий целесообразным поведением.

Проводимый эксперимент состоял в следующем [9]:

Подопытное животное помещалось в основание Т-образного лабиринта с возможностью выбора одного из двух действий:

«Повернуть влево»;

«Повернуть вправо».

По условиям эксперимента в конце каждого из двух ответвлений реализовывались неизвестные подопытному животному благоприятные (или неблагоприятные) условия с независимыми вероятностями для каждого из двух событий (ответвлений).

Требовалось установить способность подопытных животных различать свойства окружающей среды, носящие вероятностный характер.

Оказалось, что, несмотря на ошибки выбора в начале серии экспериментов, животные в последствии верно ассоциировали выбираемый ими поворот (выполняемое действие) с тем, для которого вероятность штрафа (вероятность попасть в неблагоприятную ситуацию) была минимальной.

Стационарная окружающая среда

Стационарная окружающая среда E является математическим описанием условий Т-образного лабиринта.

В Т-образном лабиринте было всего два доступных для выполнения подопытным животным действия: «повернуть влево» и «повернуть вправо». В каждом из выбранных ответвлений ожидалась своя вероятность поощрения P_l и P_r .

Обобщением этой ситуации является стационарная среда с m исходами (действиями).

Для каждого из m действий задается совокупность значений вероятностей: либо поощрения p_i , либо штрафа $1 - p_i$, $i=1, \dots, m$.

Выбор оптимального действия в такой среде сводится к определению действия с максимальной вероятностью поощрения (минимальной вероятностью штрафа).

При этом, ни подопытное животное в биологических экспериментах, ни соответствующий ему по поведению автомат не располагают знаниями о исходных значениях вероятностей, но вынуждены эту информацию получать в опосредованной форме через сигналы поощрения и штрафа, получаемые от окружающей среды.

Типы конструкций автоматов

Первой конструкцией автомата, способного вести себя целесообразно в описанной выше стационарной среде E , является автомат с линейной тактикой, предложенный М. Л. Цетлиным.

Автомат $L1$ (с глубиной памяти $q=1$) состоит из m состояний $S=s_1, \dots, s_m$.

За каждым из m состояний si закреплено действие $di, i=1, \dots, m$.

Автомат L1, функционирует следующим образом:

В некоторый момент времени $t=t^*$, находясь в состоянии si , автомат L1 выполняет выходное действие $di, i=1, \dots, m$.

В ответ на это действие окружающая среда и формирует сигнал, принимающий значение «Поощрение» в соответствии с вероятностью pi (или «Штраф» с вероятностью $1 - pi$).

Этот сигнал подается на вход автомата l1.

Если автомат L1 на входе получает сигнал «Штраф», то он переходит в состояние $si+1$ и, следовательно, переключает внешнее действие с di на $di+1$.

При получении сигнала «поощрение» автомат L1 остается в исходном состоянии si , при этом смена действия не происходит.

Таким образом, в следующий момент времени t^*+1 автомат L1 выполнит действие в соответствии со своим внутренним состоянием, и процесс взаимодействия автомата со средой повторится описанным выше образом.

Закрепляя за каждым из m действий di q последовательных состояний, мы получим последовательность линейных автоматов с монотонно возрастающей глубиной памяти q .

Цетлину удалось показать, что последовательность таких автоматов $L1, L2, \dots, Lq$, функционирующих в стационарной среде E , является асимптотически оптимальной, т. е. чем больше глубина памяти линейного автомата Lq , тем дольше такой автомат выполняет самое оптимальное действие (с максимальной вероятностью поощрения), и почти никогда не покидает связанные с ним состояния.

Усложнением автомата с линейной тактикой является автомат Кринского.

Он характеризуется тем, что при получении поощрения этот автомат всегда переходит в самое глубокое состояние, соответствующее текущему, выполняемому им, действию.

При получении штрафа от окружающей среды этот автомат реагирует точно также, как и автомат с линейной тактикой, понижая на единицу номер состояния текущего (выполненного) действия.

Для этого автомата также доказана теорема об асимптотической оптимальности при функционировании в стационарных средах E .

Следующим по степени усиления свойства инерционности является автомат Роббинса.

Он отличается от автомата Кринского тем, что в отличие от него, при смене действия автомат Роббинса сразу переходит в самое глубокое состояние, соответствующее новому действию.

В остальном он ничем не отличается от автомата Кринского.

Для него также верна теорема об асимптотической оптимальности в стационарных средах.

Заметим, что у трех автоматов: автомата с линейной тактикой, автомата Кринского и автомата Роббинса при глубине памяти $q=1$ алгоритмы функционирования полностью совпадают, утрачивая отличия между ними, что будет проиллюстрировано при анализе

их функционирования.

Несколько другой подход к усилению инерционности применен в конструкции автомата Крылова.

Он сочетает в себе элементы поведения, как детерминированного, так и стохастического автомата.

При получении поощрения автомат функционирует точно также, как и автомат с линейной тактикой, увеличивая вплоть до самого глубокого номер состояния, соответствующего действию, за выполнение которого этот автомат получает поощрение от среды E .

При получении штрафа автомат Крылова «подбрасывает монетку».

При выпадении «орла» автомат не меняет своего состояния, а при выпадении «решки» он уменьшает номер текущего состояния, соответствующего действию, за которое получен штраф от среды.

Этот автомат также является асимптотически оптимальным в условиях стационарных сред.

Динамические среды

Первоначально формальным языком описания, как стационарных, так и динамических сред, с функционирующими в них автоматами, послужила теория игр.

Действия автомата из множества D взаимно однозначно соотносятся со стратегиями, доступными игроку в соответствующей (заданной) игре с конечным числом стратегий.

Например, совокупность из m действий, доступных автомату вместе со значениями вероятностей поощрений (штрафов) в стационарной среде E , на языке теории игр формулируется, как игра с природой:

- 1) задана одностолбцовая платежная матрица с m неотрицательными значениями;
- 2) элементы матрицы игры нормируются относительно максимального значения платежа и рассматриваются, как вероятности поощрений от стационарной среды E ;
- 3) значение поощрения равно единице, значение штрафа равно нулю;
- 4) действиям автомата (игрок 1) соответствуют номера строк заданной матрицы;
- 5) природа (игрок 2) выбрала свою стратегию (единственный столбец) и не меняет ее в течение всех партий игры;
- 6) целью игры является максимизация выигрыша автомата.

Игрок 1 (автомат), в каждый момент времени t разыгрывает партию игры с природой и, в зависимости от выбранного действия $d^*=d_i$ и соответствующей этому действию вероятности $p^*=p_i$, на входе получает сигнал поощрения (или штрафа) от окружающей среды I , $i=1, \dots, m$.

Поскольку оптимальная стратегия игры с природой известно и соответствует выбору игроком действия с максимальной ценой $p_i=p_{\max}$, то создается возможность сравнить эффективность действия осведомленного игрока, которому известна вся платежная матрица, с действиями автомата, которому содержимое этой матрицы не известно.

Из теорем об асимптотической оптимальности рассмотренных выше типов автоматов следует, что чем больше глубина памяти q автомата, тем точнее его совокупные действия соответствуют оптимальной стратегии.

Игры двух автоматов

Случай матричной игры двух игроков с нулевой суммой – это пример того, как один автомат способен создать среду для другого автомата.

В этом случае, действиям первого автомата сопоставляются номера строк, а действиям второго автомата номера столбцов платежной матрицы.

Если первый автомат с вероятностью p получает поощрение, то второй автомат получает штраф, и наоборот, реализуя принцип антагонистической игры с нулевой суммой.

Задача первого автомата состоит в максимизации поощрений, задача второго автомата – в минимизации штрафов.

Если игра допускает решение в чистых стратегиях, то автоматы реализуют друг для друга стационарную среду, для которой повышение целесообразности связано с ростом глубины памяти q .

Решение в смешанных стратегиях требует от автоматов оптимальной глубины памяти, удовлетворяющей свойствам данной динамической среды и типа автомата.

Аналогичным образом строится игра двух автоматов в случае бескоалиционных, неантагонистических, биматричных игр с конечным числом стратегий.

Игра в размещения

Примером того, как коллектив автоматов формирует динамическую среду для отдельного автомата, является игра в размещения.

Биологической предпосылкой для игры в размещения служит следующая задача:

1) имеется конечное число участков (пастбищ, или охотничьих территорий) с различной степенью производительности пищи. На этих участках некоторым образом размещаются животные;

2) если два и более животных в текущий момент времени оказываются на одном участке, то производительность пищи участка делится поровну между этими животными.

Задача животного состоит в максимизации количества добываемой им пищи.

Описание игры в размещения легко формулируется на языке теории игр следующим образом:

1) пусть задано m стратегий $d=d_1, \dots, d_m$. Значение каждой стратегии равно математическому ожиданию события «Поощрение / Штраф» в случае ее выбора;

2) в рамках заданных стратегий функционируют n автоматов: $n < m$. Выигрыш от стратегии $d_i, i=1, \dots, m$, в любой момент времени t поровну делится между всеми выбравшими эту стратегию автоматами;

3) требуется найти условия, обеспечивающие максимальный выигрыш для всей совокупности автоматов.

Эти условия делятся на три основные группы:

- типы исследуемых автоматов, определяющие структуру и особенности их поведения;
- значение глубины памяти q , определяющее степень инерционности автомата (скорость переключения между действиями);
- структурные изменения автоматов, направленные на организацию различных способов взаимодействия в коллективе.

Мы будем анализировать рассмотренные выше типы автоматов: автомат с линейной тактикой, автомат Кринского, автомат Роббинса и автомат Крылова.

Мы также проанализируем влияние глубины памяти q на степень инерционности автоматов, т.е. на скорость переключения автоматов между действиями в среде E .

Взаимодействие автоматов варьируется от полной изолированности автоматов друг от друга, до введения внешней по отношению к структуре автоматов процедуры общей кассы.

Оценка скорости переключения

Скорость переключения действий автомата определяет минимальное время, за которое автомат способен использовать все доступные стратегии.

Требуется определить необходимый интервал времени функционирования конкретного типа автомата для сбора достоверной статистики.

Поскольку все автоматы образуют однородный коллектив, и выбор действия каждым из n автоматов является случайным, то, начиная с некоторого момента времени $t=t^*$ суммы поощрений и штрафов автоматов по всему коллективу примут статистически близкие значения. В этой связи в качестве индикатора такого состояния системы «коллектив автоматов» целесообразно принять коэффициент вариации сумм поощрений и штрафов по всему коллективу.

Очевидно, что если коэффициент вариации не увеличивается, то коллектив автоматов принял статистически равновесное (не улучшаемое) состояние.

Вычислительные эксперименты показали, что наилучший результат достигается тогда, когда время функционирования системы, состоящей из n автоматов, определяется временем функционирования однотипного эталонного автомата, не принадлежащего исследуемому коллективу, т. е. не подверженного влиянию коллектива, но функционирующего в той же среде.

В этой связи рассмотрим сравнение времени функционирования отдельного автомата и автомата в коллективе.

Утверждение. Средняя скорость переключения между действиями одного из изолированных автоматов в игре в размещения не ниже значения этой скорости автомата, функционирующего в соответствующей стационарной среде.

Действительно, если у автомата при выборе текущего действия $d^*=d_i$, $i=1, \dots, m$, в текущей партии отсутствуют соседи, то его новое состояние будет обусловлено только значением отклика окружающей среды на выбор данного действия в соответствии с вероятностью получения поощрения $p^*=p_i$. В этом случае его скорость смены действий будет совпадать с скоростью автомата в соответствующей стационарной среде E .

Если при выборе данной стратегии $d^*=di$ у рассматриваемого автомата имеются соседи (один, или более одного) на i -ой площадке, то текущая ситуация распадается на два случая:

Если сигнал поощрения среды достался рассматриваемому автомату, то в текущей партии факт наличия соседей не влияет на его скорость переключения между действиями.

Если, наоборот, при получении от среды сигнала поощрения в рамках дележа пищи рассматриваемому автомату достаётся штраф, то такой сигнал может лишь уменьшить (по крайней мере, не увеличить) время его пребывания на текущей площадке.

Таким образом, средняя скорость переключения между действиями изолированного автомата в игре в размещения не меньше, чем скорость переключения автомата в соответствующей стационарной среде E .

На практике скорость переключения автомата в коллективе выше скорости переключения эталонного автомата.

Проведение вычислительного эксперимента

Для проведения игры в размещения задано 10 площадок. В качестве доступных для автоматов действий.

Величина поощрения равна (+1), величина штрафа (-1).

Эти величины определяют следующие значения вероятностей получения поощрений:

Табл. 1. Платежная матрица

Цена	0,0	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8	0,9
Поощрения	0,50	0,55	0,60	0,65	0,70	0,75	0,80	0,85	0,90	0,95

На первом этапе проведения серии вычислительных экспериментов потребовалось определить величины эффективности исследуемых типов автоматов и статистически достоверные интервалы времени для формирования этих величин.

В качестве опорной величины такого интервала времени было выбрано время (количество ходов) последовательного прохождения фиксированным автоматом с заданной глубиной памяти всех площадок (от наихудшей к наилучшей).

В качестве оценки эффективности функционирования заданного автомата целесообразно принять среднюю величину эффективности за ход, т. е. сумму поощрений и штрафов, полученных за время однократного прохождения автоматом всех площадок (выполнением всех доступных действий).

Фактически эта величина равна математическому ожиданию поощрений и штрафов автомата на всем множестве действий в случае однократного их выполнения.

Поскольку время пребывания автомата на конкретной площадке зависит от совокупного влияния конструкции автомата, текущей глубины памяти и последовательности штрафов и поощрений, обусловленной значениями генератора случайных чисел, то было принято решение построить серию из фиксированного числа независимых прохождений.

Это позволило:

1. Получить средние величины математических ожиданий рассматриваемых типов автоматов в соответствии со значениями глубины памяти.
2. Получить средние, статистически достоверные величины интервалов времени, требующиеся для выполнения автоматами всех доступных действий при условии, что в качестве минимальной единицы времени было выбрано значение одного хода, выполняемого автоматом.
3. Проверить корректность обработки результатов экспериментов при помощи соответствующей величины коэффициента вариации значений средней эффективности по всей серии экспериментов для каждого из четырех типов автоматов, которая не только не превышала 10 процентов, но с ростом глубины памяти вне зависимости от типа автомата неуклонно уменьшалась.

Для проведения серии вычислительных экспериментов потребовалось создание специализированного программного обеспечения, реализующего алгоритмы поведения исследуемых типов автоматов и окружающей среды по правилам игры в размещения, позволяющего задавать соответствующие типы автоматов, их характеристики (значение глубины памяти) и различные параметры окружающей среды, коллективы автоматов, как совокупности независимых агентов.

Для каждого типа автомата была проведена серия из 10 независимых экспериментов, после чего полученные данные усреднялись.

Для каждой серии экспериментов генератор случайных чисел запускался с фиксированного начального значения, порождая, для всех типов автоматов идентичную псевдослучайную стационарную среду E .

Обозначим типы автоматов следующим образом:

Автомат с линейной тактикой- АЛТ;

Автомат Кринского- АКРН;

Автомат Роббинса- АРББ;

Автомат Крылова- АКРЛ.

Табл. 2. Математические ожидания поощрений и штрафов

Автомат	Память 1	Память 2	Память 3	Память 4	Память 5	Память 6	Память 7	Память 8
АЛТ	0,63679	0,76809	0,80862	0,87499	0,88372	0,88475	0,79856	0,88803
АКРН	0,63679	0,76809	0,80474	0,87788	0,87707	0,88170	0,88255	0,87800
АРББ	0,63679	0,75081	0,80495	0,87277	0,87602	0,89240	0,89255	0,89688
АКРЛ	0,630618	0,760276	0,848816	0,884885	0,886225	0,89742	0,89549	—

Считая значение математического ожидания каждого автомата, как результат выполнения некоторого алгоритма, воспользуемся величиной среднего времени работы автомата, как оценкой быстродействия этого алгоритма для достижения полученного результата.

Табл. 3. Среднее время работы автомата

Автомат	Память 1	Память 2	Память 3	Память 4	Память 5	Память 6	Память 7	Г
АЛТ	74	695	3761	251394	3648854	19393688	927170104	224
АКРН	74	695	4527	264297	3795658		48562927	927
АРББ	74	759	5179	264297	3795658	53461387	1029193271	264
АКРЛ	116	2152	50814	2426164	89700530	3627473535	126294097025	—

Для анализа поведения автоматов в игре в размещения были выбраны коллективы, состоящие из 4 автоматов, а в качестве критерия оптимальности поведения была выбрана, как наиболее гибкая, сумма поощрений и штрафов по всему коллективу.

Дополнительно, такая величина оптимальности показывает среднестатистическое расположение автоматов по площадкам, т. Е. с ростом памяти автоматы стремятся разместиться на наилучших площадках, максимизируя суммы поощрений и минимизируя фактор конкуренции в коллективе, как отрицательный фактор.

Табл. 4. Интегральные оценки эффективности функционирования в игре в размещения.

Память	АЛТ	АКРН	АРББ	АКРЛ
1	1,4865	1,4865	1,4865	1,5135
2	2,2266	2,2266	2,1267	2,4860
3	2,6450	2,6662	2,5837	2,6814
4	2,8363	2,8012	2,7534	2,8846
5	2,9152	2,8955	2,8707	2,9365
6	2,9541	2,9331	2,9195	2,9631
7	2,9724	2,9561	2,9500	—

Из предыдущей таблицы мы можем видеть среднее время (количество ходов), требуемое для получения указанного результата коллективом, так как в качестве времени функционирования коллектива выступает однотипный эталонный автомат.

Автомат Крылова [\[10\]](#) располагает ярко выраженным механизмом инерционности поведения, обеспечивающим, как длительное (по сравнению с другими автоматами) пребывание в наилучшем состоянии, так и наиболее медленное приближение к наилучшему размещению коллектива в целом.

Заключение

Рассмотренные выше системы (коллективы), состоящие из достаточно простых автоматов, обладают свойством целесообразного поведения. Целесообразное поведение таких систем наблюдается уже при небольшом количестве входящих в них автоматов. Разнообразие поведения также достигается уже при незначительных изменениях в способах обработки входных сигналов от среды.

Нами были рассмотрены 4 типа автоматов.

Выявлены две автоматные стратегии:

1. За счет высокой скорости переключения посетить все доступные площадки (автомат с линейной тактикой);

2. за счет высокой инерционности длительное время оставаться в наилучшем состоянии (автомат Крылова).

Однако, второй результат достигается существенно длительным периодом времени для перехода коллектива в оптимальное статистически устойчивое состояние, что при сравнительно небольших значениях глубины памяти приводит к значительным вычислительным затратам, что иллюстрирует таблица средних периодов времени функционирования эталонных автоматов.

Суммы поощрений и штрафов численно равны суммам математических ожиданий наилучших площадок, занимаемых автоматами, следовательно, дальнейшее усложнение конструкций автоматов, т. е. рост глубины памяти не может привести к значительному увеличению эффективности функционирования коллектива в целом, так как разница между возможным наилучшим значением и достигнутым результатом составляет в среднем не более 10- 20 процентов от возможного при значительном увеличении времени (модельного и реального) функционирования.

Полученные результаты могут быть положены в основу формирования оптимальных по сложности и вычислительным затратам коллективов автоматов для решения более сложных оптимизационных задач.

К таким задачам относятся:

1. Организация различных форм информационного взаимодействия между отдельными автоматами с целью повышения эффективности функционирования коллектива в целом. Такое влияние характеризуется достижением заданного (уже известного) уровня эффективности функционирования коллектива автоматов с меньшим требуемым значением глубины памяти, приводящим к более низким вычислительным и временным затратам при проведении вычислительных экспериментов, так как коллективы синхронно функционирующих автоматов плохо поддаются процессу распараллеливания общего потока вычислений. Фактор временных затрат особенно ощущается, например, при исследовании поведения автомата Крылова уже при сравнительно небольших значениях глубины памяти;
2. внесение в каждый из рассмотренных типов автоматов структурных изменений, приводящих коллектив автоматов к формированию процедуры общей кассы, как внутреннего свойства коллектива без введения такой внешней централизованной процедуры;
3. оценка влияния вносимых изменений на эффективность функционирования каждого из рассмотренных типов автоматов;
4. формирование автоматной реализации роевых (муравьиных) алгоритмов, характеризующихся согласованным движением всего коллектива к нахождению экстремума целевой функции в условиях отсутствия централизованного управления согласованными действиями отдельных агентов (автоматов).

Библиография

1. Стефанюк В.Л. Локальная организация интеллектуальных систем. М.: Физматлит. 2004. С. 328.
2. Гаазе-Рапопорт М.Г., Поспелов Д.А. От амебы до робота. Модели поведения. М.: Ленанд. 2019. С. 304.
3. Журавлев А.Л., Савченко Т.Н., Головина Г.М. Математическая психология: школа В. Ю. Крылова-Сер. Научные школы Института психологии РАН. М.: Изд-во ИП РАН. 2010. С.

512.

4. Жданов А.А. Автономный искусственный интеллект. М.: Лаборатория знаний. 2024. С. 362.

5. Димитриченко Д.П. Оптимизация рекуррентной нейронной сети при помощи автоматов с переменной структурой // Программные системы и вычислительные методы. 2023. № 4. С.30-43. DOI: 10.7256/2454-0714.2023.4.69011 EDN: FEIPTC URL: https://e-notabene.ru/itmag/article_69011.html

6. Карпов В.Э., Карпова И.П., Кулинич А.А. Социальные сообщества роботов. М.: УРСС. 2019. С. 352.

7. Карпов В.Э., Королева М.Н. К вопросу о формализации этики поведения коллаборативного робота // Информационные и математические технологии в науке и управлении. 2022. № 4 (28). С. 223-233.

8. Пospelов Д.А. Игры и автоматы. М.: Энергия. 1966. С. 136.

9. Цетлин М.Л. Исследования по теории автоматов и моделированию биологических систем. М.: Наука. 1969. С. 316.

10. Пospelов Д.А. Вероятностные автоматы. М.: Энергия. 1970. С. 88.

11. Варшавский В.И. Коллективное поведение автоматов. М.: Наука. 1973. С. 408.

12. Варшавский В.И., Пospelов Д.А. Оркестр играет без дирижера: размышления об эволюции некоторых технических систем и управление ими. М.: Наука. 1984. С. 208

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Рецензируемая статья посвящена сравнительному анализу целесообразного коллективного поведения четырех типов автоматов в условиях игры в размещения.

Методология выполненной работы базируется на использовании подходов и методов из теорий игр, массового обслуживания и машинного обучения, сопряжена с применением автоматных моделей и моделированием коллективного поведения при помощи автоматов, которые получили название игры в размещения, а также на описании проведения вычислительного эксперимента.

Актуальность работы определяется тем, что организация сложного поведения децентрализованных систем находит свое отражение в таком разделе машинного обучения, как «Коллективное поведение автоматов», имеющем прикладное значение в решении задач автоматной оптимизации, интеллектуального управления, построении робототехнических систем, а также в математической психологии.

Научная новизна рецензируемого исследования, к сожалению, авторами не сформулирована.

Структурно в статье выделены следующие разделы: Введение, Автоматная реализация; Общая постановка задачи, Биологические предпосылки, Типы конструкций автоматов, Динамические среды, Игры двух автоматов, Игра в размещения, Оценка скорости переключения, Проведение вычислительного эксперимента, Заключение и Библиография.

Авторами приведены описания автоматов Цетлина, Кринского, Роббинса, Крылова. Эти 4 типа автоматов рассмотрены и при проведении вычислительного эксперимента. Были выявлены две автоматные стратегии: во-первых, за счет высокой скорости переключения посетить все доступные площадки (автомат с линейной тактикой); во-вторых, за счет высокой инерционности длительное время оставаться в наилучшем состоянии (автомат Крылова). В публикации по итогам вычислительного эксперимента

сделан вывод о том, что автомат Крылова располагает ярко выраженным механизмом инерционности поведения, обеспечивающим, как длительное (по сравнению с другими автоматами) пребывание в наилучшем состоянии, так и наиболее медленное приближение к наилучшему размещению коллектива в целом.

Библиографический список включает 10 источников – научные публикации отечественных авторов по рассматриваемой теме на русском языке, опубликованные за период с 1966 по 2023 годы. В тексте публикации имеются адресные отсылки к списку литературы, подтверждающие наличие апелляции к оппонентам.

Из резервов улучшения работы, следует указать следующие. Во-первых, в вводной части публикации уместно в более доступной для широкого круга читателей форме сформулировать актуальность рассматриваемых вопросов и сферу их применения. Во-вторых, после ознакомления со статьей не ясно, каким образом, с использованием каких инструментальных средств были получены значения времени работы автоматов, на основе которых определены средние величины. В-третьих, в публикации не сформулирована цель работы и не отражены полученные авторами элементы приращения научного знания, научная новизна работы не раскрыта. В-четвертых, при описании практического применения полученных результатов исследования не стоит ограничиваться общей фразой: «Полученные результаты могут быть положены в основу формирования оптимальных по сложности и вычислительным затратам коллективов автоматов для решения более сложных оптимизационных задач», а перечислить хотя бы некоторые такие задачи.

Рецензируемый материал соответствует направлению журнала «Программные системы и вычислительные методы», отражает результаты проведенного авторского исследования, может вызвать интерес у читателей, но материал нуждается в доработке в соответствии с высказанными замечаниями.

Результаты процедуры повторного рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Представленная статья на тему «Анализ целесообразного поведения различных типов автоматов в условиях игры в размещения» соответствует тематике журнала «ПРОГРАММНЫЕ СИСТЕМЫ И ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ» и посвящена актуальному вопросу, связанного с ростом вычислительных возможностей многопользовательских и многопоточных систем возникает объективная потребность в анализе и управлении совокупностью агентов действующих децентрализованно, в соответствии с собственным целеполаганием и взаимодействующих друг с другом и заданной окружающей средой.

В статье представлен широкий анализ литературных российских источников по вопросам применения автоматного подхода, предоставляющего в распоряжение исследователя формализованного (в рамках дискретной математики) метода, позволяющего выполнить формальную постановку задачи и проанализировать поведение агентов и самой заданной среды обитания в терминах автоматной модели (коллектива автоматов), входных, выходных и внутренних алфавитов, а также правил перехода и результатов выбора.

Авторами статьи самостоятельно проведен комплексный анализ систем (коллективов), состоящих из достаточно простых автоматов, обладающих свойством целесообразного поведения. Авторами были рассмотрены 4 типа автоматов и выявлены две автоматные стратегии: за счет высокой скорости переключения посетить все доступные площадки (автомат с линейной тактикой); за счет высокой инерционности длительное время

оставаться в наилучшем состоянии (автомат Крылова).

Стиль и язык изложения материала является достаточно доступным для широкого круга читателей. Статья по объему соответствует рекомендуемому объему от 12 000 знаков.

Статья достаточно структурирована - в наличии введение, заключение, внутреннее членение основной части (авторами рассмотрены вопросы: Автоматная реализация, Биологические предпосылки, Стационарная окружающая среда, Типы конструкций автоматов, Динамические среды и др.).

Авторами проведена серия вычислительных экспериментов для которых потребовалось создание специализированного программного обеспечения, реализующего алгоритмы поведения исследуемых типов автоматов и окружающей среды по правилам игры в размещения, позволяющего задавать соответствующие типы автоматов, их характеристики (значение глубины памяти) и различные параметры окружающей среды, коллективы автоматов, как совокупности независимых агентов. Для каждого типа автомата авторами была проведена серия из десяти независимых экспериментов, после чего полученные данные усреднялись.

Для каждой серии экспериментов генератор случайных чисел запускался с фиксированного начального значения, порождая, для всех типов автоматов идентичную псевдослучайную стационарную среду E.

Практическая значимость четко обоснована.

К недостаткам можно отнести следующие моменты: отсутствует четкое выделение предмета, объекта.

Рекомендуется четко сформулировать предмет, объект. Также будет целесообразным добавить о перспективах дальнейшего исследования.

Статья «Анализ целесообразного поведения различных типов автоматов в условиях игры в размещения» требует доработки по указанным выше замечаниям. После внесения поправок рекомендуется к повторному рассмотрению редакцией рецензируемого научного журнала.

Программные системы и вычислительные методы

Правильная ссылка на статью:

Захаров А.А. Метод обнаружения объектов на изображениях на основе нейронных сетей на графах и небольшого количества обучающих примеров // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.72558 EDN: UTTFCH URL: https://nbpublish.com/library_read_article.php?id=72558

Метод обнаружения объектов на изображениях на основе нейронных сетей на графах и небольшого количества обучающих примеров

Захаров Алексей Александрович

кандидат технических наук

доцент; кафедра программной инженерии; Муромский институт (филиал) федерального государственного бюджетного образовательного учреждения высшего образования «Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»

ведущий научный сотрудник; Муромский институт (филиал) федерального государственного бюджетного образовательного учреждения высшего образования «Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых»

602264, Россия, Владимирская область, г. Муром, ул. Орловская, 23, ауд. 402

✉ aa-zaharov@ya.ru



[Статья из рубрики "Компьютерная графика, обработка изображений и распознавание образов"](#)

DOI:

10.7256/2454-0714.2024.4.72558

EDN:

UTTFCH

Дата направления статьи в редакцию:

03-12-2024

Дата публикации:

11-12-2024

Аннотация: В представленной работе объектом исследования являются системы компьютерного зрения. Предмет исследования – метод обнаружения объектов на изображениях на основе нейронных сетей на графах и небольшого количества обучающих примеров. Подробно рассматриваются такие аспекты темы как использование структурного представления сцены для повышения точности обнаружения объектов. Предлагается совместное использование информации о структуре сцены на основе нейронных сетей на графах и обучения с «нескольких

выстрелов» для повышения точности обнаружения объектов. Устанавливаются отношения между классами с помощью внешних семантических связей. Для этого предварительно создаётся граф знаний. Метод содержит два этапа. На первом этапе выполняется обнаружение объектов на основе обучения с «нескольких выстрелов». На втором этапе выполняется повышение точности обнаружения с использованием нейронной сети на графах. Основой разрабатываемого метода является использование свёртки на основе спектральной теории графов. Каждая вершина представляет собой категорию в графе знаний, а вес ребра графа рассчитывается на основе условной вероятности. На основе свёртки объединяется информация из соседних вершин и рёбер для обновления значений вершин. Научная новизна разработанного метода заключается в совместном использовании свёрточных сетей на графах и обучения с «нескольких выстрелов» для увеличения точности обнаружения объектов. Особым вкладом автора в исследование темы является применение свёрточной сети на основе графа знаний для улучшения результатов работы метода обнаружения объектов при использовании малого количества обучающих примеров. Метод исследовался на тестовых наборах изображений из области компьютерного зрения. Используя наборы данных PASCAL VOC и MS COCO продемонстрировано, что предлагаемый метод увеличивает точность обнаружения объектов за счет анализа структурных взаимосвязей. Средняя точность обнаружения объектов при использовании разработанного метода увеличивается на 1-5% по сравнению с методом обучения с «нескольких выстрелов» без использования структурного представления.

Ключевые слова:

компьютерное зрение, обнаружение объектов, свёрточные сети, небольшой набор данных, глубокое обучение, ограниченная аннотация, граф, распознавание образов, искусственный интеллект, структурное представление сцен

Исследование выполнено за счет гранта Российского научного фонда № 23-21-00486, <https://rscf.ru/project/23-21-00486/>

Введение

Обнаружение объектов – это важная задача компьютерного зрения, которая заключается в том, чтобы найти интересующий объект на входном изображении, а затем точно отнести его к определенному классу. Наиболее значимыми характеристиками результатов обнаружения объектов являются точность локализации и классификации, а также скорость обнаружения. Обнаружение объектов служит основой для многих других областей: автономная навигация, человеко-машинные интерфейсы, контроль технологических процессов, дистанционное зондирование Земли, медицинская диагностика, биометрия, видеонаблюдение и т.д.

В последние годы быстрое развитие методов глубокого обучения значительно способствовало прогрессу в области обнаружения объектов. В настоящее время разработано большое количество методов обнаружения объектов с использованием нейронных сетей. Методы обнаружения объектов на основе глубокого обучения принято делить на одноэтапные и двухэтапные [1]. К одноэтапным методам относятся YOLO [2], SSD [3], RetinaNet [4] и др. К двухэтапным методам принадлежат Fast R-CNN [5], Faster R-CNN [6], Mask R-CNN [7] и др.

Следует отметить, что методы обнаружения объектов на основе глубокого обучения сталкиваются со следующими критическими проблемами:

- многообразие сцен при реальных наблюдениях. Большое количество методов обнаружения объектов достаточно хорошо работает в лабораторных условиях: равномерное освещение, однородный статичный фон, отсутствие движения камеры и т.д. Однако в реальных условиях наблюдения точность обнаружения объектов значительно уменьшается. Это связано с наличием сложного текстурированного фона и низкого контраста изображений, присутствием посторонних подвижных объектов, наличием взаимных перекрытий, затененных областей и шумов.
- необходимость разметки больших наборов изображений ручным способом. Ключевым компонентом революции в области глубокого обучения была доступность больших аннотированных наборов данных. Несмотря на то, что большинство наборов данных компьютерного зрения маркируются с помощью краудсорсинга, этот процесс по-прежнему является дорогостоящим и требует много времени, что становится узким местом при развертывании систем глубокого обучения. Многие существующие методы обнаружения, основанные на глубоком обучении, показывают в лабораторных условиях хорошие результаты с использованием больших наборов данных. Однако эти методы труднореализуемы в реальных условиях из-за невозможности создания больших аннотированных наборов данных.
- необходимость обнаружения объектов по категориям, у которых количество экземпляров в обучающем наборе очень мало. Часто возникает проблема обнаружения объектов по категориям, у которых нет экземпляров в обучающем наборе или их количество ограничено. Если количество обучающих примеров слишком мало, по сравнению со всеми возможными вариациями, то возникает проблема обучения с использованием небольшой выборки.

Таким образом, одной из основных проблем при реализации методов обнаружения объектов на основе глубокого обучения является необходимость создания большого объема аннотированных данных, что не всегда возможно по экономическим и техническим причинам.

В последние годы активно разрабатываются методы обнаружения объектов на основе обучения с «нескольких выстрелов» (few-shot learning), которые пытаются решить задачу с использованием небольшого количества примеров [\[8\]](#). Часто бывает, что данные базового класса ограничены несколькими примерами для обучения. В этом случае модель предварительно обучается на крупномасштабном наборе данных из другой области. Основное же назначение нескольких примеров – это адаптация представления к предметной области [\[9\]](#). Таким образом, для обучения модели требуется значительно меньше маркированных данных.

Известны следующие методы обнаружения объектов, основанные на обучении с «малым количеством выстрелов»: Multi-Scale Positive Sample Refinement for Few-Shot Object Detection (MPSR) [\[10\]](#), Frustratingly Simple Few-Shot Object Detection (TFA) [\[11\]](#), Few-Shot Object Detection via Feature Reweighting (MetaYOLO) [\[12\]](#) и др. Однако точность методов обнаружения с использованием небольшого количества обучающих данных остается низкой.

Предлагается использовать структурное представление сцены для повышения точности обнаружения объектов. Для описания структуры будут использоваться графы.

Особенности графа позволяют оценить связи между элементами изображения. Графы используются в различных областях компьютерного зрения для сегментации изображений [13], обнаружения значимых областей [14], кластеризации [15] и др. Возможности графов позволяют анализировать структурные отношения между объектами сцен, что даёт возможность собирать больше информации по сравнению с локальным анализом данных.

В последние годы изучение графов быстро продвинулось вперед благодаря наличию больших наборов данных, мощных вычислительных ресурсов, а также достижениям в области машинного обучения и искусственного интеллекта [16]. Методы глубокого обучения могут эффективно кодировать и представлять данные о графах в виде векторов. Эти векторы затем можно использовать в различных задачах с высокой производительностью. Нейронная сеть на графах (Graph Neural Network, GNN) – это архитектура глубокого обучения, специально разработанная для данных, описанных при помощи графов [17-20]. В отличие от традиционных алгоритмов глубокого обучения, которые в первую очередь были разработаны для текста и изображений, GNN созданы специально для обработки и анализа структурированных наборов данных.

Целью исследования является разработка метода, повышающего точность обнаружения объектов. Предполагается, что при использовании структурного представления точность обнаружения объектов на основе небольшого количества обучающих примеров повысится.

Научная новизна разработанного метода заключается в совместном использовании нейронных сетей на графах и обучения с «нескольких выстрелов» для увеличения точности обнаружения объектов.

Разработка метода обнаружения объектов на изображениях на основе нейронных сетей на графах и небольшого количества обучающих примеров

Разработанный метод включает два этапа.

Этап 1. Обнаружение объектов с использованием обучения с «нескольких выстрелов» [12].

Этап 2. Повышение точности обнаружения с использованием нейронной сети на графах. Структурная схема метода представлена на рисунке 1.

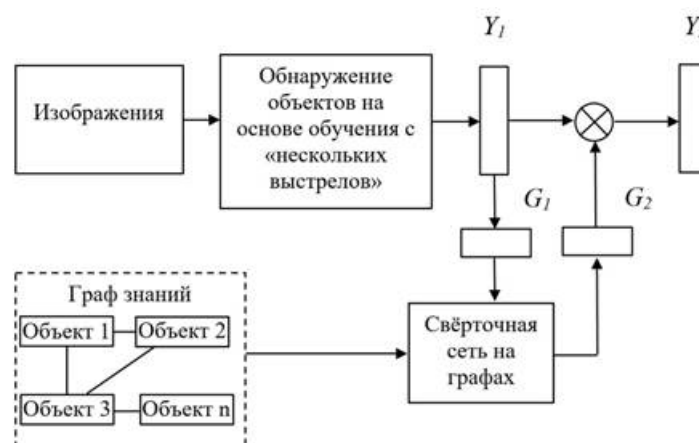


Рис. 1. Структурная схема метода обнаружения объектов на изображениях на основе нейронных сетей на графах и небольшого количества обучающих примеров

На этапе 1 происходит обнаружение объектов с использованием обучения с «нескольких выстрелов». Далее на основе обнаруженных ограничивающих прямоугольников объектов строится матрица вероятностей Y_1 :

$$Y_1 = B \times C,$$

где B – количество обнаруженных ограничивающих областей, C – количество классов, Y_1 – вероятность c -ого класса b -ой ограничивающей области.

На этапе 2 осуществляется повышение точности обнаружения с использованием нейронной сети на графах. Связи между объектами задаются с использованием графа знаний. Граф знаний представляет собой семантическую сеть, которая хранит информацию о различных классах объектов и связях между ними. Граф знаний, описывающий связи между классами объектов, показан на рис. 2.

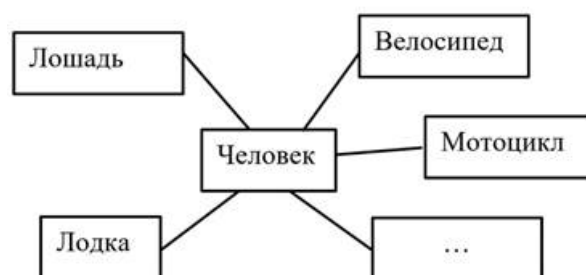


Рис.2. Граф знаний, описывающий связи между классами объектов

Предполагается, что совместное присутствие на рисунке таких объектов, как человек и велосипед, человек и мотоцикл, человек и лодка и т.д. поможет повысить точность обнаружения с использованием имеющихся связей (рис. 3).

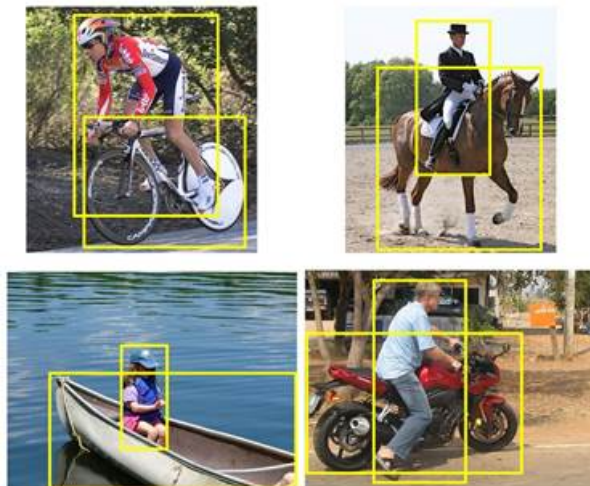


Рис. 3. Совместное присутствие на изображениях объектов различных классов

В графе знаний каждая вершина представляет некоторый класс, а ребро графа от вершины V_1 до вершины V_2 является условной вероятностью [21]

$$P(V_2 | V_1).$$

Например, если человек и велосипед появляются вместе в наборе данных 10 раз, а всего человек появляется в наборе данных 20 раз, то ребро от вершины класса «человек» до вершины класса «велосипед» будет иметь значение

$$P (\text{человек/ велосипед}) = 0,5.$$

Граф знаний описывается матрицей смежности $A = \mathbb{R}^{n \times n}$, где n — количество классов, представленных графом.

На вход свёрточной сети на графах подается следующий вектор:

$$G_{1,c} = \max_{b=1, 2, \dots, B} (Y_{1,bc}), \quad c=1, 2, C,$$

где $G_{1,c}$ — значение максимальной вероятности c -ого класса среди всех обнаруженных ограничивающих прямоугольников.

В задаче классификации узлов GNN использует информацию для создания векторного представления каждого узла в графе. Такое представление включает в себя не только исходные характеристики вершины, но также информацию о связях между вершинами. Вместо того, чтобы ограничиваться исходными атрибутами, GNN добавляет к свойствам исходных вершин атрибуты из соседних вершин и ребер, что делает представление гораздо более полным и значимым. Новые представления вершин затем используются для выполнения конкретных задач, таких как классификация вершин, регрессия или прогнозирование связей. В частности, GNN определяет операцию свёртки графа, которая объединяет информацию из соседних вершин и рёбер для обновления представлений. Эта операция выполняется итеративно, что позволяет модели изучать более сложные взаимосвязи между вершинами по мере увеличения количества итераций (рис. 4). В работе сеть содержит четыре свёрточных слоя.

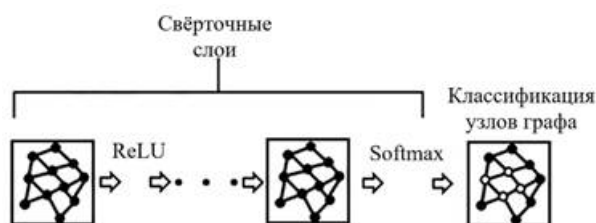


Рис. 4. Классификация узлов графа на основе свёрточной сети

Свёрточная сеть на графах описывается следующим правилом потока информации между слоями [17]:

$$H_{l+1} = \text{ReLU}((aD^{-1} + I)H_l W_l) + B_l,$$

где I — единичная матрица, H_l — активационная матрица слоя l , H_{l+1} — активационная матрица слоя $l+1$, D — степеньная матрица графа, W_l — матрица весов, B_l — переменная для управления средним значением выходного сигнала, a — настраиваемый параметр, который используется для определения влияния связанных узлов. По умолчанию $a=0.5$.

Результат работы нейронной сети на графах представлен в виде матрицы корректирующих весов:

$$G_2 = \text{GNN}(G_1).$$

Окончательный результат представляет собой поэлементное произведение исходной матрицы Y_1 и матрицы корректирующих весов G_2 :

$Y_2=Y_1G_2.$

Исследование метода

Для проведения исследования использовались изображения из набора данных PASCAL VOC. Для каждой категории отбирались *K* экземпляров: *K* = 1, 3, 5, 10.

Была рассчитана средняя точность (Mean average precision):

$mAP=(SUM_{k=1...n}(AP_k))/n,$

..

где *n* – количество классов, *AP_k*– средняя точность класса *k*

Результаты экспериментов были сведены в таблицу 1.

Таблица 1. Расчет показателя *mAP* для разного количества экземпляров

Метод обнаружения объектов	<i>mAP</i> (%)			
	1 экземпляр	3 экземпляра	5 экземпляров	10 экземпляров
Meta-RCNN (few-shot)	17.38	25.92	32.63	48.35
Meta-RCNN (few-shot) + нейронная сеть на графах	18.89	28.78	37.51	53.78

В процессе обучения нейронной сети на графах были рассчитаны потери. Нейронная сеть на графах обучалась в течение 500 эпох (рис. 5).

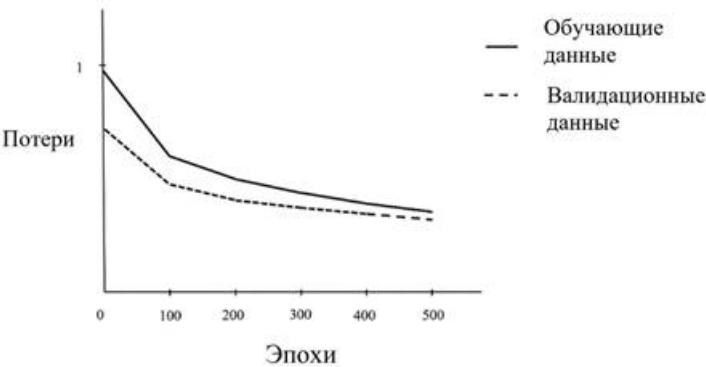


Рис. 5. График функции потерь при обучении нейронной сети на графах

Для оптимизации применялся алгоритм ADAM. ADAM позволяет настраивать скорость обучения для каждого параметра. Для обучения модели использовался графический процессор GeForce RTX 3060. Таким образом, разработанный метод позволяет повысить среднюю точность на 1-5% за счет анализа структурных связей между объектами.

Заключение

В статье предложен метод, который частично компенсирует недостатки метода обучения

с «малым количеством выстрелов» при обнаружении объектов. Метод основан на использовании нейронной сети на графах для описания структуры анализируемой сцены. Эксперименты показали эффективность разработанного метода. Средний показатель точности обнаружения объектов увеличился до пяти процентов. Также было показано, что предложенный метод позволяет достичь еще большей точности при увеличении объема обучающих данных.

Библиография

1. Zou Z., Chen K., Shi Z., Guo Y., Ye J. Object Detection in 20 Years: A Survey // Proceedings of the IEEE. 2023. Vol. 111 (3). Pp. 257-276.
2. Redmon J., Divvala S., Girshick R., Farhadi A. You only look once: Unified, real-time object detection // IEEE Conference on Computer Vision and Pattern Recognition. 2016. Pp. 779-788.
3. Liu W., Anguelov D., Erhan D., Szegedy C., Reed S., Fu C.Y., Berg A. C. Ssd: Single shot multibox detector // European Conference on Computer Vision. 2016. Pp. 21-37.
4. Lin T.Y., Goyal P., Girshick R., He K., Dollar P. Focal loss for dense object detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2018. Vol. 42(2). Pp. 318-327.
5. Girshick P. Fast R-CNN // 2015 IEEE International Conference on Computer Vision (ICCV). 2015. Pp. 1440-1448.
6. Ren S., He K., Girshick R., Sun J. Faster R-CNN: Towards real-time object detection with region proposal networks // Advances in Neural Information Processing System. 2015. Pp. 91-99.
7. He K., Gkioxari G., Dollar P., Girshick R. Mask R-CNN // Proceedings of the IEEE International Conference on Computer Vision. 2017. Pp. 2961- 2969.
8. Köhler M., Eisenbach M., Gross H. M. Few-Shot Object Detection: A Survey // IEEE Transactions on Neural Networks and Learning Systems. 2024. Vol. 35 (9). Pp. 11958 – 11978.
9. Huang G., Laradji I., Vazquez D., Lacoste-Julien S., Rodriguez P. A Survey of Self-Supervised and Few-Shot Object Detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2023. Vol. 45(4). Pp. 4071-4089.
10. Wu J., Liu S., Huang D., Wang Y. Multi-scale positive sample refinement for few-shot object detection // European Conference on Computer Vision. 2020. Pp. 456-472.
11. Wang X., Huang T. E., Gonzalez J., Darrell T., Yu F. Frustratingly simple few-shot object detection // Proceedings of the 37th International Conference on Machine Learning (ICML). 2020. Pp. 9919-9928.
12. Kang B., Liu Z., Wang X., Yu F., Feng J., Darrell T. Few-shot object detection via feature reweighting // 2019 IEEE/CVF International Conference on Computer Vision. 2019.
13. Захаров А.А., Тужилкин А.Ю. Сегментация спутниковых изображений на основе суперпикселей и разрезов на графах // Программные системы и вычислительные методы. 2018. № 1. С.7-17. DOI: 10.7256/2454-0714.2018.1.25629 URL: https://e-notabene.ru/itmag/article_25629.html
14. Захаров. А.А., Титов Д.В., Жизняков А.Л., Титов В.С. Метод визуального внимания на основе ранжирования вершин графа по разнородным признакам изображений // Компьютерная оптика. 2020. Т. 44, № 3. С. 427-435.
15. Barinov A.E., Zakharov A.A. Clustering using a random walk on graph for head pose estimation // International Conference on Mechanical Engineering, Automation and Control Systems, MEACS. 2015.
16. Cao P., Zhu Z., Wang Z., Zhu Y., Niu Q. Applications of graph convolutional networks in computer vision // Neural Computing and Applications. 2022. № 34. Pp. 13387-13405.

17. Kipf T.N. Deep Learning with Graph-Structured Representations, Universiteit van Amsterdam, 2020.
18. Li W., Liu X., Yuan Y. SIGMA++: Improved Semantic-Complete Graph Matching for Domain Adaptive Object Detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2023. Vol.45 (7). Pp. 9022-9040.
19. Chen C., Li J., Zhou H.Y., Han X., Huang Y., Ding X., Yu Y. Relation matters: Foreground-aware graph-based relational reasoning for domain adaptive object detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2023. Vol. 45 (3). Pp. 3677-3694.
20. Chen T., Lin L., Chen R., Hui X., Wu X. Knowledge-Guided Multi-Label Few-Shot Learning for General Image Recognition // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2022. Vol.44 (3). Pp.1371-1384.
21. Liu Z., Jiang Z., Feng W., Feng H. OD-GCN: Object Detection Boosted by Knowledge GCN // 2020 IEEE International Conference on Multimedia & Expo Workshops (ICMEW). 2020

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Представленная статья на тему «Метод обнаружения объектов на изображениях на основе нейронных сетей на графах и небольшого количества обучающих примеров» соответствует тематике журнала «Программные системы и вычислительные методы» и посвящена вопросу обнаружения объектов, что является важной задачей компьютерного зрения, которая заключается в том, чтобы найти интересующий объект на входном изображении, а затем точно отнести его к определенному классу. Наиболее значимыми характеристиками результатов обнаружения объектов являются точность локализации и классификации, а также скорость обнаружения. Обнаружение объектов служит основой для многих других областей: автономная навигация, человеко-машинные интерфейсы, контроль технологических процессов, дистанционное зондирование Земли, медицинская диагностика, биометрия, видеонаблюдение и т.д.

В статье представлен широкий анализ литературных российских и зарубежных источников, посвященных методам обнаружения объектов с использованием нейронных сетей, в том числе одноэтапных (YOLO, SSD, RetinaNet и др) и двухэтапных (Fast R-CNN, Faster R-CNN, Mask R-CNN и др.); методам обнаружения объектов на основе обучения с «нескольких выстрелов» (few-shot learning), которые пытаются решить задачу с использованием небольшого количества примеров; методам обнаружения объектов, основанным на обучении с «малым количеством выстрелов» (Multi-Scale Positive Sample Refinement for Few-Shot Object Detection (MPSR), Frustratingly Simple Few-Shot Object Detection (TFA).

Авторами статьи предложен метод, основанный на использовании нейронной сети на графах для описания структуры анализируемой сцены, проведен эксперимент по совместному использованию нейронных сетей на графах и обучения с «нескольких выстрелов» для увеличения точности обнаружения объектов. По заявлению авторов эксперимент показал эффективность разработанного метода. Средний показатель точности обнаружения объектов увеличился до пяти процентов. Также было показано, что предложенный метод позволяет достичь еще большей точности при увеличении объема обучающих данных.

Стиль и язык изложения материала является достаточно доступным для широкого круга читателей. Статья по объему соответствует рекомендуемому объему от 12 000 знаков.

Сформулирована цель исследования и научная новизна, которая заключается в совместном использовании нейронных сетей на графах и обучения с «нескольких выстрелов» для увеличения точности обнаружения объектов. Практическая значимость прослеживается из содержания статьи и заключается в повышении точности обнаружения объектов на изображениях.

Статья достаточно структурирована - в наличии введение, заключение, внутреннее членение основной части. Было бы целесообразным добавить о перспективах дальнейшего исследования.

Статья «Метод обнаружения объектов на изображениях на основе нейронных сетей на графах и небольшого количества обучающих примеров» рекомендована к публикации в журнале «Программные системы и вычислительные методы».

Программные системы и вычислительные методы

Правильная ссылка на статью:

Князев М.А., Шаброва А.С., Крючков А.А. Подход к выбору механизмов защиты устройств персонального Интернета вещей на основе математической модели с двумя критериями // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.72839 EDN: ZOSMZM URL: https://nbpublish.com/library_read_article.php?id=72839

Подход к выбору механизмов защиты устройств персонального Интернета вещей на основе математической модели с двумя критериями

Князев Максим Андреевич

ORCID: 0009-0007-3931-7442

аспирант, кафедра информационной безопасности ; ФГБОУ ВО «МИРЭА – Российский технологический университет», Институт Искусственного Интеллекта

119454, Россия, г. Москва, пр-т Вернадского, 78

✉ maxiknyaz@mail.ru



Шаброва Анна Сергеевна

ORCID: 0009-0009-1675-1558

студент, кафедра Информационная безопасность; МГТУ им. Н.Э. Баумана

105005, Россия, г. Москва, ул. 2-я Бауманская, 5, стр. 4

✉ shabrova.anna.2410@list.ru



Крючков Андрей Андреевич

ORCID: 0009-0002-4750-6204

старший преподаватель; кафедра информационной безопасности; ФГБОУ ВО «МИРЭА – Российский технологический университет»

119454, Россия, г. Москва, пр-т Вернадского, 78

✉ kryuchkov_a@mirea.ru



[Статья из рубрики "Кодирование и защита информации"](#)

DOI:

10.7256/2454-0714.2024.4.72839

EDN:

ZOSMZM

Дата направления статьи в редакцию:

19-12-2024

Дата публикации:

26-12-2024

Аннотация: Существующие методы защиты устройств персонального Интернета вещей (PIoT) требуют постоянной и непрерывной модернизации с учетом возможного возникновения новых угроз и уязвимостей. Важной и актуальной задачей при этом является разработка универсального и эффективного подхода к обеспечению безопасности таких устройств, учитывающего ограниченность ресурсов производителей потребительской электроники сегмента IoT. В данном исследовании предлагается использование математической модели с критериями сложности реализации и универсальности механизмов защиты для выполнения ранжирования механизмов защиты с целью повышения защищённости портативных умных устройств при их оптимальной реализации, с учётом затрат разработчика и в соответствии с требованиями действующего законодательства в области информационной безопасности. Объектом исследования данной работы является процесс обеспечения информационной безопасности устройств персонального Интернета вещей, учитывающий существующие нормативные и технические требования, а также ограниченность ресурсов производителей и разработчиков. Предметом исследования выступает совокупность механизмов защиты PIoT-устройств, отобранных и ранжируемых на основе разработанной математической модели с двумя критериями. Предложен подход к выбору механизмов защиты устройств персонального Интернета вещей на основе математической модели с критериями сложности реализации и универсальности механизмов защиты. В рамках представленного исследования был проведен подробный анализ рекомендаций и требований к обеспечению безопасности устройств персонального Интернета вещей в международных и отечественных стандартах и исследованы возможности их реализации при эффективном распределении ресурсов производителя посредством математической модели с двумя критериями. Научная новизна данной исследовательской работы заключается в том, что был предложен оригинальный подход к выбору механизмов защиты PIoT-устройств на основе математической модели с двумя критериями, позволяющий при минимизации затрат на разработку и эксплуатацию эффективно учитывать актуальные угрозы и нормативные требования. В результате проведенного исследования были сделаны выводы о том, что внедрение подхода к выбору механизмов защиты устройств персонального Интернета вещей на основе математической модели с критериями сложности и универсальности является перспективным и потенциально наиболее эффективным средством решения существующих проблем выбора механизмов обеспечения безопасности устройств персонального Интернета вещей в условиях ограниченности ресурсов производителя.

Ключевые слова:

безопасность IoT, персональный Интернет вещей, методика обеспечения безопасности, PIoT, PIoT устройства, защита умных устройств, интернет, защита устройств, портативные умные устройства, математическая модель

ВВЕДЕНИЕ

В современном мире одной из наиболее динамично развивающихся сфер является

Интернет вещей (IoT, Internet of Things) ^[1]. Большинство среднестатистических пользователей регулярно взаимодействует с портативными умными устройствами, начиная от фитнес-трекеров и заканчивая наушниками с беспроводной передачей данных. Подобные устройства относятся к классу персонального Интернета Вещей (PIoT, Personal Internet of Things) ^[2]. Количество брендов, под которыми они разрабатываются и распространяются, стремительно растет. Несмотря на имеющиеся преимущества ^[3], IoT- и PIoT-устройства сохраняют ряд уязвимостей, тем самым создавая возможности для проведения различных видов атак, что подчеркивает потребность в разработке эффективных методов обеспечения безопасности.

Предлагаемый в данной статье подход к выбору механизмов защиты устройств персонального Интернета вещей может быть полезен для проведения дальнейших исследований в области защиты подобного рода устройств, а также для пересмотра и модернизации существующих подходов к обеспечению защищенности данных, обрабатываемых в рамках систем PIoT.

Целью исследования является разработка методики обеспечения безопасности портативных устройств, в частности, и устройств Интернета вещей в целом.

ТЕКУЩЕЕ СОСТОЯНИЕ ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ СЕГМЕНТА УСТРОЙСТВ ПЕРСОНАЛЬНОГО ИНТЕРНЕТА ВЕЩЕЙ

Актуальность текущего исследования обусловлена стремительным ростом количества PIoT-устройств, что способствует увеличению числа потенциальных угроз утечек персональных данных пользователей и расширению множества методов и технологий, используемых для совершения атак на подобные классы и системы устройств ^[4]. В современных реалиях существует необходимость непрерывного обновления и усиления мер безопасности портативных умных устройств с целью защиты потребителей от уже имеющихся и вновь выявляемых угроз ^[5].

Исследование IoT Analytics, проведенное в 2023 году, продемонстрировало увеличение расходов компаний-производителей пользовательской электроники сегмента Интернета вещей на 21.5% по сравнению с предыдущим годом, что также сопровождалось и ростом инцидентов с участием PIoT-устройств ^[1]. Согласно прогнозам Statista, к 2030 году количество умных устройств превысит 29 миллиардов ^[2]. По данным исследования Omdia от февраля 2024 года число IoT-устройств с технологией eSIM уже превысило 1 миллиард, причем большая часть из них относится к сегменту персонального Интернета вещей. Ожидается, что к 2030 году их количество увеличится более чем в 3.5 раза, достигнув примерно 13% от общего числа устройств Интернета вещей ^[3]. Потребительский спрос на умные устройства ежегодно растет на 18% ^[4].

Исследования в области безопасности IoT и PIoT подтверждают необходимость разработки новых комплексных подходов к обеспечению безопасности и создания правовой базы для регулирования данного сегмента. Эксперты компании HP, проводившие исследование в направлении защищенности устройств Интернета вещей в 2014 году, пришли к выводам о том, что не существует полностью безопасных систем IoT, а сами устройства уязвимы для целевых атак ^[5]. В октябре 2017 года Еврокомиссия предложила обязательную сертификацию для устройств Интернета вещей, чтобы усложнить хакерам создание ботнетов на их основе ^[6]. Аналитики Kaspersky Digital Footprint Intelligence в 2023 году зафиксировали более 700 предложений в даркнете по

проведению DDoS-атак с использованием IoT-ботнетов, а также услуги по взлому PIIoT-устройств и продаже вредоносного программного обеспечения^[7].

В январе 2020 года правительство Великобритании опубликовало законопроект, направленный на защиту IoT-устройств^[8]. Министр цифровых технологий Мэтт Уормен отметил, что данный акт обязывает производителей учитывать действия злоумышленников для защиты конфиденциальности и безопасности пользователей. В 2022 году Великобритания стала первой страной, принявшей закон^[9] о безопасности потребительских IoT-устройств. Аналогичные законопроекты рассматриваются в России, Китае и США. В Евросоюзе комплексные работы по модернизации действующих законодательных актов, регулирующих IoT и PIIoT, запланированы на 2024 год^[10].

Обозначенные выше исследования и законопроекты свидетельствуют о повышенном интересе к проблематике IoT как со стороны экспертов в области информационной безопасности, так и с позиции государственных органов в контексте правового регулирования данной сферы^[6].

ОБЕСПЕЧЕНИЕ БЕЗОПАСНОСТИ УСТРОЙСТВ ПЕРСОНАЛЬНОГО ИНТЕРНЕТА ВЕЩЕЙ

В современных реалиях рынка основной упор в разработке потребительской электроники, в том числе и умных устройств, осуществляется коммерческими предприятиями, главной целью которых является финансовая выгода и приращение прибыли. В данных условиях немаловажной потребностью для компаний-производителей выступает оптимальное управление собственными ресурсами, что также необходимо учитывать при обеспечении безопасности IoT- и PIIoT-устройств.

Авторы статьи убеждены в том, что помимо очевидной потребности в правовом регулировании сегмента IoT посредством формирования грамотной законодательной базы, невозможно исключать и наиболее эффективное распределение существующих ресурсов организации на реализацию механизмов защиты производимых устройств. Подразумевается, что должно быть обеспечено выполнение требований законодательства при разработке перечня доступных и универсальных механизмов защиты с целью оптимизации выделяемых на их обеспечение и интеграцию временных, финансовых и интеллектуальных затрат.

Исходя из данного фактора, авторами были решены следующие задачи формируемого в рамках исследования подхода к обеспечению безопасности PIIoT-устройств:

- определены критерии оценки механизмов защиты;
- проведен анализ и сбор данных по разрабатываемому устройству;
- сформирована модель угроз и нарушителя;
- определены механизмы защиты;
- проведено ранжирование механизмов защиты в соответствии с введенными критериями;
- реализованы наиболее приоритетные механизмы защиты.

Указанные задачи включают в себя обеспечение безопасности со стороны действующего законодательства и оптимизацию процесса выбора механизмов защиты для PIIoT-устройств посредством разработки математической модели определения механизмов

защиты с критериями сложности реализации и универсальности механизмов защиты [7]. В рассматриваемом контексте использование этих двух критериев [8] обусловлено наиболее точным и широким охватом, а также определением оптимальных и эффективных механизмов защиты в том случае, когда можно утверждать о корректности и полноте заданных разработчиком параметров в ходе построения математической модели. Рассмотрим каждую задачу более подробно.

КРИТЕРИИ ДЛЯ ВЫБОРА МЕХАНИЗМОВ ЗАЩИТЫ

Основными критериями приоритета механизмов защиты можно определить:

- универсальность механизма защиты;
- сложность реализации механизма.

Универсальность механизма защиты отражает его способность одновременно удовлетворять требованиям бóльшего числа мер защиты. Сложность реализации каждого механизма оценивается в сравнении с ранее определёнными механизмами, поскольку этот показатель является исключительно субъективной величиной.

В целях дальнейшего ранжирования механизмов предлагается использовать диалоговый метод при подборе комбинации механизмов для обеспечения безопасности в рамках ограничения параметров защиты со стороны затрат и ресурсов организации-разработчика устройств персонального Интернета вещей.

АНАЛИЗ УСТРОЙСТВА И МОДЕЛИРОВАНИЕ УГРОЗ И НАРУШИТЕЛЯ

Следующей решаемой задачей обеспечения безопасности PИoT- и IoT-устройств является проведение предварительного анализа исследуемого устройства. В рамках этого анализа необходимо учитывать такие тактико-технические характеристики PИoT-устройств, как семейство микроконтроллеров, выполняющих роль основного управляющего элемента системы, вспомогательные модули и платы расширений, являющиеся наиболее уязвимыми компонентами IoT-устройств, а также программные компоненты и библиотеки, исследование которых может предоставить важную информацию о возможных уязвимостях программного обеспечения устройства.

Одними из наиболее распространенных семейств микроконтроллеров, выбираемых разработчиками IoT-решений, являются STM32 и ARM Cortex-M0. Обладание информацией об архитектуре управляющих элементов системы может предоставить сведения о потенциальных аппаратных уязвимостях и способах их эксплуатации. Например, в микроконтроллерах STM32 известны уязвимости, связанные с отладочными интерфейсами и механизмами защиты памяти [9].

Дополнительные модули устройства, такие как приёмо-передающие устройства Bluetooth, Wi-Fi, NFC и другие, могут служить точками входа для злоумышленников в систему. На текущий момент известны уязвимости технологии Bluetooth Low Energy, позволяющие осуществлять перехват данных с последующим несанкционированным доступом [10].

Программное обеспечение, версии прошивок и используемые библиотеки являются одними из наиболее информативных источников сведений об имеющихся недостатках механизмов обеспечения безопасности PИoT-устройства. Использование устаревших или уязвимых версий библиотек может привести к эксплуатации Heartbleed в OpenSSL и других известных уязвимостей [11].

Следующей задачей является формирование модели угроз и нарушителя. Опираясь на методический документ [\[11\]](#), модель угроз должна включать:

- Описание системы;
- Идентификацию потенциальных угроз;
- Классификацию нарушителя;
- Выявление потенциальных уязвимостей;
- Способы реализации угроз;
- Оценку последствий от нарушения свойств безопасности информации;
- Оценку последствий от нарушения штатного режима функционирования.

Большая часть информации, собранной на первом этапе реализации предлагаемого подхода, необходима для формирования модели угроз, что официально регламентируется регулирующим органом в сфере информационной безопасности, ФСТЭК [\[12\]](#), с целью дальнейшего описания угроз для каждого отдельного уровня системы.

На основе угроз и классификации злоумышленника, определенных в модели угроз и нарушителя, необходимо сформировать механизмы защиты. Стоит уточнить, что механизмы защиты не являются мерами защиты, существующими в рамках Приказа ФСТЭК России №21 от 18.02.2013 года [\[13\]](#). Меры защиты информации определяют тип и метод защиты, в то время как механизм защиты подразумевает конкретизацию в подходе к обеспечению безопасности.

Таким образом, на основе определенного мерой защиты метода обеспечения безопасности должен быть предложен механизм, способный в полном объеме выполнить ее требования в зависимости от функциональных возможностей и концепции разрабатываемого устройства Интернета вещей.

Предлагаемый авторами подход к обеспечению безопасности подразумевает определение механизмов защиты непосредственно экспертами организации, разрабатывающей устройства Интернета вещей. Это обусловлено тем, что невозможно предложить универсальные конкретизированные механизмы, так как их определение напрямую зависит от функциональных и иных особенностей производимого продукта, а также от сведений, содержащихся в модели угроз и нарушителя. При этом важно понимать, что данный процесс может иметь положительное влияние на универсальность защиты, так как один и тот же механизм может перекрывать одновременно более одной меры. Также открывается возможность для определения нескольких механизмов на реализацию одной меры с дальнейшим выбором наиболее эффективного из них посредством математической модели.

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ОПРЕДЕЛЕНИЯ МЕХАНИЗМОВ ЗАЩИТЫ

Следующей задачей является разработка математической модели оценки механизмов защиты, основанной на системе из двух критериев, что снижает фактор субъективности при принятии решений и обеспечивает рациональное распределение ресурсов производителя. Модель учитывает эффективность механизмов защиты в противодействии угрозам и сложность их реализации.

Предлагается ввести следующие множества:

1. $\overline{A} = \{a_1, a_2, \dots, a_n, \dots, a_N\}$ – множество механизмов защиты, которые могут быть сопоставлены с мерами защиты из модели угроз для обеспечения безопасности устройства. Элементы этого множества необходимы для последующего определения сложности реализации механизмов защиты и оценки вероятности предотвращения потенциальной атаки на устройство;
2. $\overline{U} = \{u_1, u_2, \dots, u_m, \dots, u_M\}$ – множество угроз безопасности, обозначенных в модели угроз. На основе элементов этого множества производится оценка ущерба для пользователя устройства;
3. $\overline{R} = \{r_1, r_2, \dots, r_l, \dots, r_L\}$ – множество ресурсов разработчика, которое необходимо для определения оценки возможностей для обеспечения безопасности устройства;
4. $\overline{tr} = \{tr_1, tr_2, \dots, tr_q, \dots, tr_Q\}$ – множество факторов, представляющих из себя необходимость реализации определенных механизмов защиты.

Для множеств вводятся следующие параметры:

1. $\overline{w_m \geq 0, m = 1, \dots, M}$ – оценка ущерба для пользователя устройства IoT в случае успешной реализации m -ой угрозы.
2. $\overline{p_j^{(a)} \forall j = 1, \dots, M}$ – вероятность (или возможность) появления j -ой атаки (реализация угрозы) на устройство IoT.
3. $\overline{p_{nj} \in [0, 1], \forall n = 1, \dots, N, j = 1, \dots, M}$ – вероятность (или возможность с точки зрения нечетких множеств) предотвращения j -ой атаки (реализации угрозы) при использовании n -ого механизма.
4. $\overline{c_n \geq 0, \forall n = 1, \dots, N}$ – числовая оценка сложности реализации n -ого механизма.
5. $\overline{v_{ln} \geq 0, \forall l = 1, \dots, L, n = 1, \dots, N}$ – числовая оценка возможностей для реализации n -ого механизма.
6. $\overline{V_l \geq 0, \forall l = 1, \dots, L}$ – максимальный «объем» l -ых ресурсов (финансовых, технических, кадровых, профессиональных, временных, интеллектуальных), который разработчик готов выделить на обеспечение безопасности устройства IoT.
7. $\overline{B_{\leq n \times q} = \|b_{nk}\|, n = 1, \dots, N, k = 1, \dots, Q}$ – булева матрица, задающая факторы необходимости реализации определенного механизма для выполнения меры защиты из модели угроз и нарушителя: $\overline{b_{nk} = 1}$, если n -ый механизм обеспечивает выполнение k -ого фактора необходимости реализации, $\overline{b_{nk} = 0}$ – в противном случае.

Сложность реализации каждого механизма защиты $\overline{c_n}$ оценивается на основе экспертной оценки команды разработчиков. Оценка возможностей для реализации механизмов защиты $\overline{v_{ln}}$ проводится по каждому ресурсу $\overline{r_l}$ из множества ресурсов разработчика \overline{R} . Максимальный объём ресурсов $\overline{V_l}$ устанавливается разработчиком и отражает предельное количество каждого ресурса $\overline{r_l}$, которое организация готова выделить на обеспечение безопасности устройства.

Введем логическую переменную $\overline{x_n \in \{0, 1\}, \forall n \in N}$, такую, что $\overline{x_n = 1}$, если n -ый механизм используется для защиты, $\overline{x_n = 0}$ – в противном случае. В результате образуется вектор \overline{x} .

Предлагается ввести следующие условия, задаваемые для введенных критериев:

1. Оценка универсальности – это измерение способности механизма защиты одновременно удовлетворять множеству мер защиты:

$$W(\mathbf{X}) = \sum_{j \in M} w_j p_j^{(a)} \max_{n \in N} \{p_{nj} x_n\} \quad (1)$$

Данный критерий необходимо максимизировать.

2. Сложность реализации используемых механизмов защиты:

$$C(\mathbf{X}) = \sum_{n \in N} c_n x_n \quad (2)$$

Данный критерий необходимо минимизировать.

При этом вводятся ограничения:

- на использование ресурсов:

$$\sum_{n \in N} v_{ln} x_n \leq V_l, \forall l = 1, \dots, L; \quad (3)$$

- на факторы необходимости реализации механизма (выполнение меры защиты из модели угроз и нарушителя):

$$\sum_{n \in N} b_{nk} x_n \geq 1, \forall k = 1, \dots, Q. \quad (4)$$

Ограничение на использование ресурсов (3) существует по той причине, что v_{ln} представляет собой оценку объема ресурса l , необходимого для реализации механизма a_n , а V_l – максимальное количество данного ресурса, которое доступно разработчику. Этим условием обеспечивается то, что совокупные затраты на выбранные механизмы защиты не будут превышать доступные ресурсы.

Ограничение на выполнение факторов необходимости реализации механизма (4) подразумевает, что для каждого фактора из множества Q , представляющего собой требование к механизму защиты, должен быть задействован хотя бы один механизм a_n , обеспечивающий выполнение этого фактора $b_{nk} = 1$. Это условие необходимо для гарантии того, что все меры защиты, определённые в модели угроз и нарушителя, будут выполнены.

Таким образом, решается задача булевого программирования с двумя показателями качества, где первый показатель является нелинейным, второй определен как линейный, а ограничения линейные. Сформируем систему критериев для дальнейшей оптимизации:

$$\begin{cases} W(\mathbf{X}) = \sum_{j \in M} w_j p_j^{(a)} \max_{n \in N} \{p_{nj} x_n\} \rightarrow \max \\ C(\mathbf{X}) = \sum_{n \in N} c_n x_n \rightarrow \min \end{cases} \quad (5)$$

Полученная система представляется решением процесса ранжирования механизмов защиты посредством их приоритизации.

РАСЧЕТ ПОКАЗАТЕЛЕЙ ПРИОРИТЕТА МЕХАНИЗМОВ ЗАЩИТЫ

Для дальнейшего ранжирования механизмов защиты с учетом сформированной системы критериев (5) необходимо определить их «вес». С целью расчета этого параметра предлагается нормировать веса критериев так, чтобы их сумма равнялась единице. Это

необходимо для обеспечения равномерного и сопоставимого влияния каждого критерия выбор механизмов защиты.

Первым шагом в определении численных значений критериев является фиксация количества мер защиты из модели угроз и нарушителя, выполняемых каждым из предложенных механизмов. Примеры сопоставления мер из модели угроз с механизмами защиты показаны в табл. 1. Обозначения, используемые для идентификации мер защиты, соответствуют требованиям, определённым в Приказе ФСТЭК РФ от 18.02.2013 № 21^[14]:

- УПД.6: Ограничение числа неудачных попыток доступа в информационную систему. Мера направлена на предотвращение перебора паролей и несанкционированного доступа;
- УПД.8: Оповещение пользователя при успешном входе о предыдущем доступе в информационную систему. Мера направлена на информирование пользователя о фактах входа в систему;
- АУД.4: Регистрация событий безопасности. Мера предусматривает фиксацию событий, связанных с безопасностью, для последующего реагирования и анализа;
- АУД.7: Мониторинг безопасности. Мера включает в себя постоянное отслеживание состояния безопасности информационной системы для своевременного обнаружения и предотвращения инцидентов.

Рассматриваемые меры используются для оценки универсальности механизмов защиты – их способности одновременно удовлетворять требованиям нескольких мер защиты.

Таблица 1. Количество выполняемых мер защиты отдельными механизмами

Механизм защиты	Реализуемая мера защиты из модели угроз и нарушителя	Количество выполняемых мер защиты посредством реализации механизма
Введение блокировки учетной записи после нескольких неудачных попыток входа с дальнейшим оповещением пользователя	УПД.6	1
Уведомление по электронной почте или SMS	УПД.8	1
Фиксирование данных о попытках входа с различных IP-адресов за короткий промежуток времени	АУД.4, АУД.7	2

После определения количественного показателя универсальности необходимо оценить сложность реализации каждого механизма защиты. Данный параметр является исключительно субъективным и должен предусматривать индивидуальные возможности команды разработчиков и имеющиеся ресурсы. При определении оценки рекомендуется использование 10-балльной шкалы для наибольшей наглядности показателя, что

позволяет достаточно точно оценить сложность в контексте индивидуальных особенностей организации. Максимальный балл присваивается в наивысшей степени комплексному и объемному с точки зрения реализации механизму защиты в формируемом перечне. Пример оценок сложности механизмов защиты представлен в табл. 2.

Таблица 2. Сопоставление оценки сложности с механизмом по 10 балльной шкале

Механизм защиты	Оценка сложности реализации механизма защиты по 10 балльной шкале
Введение блокировки учетной записи после нескольких неуспешных попыток входа с дальнейшим оповещением пользователя	4
Уведомление по электронной почте или SMS	4
Фиксирование данных о попытках входа с различных IP-адресов за короткий промежуток времени	3

РАНЖИРОВАНИЕ МЕХАНИЗМОВ ЗАЩИТЫ

Завершающим этапом является ранжирование механизмов защиты в соответствии с ранее определёнными параметрами критериев приоритетности с помощью сформированной математической модели. Для этого производится нормировка «весов» показателей по сложности реализации и универсальности механизмов защиты: суммируются все веса, после чего каждое значение критерия делится на полученную сумму. Пример формирования значений критериев представлен в табл. 3.

Таблица 3. Значения критериев механизмов защиты

№	Цель	Критерии	
		Реализация мер из модели угроз	Сложность реализации
	Вес критериев	0.5	0.5
	Механизмы защиты	Оценка универсальности	Оценка сложности
1	Введение блокировки учетной записи после нескольких неуспешных попыток входа с дальнейшим оповещением пользователя	0.25	0.36
2	Уведомление по электронной почте или SMS	0.25	0.36
3	Фиксирование данных о попытках входа с различных IP-адресов за короткий промежуток времени	0.50	0.28

Значения вычислены таким образом, чтобы в сумме вес всех определенных в модели критериев, а также общая оценка доступных механизмов защиты по каждому отдельному

критерию по модулю были равны единице. В данном случае рассматриваются два критерия с одинаковыми весами 0.5.

После проведенных расчетов можно определить приоритет каждого механизма защиты для их последующего ранжирования по следующей формуле:

$$F_n = \alpha \left(w_j p_j^{(a)} \max_{n \in N} \{ p_{nj} x_n \} \right) - \beta (c_n x_n).$$

(6)

где α – вес критерия универсальности механизма защиты, β – вес критерия сложности реализации механизма защиты, $w_j p_j^{(a)} \max_{n \in N} \{ p_{nj} x_n \}$ – значение оценки универсальности механизма, $c_n x_n$ – значение оценки сложности реализации механизма.

Расчет приоритетов для механизмов защиты в рассматриваемом примере дает следующие значения:

$$F_1 = 0.5 \times 0.25 - 0.5 \times (0.36) = -0.055,$$

$$F_2 = 0.5 \times 0.25 - 0.5 \times (0.36) = -0.055,$$

$$F_3 = 0.5 \times 0.5 - 0.5 \times (0.28) = 0.11.$$

Таблица иерархии механизмов на основе их приоритета для приведенного в статье примера выглядит следующим образом (табл. 4).

Таблица 4. Перечень механизмов защиты в соответствии с показателями приоритета

Механизм защиты	Показатель приоритета
Фиксирование данных о попытках входа с различных IP-адресов за короткий промежуток времени	0.11
Введение блокировки учетной записи после нескольких неуспешных попыток входа с дальнейшим оповещением пользователя	-0.055
Уведомление по электронной почте или SMS	-0.055

Данные, полученные в результате такого расчета, являются уникальными для каждого продукта и организации, так как напрямую зависят от модели угроз и нарушителя, а также от объема имеющихся в компании-разработчике ресурсов.

РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЙ И ИХ ОБСУЖДЕНИЕ

В результате проведенного исследования разработан подход к обеспечению безопасности устройств персонального Интернета вещей на основе математической модели с двумя критериями: универсальность механизма защиты и сложность реализации. Применение данного подхода позволяет определить и ранжировать механизмы защиты, оптимизируя их выбор с учётом ограничений ресурсов и специфики разрабатываемого устройства.

Анализ результатов показал, что предложенный подход достаточно эффективно решает задачу оптимизации выбора механизмов защиты. Использование математической модели дает возможность оценить каждый механизм по заданным критериям, снижая фактор

субъективности в процессе принятия решений [12]. Ранжирование механизмов защиты обеспечивает взвешенное распределение ресурсов, что особенно актуально для коммерческих организаций.

Однако следует учитывать, что точность и эффективность модели зависят от корректности исходных данных и адекватности выбранных критериев. В перспективах дальнейшего исследования целесообразно расширить набор критериев и разработать стандартизированные методы оценки, что позволит повысить точность модели и адаптировать её к различным типам устройств и условиям эксплуатации. Особое внимание следует уделить внедрению механизмов защиты, основанных на алгоритмах машинного обучения. В условиях растущей сложности сетевых угроз и увеличения объёма данных, обрабатываемых в системах PIoT, существует необходимость разработки адаптивных и гибких методов детектирования аномального поведения при проведении различных атак [13]. Также, учитывая, что протокол Bluetooth Low Energy (BLE) является одним из наиболее популярных для портативных устройств [14], важно детально проработать перечень механизмов защиты именно для этой беспроводной технологии [15].

Предлагаемый подход к выбору механизмов защиты устройств персонального Интернета вещей может способствовать повышению защищённости пользовательских умных устройств, обеспечивая рациональное распределение ресурсов и минимизируя риски компрометации информационных систем.

БЛАГОДАРНОСТИ / GRATITUDE

Авторы выражают искреннюю благодарность канд. техн. наук, А.В. Королькову за критический подход и оперативную вовлеченность в процесс редактирования полученных результатов исследования / The authors express their sincere gratitude to Ph.D. in Technical Sciences A.V. Korolkov for his critical approach and prompt involvement in the process of editing the research results.

[1] State of IoT – Spring 2023. <https://iot-analytics.com/product/state-of-iot-spring-2023/>

[2] Number of Internet of Things (IoT) connected devices worldwide from 2019 to 2023, with forecasts from 2022 to 2030. <https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/>

[3] Omdia: New Omdia research shows eSIM installed base in IoT to top 3.6 billion by 2030. <https://omdia.tech.informa.com/pr/2024/feb/new-omdia-research-shows-esim-installed-base-in-iot-to-top-3-point-6-billion-by-2030>

[4] Интернет вещей, IoT, M2M мировой рынок. [https://www.tadviser.ru/index.php/Статья:Интернет_вещей,_IoT,_M2M_\(мировой_рынок\)](https://www.tadviser.ru/index.php/Статья:Интернет_вещей,_IoT,_M2M_(мировой_рынок))

[5] HP Discovers Common Vulnerabilities in 10 IoT Devices. <https://www.eweek.com/security/hp-discovers-common-vulnerabilities-in-10-iot-devices/>

[6] Improving Internet of Things Device Certification with Policy-based Management. <https://publications.jrc.ec.europa.eu/repository/handle/JRC106530>

[7] DDos, программы-вымогатели, майнеры: «Лаборатория Касперского» проанализировала ландшафт киберугроз для интернета вещей.

https://www.kaspersky.ru/about/press-releases/2023_ddos-programmy-vymogateli-majnery-laboratoriya-kasperskogo-proanalizirovala-landshaft-kiberugroz-dlya-interneta-veshej

[8] Government to strengthen security of Internet-connected products. <https://www.gov.uk/government/news/government-to-strengthen-security-of-internet-connected-products>

[9] Product Security and Telecommunications Infrastructure Act 2022. <https://www.legislation.gov.uk/ukpga/2022/46/part/1/enacted>

[10] IoT Cybersecurity: regulating the Internet of Things. <https://www.thalesgroup.com/en/markets/digital-identity-and-security/iot/inspired/iot-regulations>

[11] «Методический документ. Методика оценки угроз безопасности информации» (утв. ФСТЭК России 05.02.2021) <https://fstec.ru/dokumenty/vse-dokumenty/spetsialnye-normativnye-dokumenty/metodicheskij-dokument-ot-5-fevralya-2021-g>

[12] Приказ ФСТЭК России от 18.02.2013 №21 «Об утверждении Составы и содержания организационных и технических мер по обеспечению безопасности персональных данных при их обработке в информационных системах персональных данных». <https://fstec.ru/dokumenty/vse-dokumenty/spetsialnye-normativnye-dokumenty/metodicheskij-dokument-ot-5-fevralya-2021-g>

[13] Приказ ФСТЭК России от 18.02.2013 №21 «Об утверждении Составы и содержания организационных и технических мер по обеспечению безопасности персональных данных при их обработке в информационных системах персональных данных». <https://fstec.ru/dokumenty/vse-dokumenty/spetsialnye-normativnye-dokumenty/metodicheskij-dokument-ot-5-fevralya-2021-g>

[14] Приказ ФСТЭК России от 18.02.2013 №21 «Об утверждении Составы и содержания организационных и технических мер по обеспечению безопасности персональных данных при их обработке в информационных системах персональных данных». <https://fstec.ru/dokumenty/vse-dokumenty/spetsialnye-normativnye-dokumenty/metodicheskij-dokument-ot-5-fevralya-2021-g>

Библиография

1. Львович И.Я., Преображенский А.П., Преображенский Ю.П., Чопоров О.Н., Проблемы использования технологии интернет вещей. *Вестник Воронежского института высоких технологий*. 2019;13(1):73-75.
2. Biswa Mohan Sahoo, Mohanty SP, Deepak Puthal, Pillai P. Personal Internet of Things (PIoT): What Is It Exactly? *Cyber Security for Next-Generation Computing Technologies*. 2021 Nov 1;10(6):58-60. DOI:10.1201/9781003404361-14
3. Fariha Eusufzai, Aldrin Nippon Bobby, Farzana Shabnam, Saifur Rahman Sabuj. Personal internet of things networks: An overview of 3GPP architecture, applications, key technologies, and future trends. *International journal of intelligent networks*. 2024 Feb 1; 5(6):77-91; DOI:10.1016/j.ijin.2024.02.001
4. Информационная безопасность в системе "Интернет вещей" / А.Г. Коробейников, А.Ю. Гришенцев, Д.И. Дикий [и др.]. *Вестник Чувашского университета*. 2018. № 1. С. 117-128.
5. Dean A, Agyeman M.O. A Study of the Advances in IoT Security. *Proceedings of the 2nd International Symposium on Computer Science and Intelligent Control - ISCSIC '18*. 2018;1-

5; DOI:10.1145/3284557.3284560

6. Каженова Ж.С. Безопасность в протоколах и технологиях IoT: обзор / Ж.С. Каженова, Ж.Е. Кенжебаева. *International Journal of Open Information Technologies*. 2022. № 3. С. 10-15. – ISSN 2307-8162

7. Керимов Вагиф Асад Оглы. Алгоритм принятия решения для одной многокритериальной задачи с матричной моделью / Вагиф Асад Оглы Керимов, Фаик Гасан Оглы Гаджиев. *Universum: технические науки*. 2023. № 2. С. 62-65.

8. Юрлов Ф.Ф. Методика комплексного применения набора принципов оптимальности при выборе эффективных решений при наличии неопределенности внешней среды и многокритериальности / Ф.Ф. Юрлов, С.Н. Яшин, А.Ф. Плеханова. *Вестник Нижегородского университета им. Н. И. Лобачевского. Серия: Социальные науки*. 2022. № 1. С. 49-55.

9. Басс А.В. Особенности работы с микроконтроллером stm32. *Известия Тульского государственного университета. Технические науки*. 2019. № 1. С. 35-40.

10. Саенко М.А. Анализ уязвимостей беспроводных каналов передачи информации / М.А. Саенко, Д.А. Мельников, М.А. Данилов. *Образовательные ресурсы и технологии*. 2023. № 1. С. 82-90.

11. Chintchik N.V. Vulnerabilities detection via static taint analysis / N.V. Chintchik, V.N. Ignatiev. *Труды Института системного программирования РАН*. 2019. Т. 31, № 3. С. 177-189.

12. Абдусаломова Н.М. Математическое моделирование научных знаний как отдельная позиция между теорией и экспериментом. *Мировая наука*. 2024. № 6. С. 44-47.

13. Истратова Е.Е. Применение нейронных сетей для обнаружения аномального трафика в сетях Интернета вещей. *International Journal of Open Information Technologies*. 2024. № 1. С. 65-70.

14. K. E. Jeon, J. She, P. Soonsawad and P. C. Ng. BLE Beacons for Internet of Things Applications: Survey, Challenges, and Opportunities. *IEEE Internet of Things Journal*. V. 5, № 2, P. 811-828, April 2018, DOI: 10.1109/JIOT.2017.2788449.

15. Prathibha Muraleedhara, Christo S, Jaya J, D. Yuvasini. Any Bluetooth Device Can be Hacked. Know How? *Cyber Security and Applications*. 2024 Feb 1;100041-1. DOI:10.1016/j.csa.2024.100041

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Представленная статья на тему «Подход к выбору механизмов защиты устройств персонального Интернета вещей на основе математической модели с двумя критериями» соответствует тематике журнала «Программные системы и вычислительные методы» и посвящена актуальной проблеме – разработке эффективных методов обеспечения безопасности IoT- и PIoT -устройств. По мнению авторов это связано с тем, что большинство среднестатистических пользователей регулярно взаимодействует с портативными умными устройствами, начиная от фитнес-трекеров и заканчивая наушниками с беспроводной передачей данных. Подобные устройства относятся к классу персонального Интернета Вещей (PIoT, Personal Internet of Things). Количество брендов, под которыми они разрабатываются и распространяются, стремительно растет. Несмотря на имеющиеся преимущества, IoT- и PIoT- устройства сохраняют ряд уязвимостей, тем самым создавая возможности для проведения различных видов атак. В статье представлен достаточно широкий анализ литературных российских и

зарубежных и интернет-источников по теме исследования. Указана теоретико-методологическая основа исследования.

Авторами самостоятельно проведен комплексный анализ RIoT-устройств с учетом таких тактико-технических характеристик устройств, как семейство микроконтроллеров, выполняющих роль основного управляющего элемента системы, вспомогательные модули и платы расширений, являющиеся наиболее уязвимыми компонентами IoT-устройств, а также программные компоненты и библиотеки, исследование которых может предоставить важную информацию о возможных уязвимостях программного обеспечения устройства. Также авторами проведен расчет показателей приоритета механизмов защиты.

Стиль и язык изложения материала является достаточно доступным для широкого круга читателей. Практическая значимость статьи четко обоснована. Статья по объему соответствует рекомендуемому объему от 12 000 знаков.

Статья достаточно структурирована - в наличии введение, заключение, внутреннее членение основной части (анализ литературы, методология, результаты исследования и обсуждение).

Авторами проведено исследование, в результате которого разработан подход к обеспечению безопасности устройств персонального Интернета вещей на основе математической модели с двумя критериями: универсальность механизма защиты и сложность реализации. Применение данного подхода позволяет определить и ранжировать механизмы защиты, оптимизируя их выбор с учётом ограничений ресурсов и специфики разрабатываемого устройства.

К недостаткам можно отнести следующие моменты: из содержания статьи не прослеживается научная новизна. Отсутствует четкое выделение предмета, объекта исследования.

Рекомендуется четко обозначить научную новизну исследования, сформулировать предмет, объект. Также будет целесообразным добавить о перспективах дальнейшего исследования.

Статья «Подход к выбору механизмов защиты устройств персонального Интернета вещей на основе математической модели с двумя критериями» требует доработки по указанным выше замечаниям. После внесения поправок рекомендуется к повторному рассмотрению редакцией рецензируемого научного журнала.

Результаты процедуры повторного рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Статья посвящена вопросам обеспечения безопасности устройств персонального Интернета вещей (PIoT), которые активно внедряются в повседневную жизнь. Исследование акцентируется на разработке методологии выбора механизмов защиты с использованием математической модели, что позволяет учесть ограниченность ресурсов и специфические угрозы. Авторы также рассматривают практические аспекты реализации предложенного подхода для достижения оптимальной защищенности устройств.

Исследование опирается на использование математической модели, включающей два ключевых критерия: универсальность механизма защиты и сложность его реализации.

Представленный подход базируется на использовании булевого программирования, что способствует минимизации субъективности при выборе защитных решений. Авторы подробно описывают этапы реализации модели, включая формирование множества угроз, определение механизмов защиты, построение модели угроз и последующее ранжирование механизмов по степени приоритета. Методология дополнена примерами расчетов, что подтверждает её применимость и достоверность.

Актуальность исследования обусловлена стремительным ростом числа РIoT-устройств, что увеличивает риски утечек данных и уязвимостей. Представленные статистические данные, включая прогнозы роста числа устройств до 29 миллиардов к 2030 году, подтверждают важность создания эффективных решений для защиты данных. Учитывая растущее число угроз и отсутствие универсальных стандартов защиты, статья предлагает своевременный и практически значимый подход.

Предложение метода оценки механизмов защиты с использованием математической модели является ключевым элементом научной новизны работы. Модель базируется на учете ограниченности ресурсов разработчиков и интеграции двух критериев, что делает её уникальной в контексте РIoT. Авторы также подчеркивают, что их подход позволяет учитывать индивидуальные особенности каждого устройства, что является значимым шагом в создании адаптивных систем защиты.

Статья написана академическим стилем с высоким уровнем технической проработки.

Структура работы:

- Введение четко формулирует цели исследования и подчеркивает актуальность проблемы.
- Анализ текущего состояния дает всесторонний обзор существующих подходов и их ограничений, подкрепленный статистическими данными.
- Методология подробно описывает этапы разработки математической модели, включая формирование критериев, оценку ресурсов и анализ угроз.
- Результаты и обсуждение акцентируют внимание на применении модели для практических задач.
- Выводы содержат рекомендации и намечают пути дальнейшего исследования.

Текст сопровождается таблицами и диаграммами, что упрощает восприятие материала. Однако было бы полезно включить иллюстрацию полной схемы процесса выбора механизмов защиты.

Список литературы разнообразен и включает актуальные работы, посвященные вопросам безопасности IoT, статистическим исследованиям и нормативной базе. Указанные источники подтверждают глубину проведенного анализа. Тем не менее, добавление большего числа примеров практического применения предложенного подхода могло бы усилить значимость работы.

Авторы аргументированно обосновывают свой подход, признавая, что универсальные механизмы защиты для РIoT-устройств создать сложно. Однако предложенная модель позволяет адаптироваться к индивидуальным условиям каждого устройства, что снижает вероятность критики со стороны оппонентов. При этом они подчеркивают необходимость дальнейшей стандартизации критериев, что открывает новые горизонты для дискуссии и совместных исследований.

В заключении авторы четко обозначают практическую ценность своей работы, подчеркивая, что предложенный подход может быть использован как производителями РIoT-устройств, так и исследователями в области информационной безопасности. Статья вызывает интерес как у научного сообщества, так и у представителей отрасли, предоставляя инструменты для повышения уровня защиты умных устройств.

Статья представляет собой высококачественное исследование, основанное на инновационной методологии и актуальных данных. Работа отличается высокой

практической значимостью и научной новизной, что делает её достойной публикации.

Рекомендация: принять статью к публикации. Более того, в случае её доработки (например, добавления дополнительных иллюстраций процесса или расширения обзора практического применения), она может быть рекомендована к включению в список лучших публикаций месяца.

Программные системы и вычислительные методы

Правильная ссылка на статью:

Хлесткин А.Ю., Райков А.В., Казанцев А.А., Емелин Д.П., Ларин Д.В. Роль операционных систем и оболочек в облачных вычислениях: анализ ОС и оболочек, используемых в облачных платформах и их влияние на облачную инфраструктуру // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.70626 EDN: KYNBQH URL: https://nbpublish.com/library_read_article.php?id=70626

Роль операционных систем и оболочек в облачных вычислениях: анализ ОС и оболочек, используемых в облачных платформах и их влияние на облачную инфраструктуру

Хлесткин Андрей Юрьевич

кандидат технических наук

доцент, кафедра информатика и вычислительная техника; Поволжский государственный университет телекоммуникаций и информатики

443010, Россия, Самарская область, г. Самара, ул. Льва Толстого, 23

✉ a.hlestkin@psuti.ru



Райков Александр Вячеславович

ORCID: 0009-0005-0033-8524

студент, кафедра информатики и вычислительной техники, Поволжский государственный университет телекоммуникаций и информатики

443010, Россия, Самарская область, г. Самара, ул. Льва Толстого, 23

✉ sraikov7@mail.ru



Казанцев Андрей Алексеевич

студент, кафедра информатики и вычислительной техники, Поволжский государственный университет телекоммуникаций и информатики

443010, Россия, Самарская область, г. Самара, ул. Льва Толстого, 23

✉ NuclearAndGoner@gmail.com



Емелин Даниил Павлович

студент, кафедра информатики и вычислительной техники, Поволжский государственный университет телекоммуникаций и информатики

443010, Россия, Самарская область, г. Самара, ул. Льва Толстого, 23

✉ demelin163@gmail.com



Ларин Денис Вячеславович

студент, кафедра информатики и вычислительной техники, Поволжский государственный университет телекоммуникаций и информатики

443010, Россия, Самарская область, г. Самара, ул. Льва Толстого, 23

✉ denlar1989@gmail.com



[Статья из рубрики "Models and methods of information security management"](#)

DOI:

10.7256/2454-0714.2024.4.70626

EDN:

KYNBQH

Дата направления статьи в редакцию:

01-05-2024

Аннотация: Основное внимание в статье уделяется операционным системам, облачным вычислениям и командным оболочкам, которые активно развиваются несколько десятков лет и уже являются частью жизни, как обычного пользователя, так и профессионала компьютерных технологий. Эти объекты рассматриваются как отдельные составляющие информационных технологий, так и их взаимосвязь и результаты этой взаимосвязи. Операционные системы в облачных серверах выполняют управленческую роль. Если быть точнее, то они управляют ресурсами физических серверов. Операционные системы или же ОС в данном случае определяют несколько параметров. К таким параметрам мы можем отнести то, как операционные системы могут использовать память, хранилище для различных виртуальных машин и управлять ими. Командные оболочки в свою очередь представляются приложением, которое предоставляет пользователю некий интерфейс командной строки, в которой тот вводит команды как по отдельности, так и запускает скрипты, состоящие из списка команд. Методы исследования включают теоретические (классификация, сравнительный анализ, анализ литературы) и практические (эксперимент, моделирование) подходы. Это позволяет провести комплексный анализ функционирования операционных систем и командных оболочек в облачных вычислениях. Научной новизной нашего исследования служит приведение скриптов для выполнения той или иной задачи в области облачных вычислений на определённой операционной системе с использованием вышеописанных командных оболочек. Таким образом, авторами были приведены теоретические данные об операционных системах и командных оболочках. Авторы привели примеры скриптов и анализ безопасности для командных оболочек Bash и Bourne Shell (sh) для операционной системы Linux и скрипты для командных оболочек Command Prompt (cmd.exe) и Windows PowerShell для операционной системы Microsoft Windows. В результате проведенной работы были составлены таблицы с влиянием операционной системы и командной оболочки на выполнение облачных вычислений. Анализ таблиц позволил охарактеризовать авторам объекты исследования данной научной работы и сделать соответствующие выводы.

Ключевые слова:

операционные системы, командные оболочки, облачные вычисления, облачные платформы, облачная инфраструктура, Linux, Microsoft Windows, Bash, Bourne Shell, Hyperfine

Введение

Операционные системы и командные оболочки по своей природе являются основополагающими элементами облачной инфраструктуры. Эти элементы обеспечивают стабильность, безопасность, эффективное использование ресурсов и удобное управление облачной инфраструктурой. Немаловажным является и правильный выбор, и настройка операционной системы и оболочки в успешности вычислений.

Облачными вычислениями можно назвать процесс предоставления удаленных вычислительных мощностей конечным потребителем. Данные мощности могут использоваться различным образом – начиная от использования рабочих мест с помощью виртуальных машин и других технологий виртуализации, и заканчивая майнингом криптовалют [\[1\]](#).

«Cloud computing» или же «Облачные вычисления» являются реализацией определённого метафорического образа Internet в виде облака, благодаря которому пользователи получают доступ ко всем возможным сервисам и ресурсам. Основы вычислений заключается в использовании виртуализированных или масштабируемых ресурсов.

В данной статье осуществлена попытка проведения структурного анализа операционных систем и оболочек и их влияние на облачную инфраструктуру.

Практическая часть

В современном мире существует множество специализированных операционных систем для запуска контейнеров. Авторами статьи в качестве примера такой операционной системы хотелось бы отметить CoreOS Container Linux. Данная операционная система автоматически обновляется, а также обеспечивает высокую доступность в использовании контейнеризованных приложений.

Выбор данной операционной системы был сделан из-за её особенностей. Так, главным преимуществом CoreOS Container Linux является концепт immutable infrastructure (неизменяемая инфраструктура). То есть операционная система и приложения запускаются в контейнерах и не подвергаются изменениям во время выполнения. Любые изменения в системе осуществляются путём замены всего образа операционной системы на новую версию.

CoreOS Container Linux предоставляет некую систему управления обновлениями, которая в свою очередь обновляет операционную систему без перезагрузки. Если сравнивать CoreOS Container Linux с операционной системой Microsoft Windows, то у второй в большинстве случаев после установки обновления требуется дополнительное действие - перезагрузка. Отсутствие перезагрузки после обновления у CoreOS Container Linux обеспечивает её безопасностью и надёжностью обновлений в кластерной среде без простоев.

В CoreOS Container Linux отсутствует избыточность, как у многих дистрибутивов Linux. CoreOS Container Linux предоставляет минимальный набор компонентов, в следствии чего идёт упрощение конфигурации, а так же обеспечение надёжной работой в распределённых средах.

Container-Optimized OS — это образ операционной системы для виртуальной машины Google Compute Engine, оптимизированный для запуска контейнеров Docker, предоставляемые Google Cloud Platform (GCP) (набор облачных служб, которая

представляет компания Google).

Container-Optimized OS обладает следующими функциями:

- Запуск контейнеров «из коробки»: экземпляры операционной системы, оптимизированные для контейнеров, поставляются с предустановленными средами выполнения Docker и containerd и cloud-init. С помощью экземпляра операционной системы, оптимизированного для контейнеров, существует возможность запускать контейнер одновременно с созданием виртуальной машины, не требуя настройки со стороны хоста.
- Меньшая поверхность атаки: ОС, оптимизированная для использования в контейнерах, занимает меньше места, что снижает потенциальную поверхность атаки для созданного экземпляра.
- Заблокировано по умолчанию: экземпляры операционной системы, оптимизированные для контейнеров, по умолчанию включают заблокированный брандмауэр и другие параметры безопасности.
- Автоматические обновления: экземпляры операционной системы, оптимизированные для контейнеров, настроены на автоматическую загрузку еженедельных обновлений в фоновом режиме; для использования последних обновлений требуется только перезагрузка.

Пользователь может выбрать данную операционную систему, если ему необходимо решать следующие задачи:

- Требуется поддержка контейнеров или Kubernetes с минимальной настройкой.
- Нужна операционная система, занимающая мало места и защищенная для контейнеров.
- Нужна операционная система, которая протестирована для запуска Kubernetes на экземплярах Google Compute Engine.

Авторами было решено сделать акцент на операционных системах Linux, Microsoft Windows и на оболочках Bourne Again Shell, Bourne Shell (sh), Command Prompt (cmd.exe), Windows PowerShell.

Финский студент (заметьте, студент) Линус Торвальд в 1991 году создал клон Unix и назвал его **Linux** (версия 0.0.1). Вся ОС размещалась в ядре, была монолитной (9300 строк кода на C, 950 - на Assembler). Работала на процессоре Intel (386). Окончательно датой выпуска считается 1994 год, версия 1.0 (165 тысяч строк кода, новая файловая система, сетевое программное обеспечение). 2-я версия - 1996 год (470 тысяч строк кода, добавлены новые драйверы устройств) [\[2\]](#).

Linux входит в число крупнейших хорошо документированных линеек программных продуктов, изучаемых до сих пор. Сообщество разработчиков Linux включает как добровольцев, так и платных разработчиков из более чем 200 компаний, включая Red Hat, IBM, Intel, Oracle, Google и Microsoft среди прочих [\[4\]](#). Зрелость проекта проявляется по нескольким показателям, таким как размер кодовой базы (более 8 миллионов строк), количество активных разработчиков (600–1200 на каждый релиз и далее) и уровень активности (до 10000 исправлений на выпуск) [\[5\]](#).

Если говорить про достоинства данной операционной системы, то стоит отметить open

source. Отвечая на вопрос, что это может значить, можно сказать, что ядро любого дистрибутива Linux имеет открытый код, что позволяет улучшить любую программу или само ядро этой операционной системы.

Linux чаще всего взаимодействует с оболочкой Bash или как её ещё называют Bourne Again Shell. На данный момент данная оболочка всё чаще установлена по умолчанию во многих дистрибутивах Linux.

Оболочка Bash включает такие инструменты, как:

- Редактирование командной строки;
- Управление заданиями;
- Неограниченный размер истории команд;
- Функции и псевдонимы оболочек;
- Индексированные массивы неограниченного размера;
- Целочисленная арифметика в любой базе от двух до шестидесяти четырех [\[5\]](#).

Авторами статьи, были написаны следующие коды на операционной системе Linux с использованием Bash для визуализации их влияния на облачную инфраструктуру Mail.

· *Скрипт для автоматического создания нового виртуального сервера:*

```
#!/bin/bash
```

```
# Аутентификация в Облаке Mail.ru
```

```
mcloud login
```

```
# Создание нового виртуального сервера
```

```
mcloud vm create --name my-server --flavor b2.micro --image centos_7
```

· *Скрипт для запуска виртуального сервера по его ID:*

```
#!/bin/bash
```

```
# Аутентификация в Облаке Mail.ru
```

```
mcloud login
```

```
# Запуск виртуального сервера по его ID
```

```
mcloud vm start --id <server_id>
```

· *Скрипт для остановки виртуального сервера по его ID:*

```
#!/bin/bash
```

```
# Аутентификация в Облаке Mail.ru
```

```
mcloud login
```

```
# Остановка виртуального сервера по его ID
```

```
mcloud vm stop --id <server_id>
```

Проанализировав данные коды, можно сделать вывод, что Bash позволяет пользователю создать скрипт, который упростит, а главное, автоматизирует операции с какими-либо облачными ресурсами, так же скрипт позволяет быстро создать виртуальный сервер и настроить его параметры.

Кроме того, основываясь на второй скрипт, можно сказать, что Bash-скрипт позволяет без привлечения особо трудоёмких и затратных ресурсов управлять виртуальным сервером, выполняя различные операции. Например, операцию, приведённую в скрипте, запуск и остановка виртуального сервера по его ID.

Не стоит оставлять без внимания оболочку **Bourne Shell**(sh). История данной оболочки начинается с Shell, вошедшего в поставку первой редакции UNIX, разработанную Кеном Томпсоном. Но данная оболочка была не совсем удобна для работы, так как имела многочисленные ограничения. В результате Стивен Борн переписал стандартный Shell UNIX.

Популярность оболочка завоевала благодаря компактности и преимущественно высокой скоростью работы. Благодаря этим преимуществам она стала оболочкой по умолчанию для ОС Solaris.

Но у оболочки Bourne есть ряд существенных недостатков:

- Нет встроенных функций для обработки логических и арифметических операций.
- Не может запоминать ранее выполненные команды, в отличие от большинства других командных оболочек.
- Не хватает комплексных функций для удобного интерактивного использования [\[6\]](#).

Чтобы просмотреть как происходит взаимодействие Bourne Shell (sh) с облачной платформой Mail на операционной системе Linux можно просмотреть следующие примитивные скрипты кодов:

· *Скрипт для автоматического создания нового виртуального сервера:*

```
#!/bin/sh
```

```
# Аутентификация в Облаке Mail.ru
```

```
mcloud login
```

```
# Создание нового виртуального сервера
```

```
mcloud vm create --name my-server --flavor b2.micro --image centos_7
```

· *Скрипт для запуска виртуального сервера по его ID:*

```
#!/bin/sh
```

```
# Аутентификация в Облаке Mail.ru
```

```
mcloud login
```

```
# Запуск виртуального сервера по его ID
```

```
mcloud vm start --id <server_id>
```

· Скрипт для остановки виртуального сервера по его ID:

```
#!/bin/sh

# Аутентификация в Облаке Mail.ru

mcloud login

# Остановка виртуального сервера по его ID

mcloud vm stop --id <server_id>
```

Авторами отмечено, что скрипт позволяет автоматизировать процесс создания и управления облачными ресурсами. Написание скриптов на Bourne Shell (sh) даёт возможность для создание виртуального сервера, а главное для его настройки. Такие опции позволяют администратору ускорить и оптимизировать рабочий процесс.

Гибкость и управляемость облачными ресурсами через запуск скриптов на Bourne Shell (sh) становится простым и доступным действием, как создание и управление объектными хранилищами.

Microsoft Windows — семейство проприетарных операционных систем корпорации Microsoft, направленных на использование графического интерфейса при управлении. Первоначально Windows представляла графическую надстройку для MS-DOS. Сейчас Windows занимает подавляющую долю на мировом рынке операционных систем [\[7\]](#).

История операционной системы Microsoft Windows начинается с ноября 1985 года. По настоящее время Microsoft Windows выпускает обновления. На декабрь 2024 года самой свежей операционной системой является Windows 11 версии 24H2.

Помимо этого, Microsoft Windows распространяет свои операционные системы на планшеты и смартфоны. Результатом такого решения стала операционная система Microsoft Windows 8, которая была выпущена в октябре 2012 года. Новая система стала значительно быстрее загружаться, но были некоторые проблемы с драйверами и запуском игр.

В Windows 10 Microsoft собирает множество данных об использовании компьютера. Примерами таких данных являются имя, адрес электронной почты и другие. Из-за этого на Microsoft обрушился шквал критики [\[8\]](#).

Одной из старейших приложений и инструментов командной строки, поставляемых с операционной системой Windows, — это командная строка, обычно называемая CMD [\[9\]](#).

Ниже представлен фрагмент стартового окна Command Prompt (cmd.exe) для удобства понимания читателями.

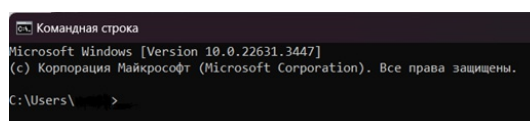


Рисунок 1. Пример Command Prompt (cmd.exe)

Для того чтобы открыть Command Prompt (cmd.exe) нужно выполнить следующие шаги:

1. Нажмите на меню «Пуск» на панели инструментов рабочего стола.

2. Введите команду «cmd» и нажмите Enter.

Наиболее распространённое применение использования Command Prompt (cmd.exe) администраторов заключается в администрировании задач, например, форматирование или управление разделами диска. Помимо этого, Command Prompt (cmd.exe) удобна для работы с просмотром файлов каталогов и управления ими.

Ниже приведены примеры скриптов для работы с облаком Mail.ru, которые можно выполнить в командной строке Command Prompt (cmd.exe) в операционной системе Microsoft Windows:

· *Скрипт для автоматического создания нового виртуального сервера:*

```
mcloud vm create --name my-server --flavor b2.micro --image centos_7
```

· *Скрипт для запуска виртуального сервера по его ID:*

```
mcloud vm start --id <server_id>
```

· *Скрипт для остановки виртуального сервера по его ID:*

```
mcloud vm stop --id <server_id>
```

Использование администраторами командной строки Command Prompt (cmd.exe) обусловлено удобством использования, так как Command Prompt (cmd.exe) позволяет выполнять операции с облачными вычислениями напрямую из стандартной командной оболочки Microsoft Windows, а значит не нужно устанавливать дополнительные инструменты.

Последней рассмотренной в данной статье командной оболочкой будет **Windows PowerShell**.

Windows PowerShell используется администраторами для автоматизации рутинных задач. Так же Windows PowerShell как и предыдущие командные оболочки позволяет менять настройки, останавливать и запускать сервера, обслуживать установленные приложения.

Windows PowerShell позволяет:

- Менять настройки операционной системы;
- Управлять службами и процессами;
- Настраивать роли и компоненты сервера;
- Устанавливать программное обеспечение;
- Управлять установленным ПО через специальные интерфейсы;
- Встраивать исполняемые компоненты в сторонние программы;
- Создавать сценарии для автоматизации задач администрирования;
- Работать с файловой системой, реестром Windows, хранилищем сертификатов и т.д. [\[10\]](#).

Windows PowerShell – командная оболочка, первоначально построенная на базе Microsoft .NET Framework, а позднее на .NET, и интегрирована с ними. В данной

командой оболочке используются разнообразные хранилища, которые напоминают файловую систему, для доступа к которым созданы поставщики.

Стоит рассмотреть и скрипты для выполнения облачных вычислений:

- Скрипт для автоматического создания нового виртуального сервера:

```
mcloud vm create --name my-server --flavor b2.micro --image centos_7
```

- Скрипт для запуска виртуального сервера по его ID:

```
mcloud vm start --id <server_id>
```

- Скрипт для остановки виртуального сервера по его ID:

```
mcloud vm stop --id <server_id>
```

PowerShell предоставляет для использования наиболее широкий набор функций и возможностей, если сравнивать его с командной строкой Command Prompt (cmd.exe).

Синтаксис у данной командной оболочки будет удобнее, что делает его оптимальным при выборе инструмента для автоматизации и написания сложных скриптов. Поддержка автодополнения и подсказки позволяют начинающему пользователю с лёгкостью познать управление данной командной оболочкой.

Анализ производительности с использованием утилиты *Hyperfine*

Для проведения анализа производительности различных оболочек в облачных вычислениях была использована утилита **Hyperfine** [\[11\]](#) — инструмент командной строки, предназначенный для тестирования скорости выполнения команд. Эта утилита предоставляет простой способ оценки времени работы различных скриптов и команд, обеспечивая точность измерений. В рамках исследования были проведены тесты, в которых скрипты выполняли аналогичные задачи, такие как создание, запуск и остановка виртуальных машин в облачной среде Mail.ru. Сравнение времени выполнения скриптов в различных оболочках позволило выявить ключевые различия в их производительности.

Для наглядности авторами разработана таблица (Рис.2) с необходимыми в результате тестирования критериями при использовании командных оболочек:

- Windows PowerShell под управлением ОС Microsoft Windows;
- Bash, под управлением ОС Linux;
- Command Prompt под управлением ОС Microsoft Windows;
- Bourne Shell под управлением ОС Linux.

Представлены результаты выполнения критериев:

- Время запуска (в секундах);
- Время отключения (в секундах).

Оболочка	Операционная система	Время запуска, с	Время отключения, с
Windows PowerShell	Windows	13	14
Bash	Linux	15	17
Command Prompt (cmd.exe)	Windows	24	29
Bourne Shell (sh)	Linux	19	20

Рисунок 2. Сравнительная таблица

Безопасность при использовании различных операционных систем и оболочек**PowerShell:**

Управление доступом: PowerShell поддерживает детализированное управление доступом, что позволяет ограничивать права пользователей и групп, а также настраивать роли для обеспечения безопасности.

Execution Policy: Эта настройка регулирует выполнение скриптов в PowerShell. Включение режимов "Restricted" или "AllSigned" помогает предотвратить запуск неподписанных или вредоносных скриптов.

Шифрование и криптография: PowerShell использует различные механизмы для защиты данных и аутентификации, включая поддержку SSL/TLS.

Уязвимости и атаки: PowerShell является популярной целью для атак, таких как использование вредоносных скриптов, что требует правильной настройки безопасности и применения многофакторной аутентификации (MFA).

Удаленное управление: PowerShell Remoting предоставляет возможность удаленного выполнения команд, что может быть использовано злоумышленниками. Поэтому важно защищать этот канал с помощью сильной аутентификации и настройки ограничений доступа.

Bash:

Переменные среды: Конфиденциальные данные, такие как пароли, могут храниться в переменных среды, что повышает риск утечек, если они не защищены.

Командные инъекции: Bash уязвим к атакам типа Command Injection, если ввод не проверяется должным образом. Поэтому важно тщательно фильтровать все данные, поступающие от пользователя.

Привилегии sudo и root: Использование прав суперпользователя может привести к серьезным уязвимостям, если эти права не ограничены или не контролируются должным образом.

Обновления безопасности: Так как Bash имеет долгую историю, важно регулярно обновлять систему для защиты от известных уязвимостей, таких как Shellshock.

Command Prompt (cmd.exe):

Ограниченная безопасность: Command Prompt имеет ограниченные возможности по работе с криптографией и безопасностью, что делает его менее гибким в плане защиты данных по сравнению с PowerShell.

Устаревшие команды: Использование старых команд в cmd.exe может привести к уязвимостям, таким как утечка данных через команды, которые предназначены для отображения содержимого конфиденциальных файлов.

Уязвимости в BAT-файлах: Скрипты в формате .bat могут быть легко выполнены в cmd.exe, что делает систему уязвимой для внедрения вредоносного кода, если эти файлы не проверяются должным образом.

Интеграция с PowerShell: Поскольку cmd.exe поддерживает интеграцию с PowerShell, уязвимости в PowerShell могут быть использованы через cmd.exe.

Bourne Shell (sh):

Инъекции команд: Подобно Bash, Bourne Shell уязвим к инъекциям команд, если данные не проверяются корректно, что может привести к выполнению произвольного кода.

Скрипты с привилегиями: Скрипты, выполняющиеся с правами суперпользователя, могут стать источником уязвимостей, особенно если они содержат ошибки или неправильно настроены.

Переменные среды: В Bourne Shell также используются переменные среды для хранения конфиденциальных данных, что повышает риск утечек, если они не защищены должным образом.

Аутентификация и шифрование: Современные версии Bourne Shell могут использовать инструменты, такие как SSH, для улучшения безопасности и аутентификации в облачных системах.

Общие угрозы для всех оболочек:

Удаленный доступ: Все оболочки подвергаются риску удаленного выполнения команд, что требует надежной настройки межсетевых экранов, использования шифрования и многофакторной аутентификации.

Уязвимости в скриптах: Ошибки в скриптах могут привести к утечке данных или выполнению вредоносного кода.

Инъекции команд: Командные инъекции являются одной из самых распространенных угроз, что может повлиять на безопасность всей системы.

Привилегированные права: Все оболочки требуют строгого контроля за использованием суперпользовательских прав, чтобы предотвратить утечку данных или компрометацию системы.

Перспективы дальнейших исследований

В рамках дальнейших исследований можно рассмотреть следующие направления:

Изучение новых оболочек и операционных систем: Оценка производительности новых или менее популярных оболочек и операционных систем в контексте облачных вычислений, а также их безопасности.

Оптимизация существующих решений: Разработка методов, направленных на повышение эффективности и безопасности текущих оболочек и операционных систем.

Влияние аппаратного обеспечения: Исследование воздействия различных типов

аппаратного обеспечения на производительность и безопасность облачных сервисов.

Использование искусственного интеллекта: Применение методов машинного обучения и ИИ для автоматизации задач администрирования и оптимизации процессов управления облачными вычислениями.

Заключение

В данной статье, авторами было рассмотрено влияние операционных систем и командных оболочек на облачные вычисления. Операционные системы, Linux и Microsoft Windows, играют важную роль в облачной инфраструктуре. Они обеспечивают стабильность и эффективное использование ресурсов. Командные оболочки, Bash, Bourne Shell (sh), Command Prompt (cmd.exe) и PowerShell, предоставляют различные средства для автоматизации задач и управления облачными ресурсами.

Авторами был проведён анализ основных особенностей каждой операционной системы и командной оболочки, а также были представлены примеры скриптов для работы с облачными вычислениями. Было показано, что выбор операционной системы и командной оболочки может существенно повлиять на скорость выполнения облачных вычислений и управления облачной инфраструктурой.

Если сравнивать коды по критерию, то самой простой и эффективной оболочкой является Windows PowerShell на операционной системе Microsoft Windows. Она позволяет писать лаконичные и простые для понимания коды.

Если говорить про синтаксически обширную оболочку и операционную систему, то тут преимущество будет у оболочек Bash и Bourne Shell (sh) на операционной системе Linux.

Так же авторами была разработана таблица (Рис.3) с необходимыми в результате анализа критериями с использованием командных оболочек которые использовались в этапах исследования:

- Bash, под управлением ОС Linux;
- Bourne Shell под управлением ОС Linux;
- Windows PowerShell под управлением ОС Microsoft Windows;
- Command Prompt под управлением ОС Microsoft Windows.

Проведена оценка на соответствие следующим критериям в оболочках:

- Лаконичность;
- Простота синтаксиса;
- Объёмность кода;
- Удобство использования.

Критерий/Операционная система, оболочка	Оболочка Bash, операционная система Linux	Оболочка Bourne Shell (sh), операционная система Linux	Оболочка Windows PowerShell, операционная система Microsoft Windows	Оболочка Command Prompt (cmd.exe), операционная система Microsoft Windows
Лаконичность	-	-	+	-
Простота синтаксиса	-	-	+	+
Объёмность кода	+	+	+	+
Удобство использования	-	+	+	+

Рисунок 3. Критерии соответствия

Таким образом, понимание роли операционных систем и командных оболочек в облачных вычислениях имеет важное значение для администраторов систем, разработчиков и всех заинтересованных лиц в сфере информационных технологий.

Библиография

1. Савельев Д.Н., Гаврилов С.В. Роль операционных систем в облачных вычислениях: вызовы и перспективы / Савельев Д.Н., Гаврилов С.В. [Электронный ресурс] // Электронный периодический научный журнал sci-article.ru : [сайт]. – URL: <https://sci-article.ru/stat.php/stat.php?i=1697382177> (дата обращения: 15.04.2024).
2. Трубачева С.И. Почему Linux и системы реального времени? [Текст] / Трубачева С.И. // Вестник Волжского университета им. В.Н. Татищева. – 2015. – № 2(24). – С. 99-105.
3. Колосов, Л. С., Умаралиев, И. В. Обзор эволюции функционала операционной системы GNU/Linux при эволюции ядра [Текст] / Л. С. Колосов, И. В. Умаралиев // Научный аспект. – 2023. – № 7. – С. 1390-1393.
4. Passos L., Czarnecki K., Wasowski A. (2012). Towards a catalog of variability evolution patterns: the Linux kernel case. In FOSD '12 Proceedings of the 4th International Workshop on Feature-Oriented Software Development (pp. 62-69). Association for Computing Machinery.
5. Upasana Why do you need the different Linux Shells? / Upasana [Электронный ресурс] // edureka : [сайт]. – URL: <https://www.edureka.co/blog/types-of-shells-in-linux/#differenttypesofshells> (дата обращения: 18.04.2024).
6. siberianMan О разных командных оболочках Linux и Unix / siberianMan [Электронный ресурс] // Хабр : [сайт]. – URL: <https://habr.com/ru/articles/157283/> (дата обращения: 18.04.2024).
7. Леонтьев, В. О., Великосельский, С. А. Семейство Microsoft Windows [Текст] / В. О. Леонтьев, С. А. Великосельский // Аллея Науки. – 2018. – № 3(19). – С. 727-729.
8. Влад Массино, Арам Папоян Три десятилетия Windows / Влад Массино, Арам Папоян [Электронный ресурс] // газета.ru : [сайт]. – URL: https://www.gazeta.ru/tech/2015/11/19/7902437/windows_30th_anniversary.shtml (дата обращения: 18.04.2024).
9. Gaurav Bidasaria Command Prompt vs PowerShell vs Windows Terminal: How They Differ / Gaurav Bidasaria [Электронный ресурс] // techwiser : [сайт]. – URL: <https://techwiser.com/command-prompt-vs-powershell-vs-windows-terminal-comparison/> (дата обращения: 18.04.2024).
10. ru_vds Что такое Windows PowerShell и с чем его едят? Часть 1: основные возможности / ru_vds [Электронный ресурс] // Хабр : [сайт]. – URL: <https://habr.com/ru/companies/ruvds/articles/487876/> (дата обращения: 18.04.2024)
11. sharkdp Hyperfine releases v.1.19.0 / sharkdp [Электронный ресурс] // github : [сайт]. – URL: <https://github.com/sharkdp/hyperfine> (дата обращения: 01.12.2024)

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Статья представляет интересный анализ операционных систем и командных оболочек. Основная часть статьи начинается с исторического обзора операционной системы Linux, отмечая ее значимость и активное участие сообщества разработчиков. Затем

рассматриваются преимущества Linux, в том числе открытый исходный код и взаимодействие с оболочкой Bash. Также выполнен обзор операционных систем Microsoft Windows и Linux, а также их командных оболочек, с акцентом на их роль в облачных вычислениях. Текст содержит информацию о истории и особенностях различных версий Windows, включая Windows 8, Windows 10 и Windows 11. Также рассматривается командная строка Command Prompt и Windows PowerShell, их функциональность и преимущества.

Приведенные примеры кода на Bash позволяют читателям лучше понять, как данная оболочка может быть использована для автоматизации задач управления облачными ресурсами. Примеры ясно демонстрируют функциональность Bash и его преимущества при работе с облачной инфраструктурой.

Общие замечания:

Во-первых, в статье утверждается, что оболочка Bourne Shell (sh) была разработана Стивом Борном. Однако фактически она была создана Кеном Томпсоном.

Далее, некоторые недостатки оболочки Bourne Shell, перечисленные в статье, могут быть недостоверными или устаревшими, так как она является одной из первых оболочек UNIX и в последующих версиях могли быть устранены некоторые из них.

Также следует отметить, что в разделе о Windows PowerShell неправильно утверждается, что командная оболочка была создана на основе платформы .NET Framework. Фактически, она была разработана с использованием языка программирования C# и базируется на .NET Framework или .NET Core в зависимости от версии.

Раздел "Заключение" содержит общие выводы о выборе операционной системы и командной оболочки, но не предоставляет новых исследовательских данных или выводов, которые бы уточнили значимость рассмотренных аспектов для облачных вычислений.

Отдельно хотелось бы отметить несколько ошибок и неточностей в тексте:

1. В некоторых местах статьи присутствуют ошибки в написании слов и использовании пунктуации, что может затруднить понимание текста: например: "на Bourne Shell (sh) становится простим и ...".
2. В разделе о Windows PowerShell некорректно указаны примеры скриптов для работы с облаком Mail.ru. Они дублируются и содержат ошибку в одной из команд - вместо "mcloud vm stop" должно быть "mcloud vm stop".
3. Некоторые изображения, такие как Рис 1, могут быть уменьшены до размеров полезного текста на скриншоте, с целью показа существенной информации без лишнего пространства, заполненного черным цветом или Рис.2 – Логотип PowerShell, который может быть ненужным и не добавлять ценности к тексту.

Заключительное замечание о том, что исследование в области облачных вычислений продолжается, является очевидным и не добавляет информативности.

В целом, статья представляет интересную информацию о роли операционных систем и командных оболочек в облачных вычислениях, но требует доработки с точки зрения правильности изложения и добавления новых выводов на основе проведенного анализа, а также обзора современных технологий, таких как специализированные операционные системы, разработанные для запуска контейнеров, которые становятся все более популярными в облачных средах, и облачные платформы, такие как Amazon Web Services (AWS), Google Cloud Platform (GCP) и Microsoft Azure, которые предлагают разнообразные операционные системы для развертывания в облаке.

Результаты процедуры повторного рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не

раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Предмет исследования. Название статьи свидетельствует о том, что она должна быть посвящена роли операционных систем и оболочек в облачных вычислениях: анализ ОС и оболочек, используемых в облачных платформах и их влияние на облачную инфраструктуру. Содержание статьи не противоречит заявленной теме, но и не раскрывает её полностью, о чём более подробно будет сказано в остальных блоках настоящей рецензии.

Методология исследования базируется преимущественно на изложении общеизвестных фактов и суждений. В отдельных фрагментах статьи применяются методы анализа и синтеза данных. Также автор использует графический инструментарий, однако неясно, что хотел сказать автор, приводя рисунок 1? В тексте статье есть только информация о том, что приведено на рисунке 1, но зачем это представлено и какие выводы делает автор читатель из текущей редакции узнать не сможет. В свою очередь, рисунок 2 представляет, по сути, таблицу, имеющую заголовок «критерии», но неясно критерии осуществления какого действия автор привёл в данной таблице. Также под графическими объектами не приведены источники, что снижает впечатление от ознакомления с рецензируемой статьёй.

Актуальность исследования вопросов, связанных с ролью операционных систем и оболочек в облачных вычислениях: анализ ОС и оболочек, используемых в облачных платформах и их влияние на облачную инфраструктуру, не вызывает сомнения, т.к. это одна из активно обсуждаемых тем в профессиональном сообществе.

Научная новизна в представленном на рецензирование материале не обнаружена.

Стиль, структура, содержание. Стиль изложения научный с точки зрения используемых слов и словосочетаний, но при этом научные методы исследования практически не применяются. Структура статьи автором не выстроена. При проведении доработки статьи рекомендуется использовать следующие компоненты: «Введение», «Постановка проблемы», «Методы и условия исследования», «Результаты исследования», «Обсуждение результатов исследования», «Выводы и дальнейшие направления исследования». Следование данной структуры одновременно решит несколько задач: во-первых, внесёт позитивный вклад в формирование отсутствующей в текущей редакции научной новизны; во-вторых, значительно расширит потенциальную читательскую аудиторию. Также автору рекомендуется обозначить выявленные по итогам исследования проблемы и сформировать комплекс обоснованных рекомендаций по их решению.

Библиография. Библиографический список включает 10 наименований. Ценно, что автор опирается не только на отечественные научные публикации, но и на зарубежные. Это позволяет учесть множество мнений по предмету исследования. Однако 10 источников не является достаточным для всестороннего изучения палитры существующих подходов к исследованию заявленных в заголовке вопросов. Поэтому при проведении доработки статьи рекомендуется увеличить количество источников.

Апелляция к оппонентам. Несмотря на сформированный список источников, какой-либо научной дискуссии в тексте не обнаружено. При проведении доработки и увеличении числа источников рекомендуется обсудить полученные результаты с теми итогами исследований, что содержатся в научных трудах других авторов. Это также окажет позитивное воздействие на формирование отсутствующей в текущей редакции научной новизны.

Выводы, интерес читательской аудитории. В текущей редакции статья не будет востребована у читательской аудитории, но при проведении глубокой содержательной

доработки с учётом высокой актуальности выбранной темы исследования она будет пользоваться спросом у широкого круга лиц.

Результаты процедуры окончательного рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

В статье рассматривается влияние различных операционных систем и командных оболочек на облачные вычисления. Авторы анализируют использование ОС Linux, Microsoft Windows и различных оболочек (Bash, Bourne Shell, Command Prompt, Windows PowerShell) в контексте их применения в облачной инфраструктуре. В современных условиях развития информационных технологий облачные вычисления становятся неотъемлемой частью инфраструктуры множества компаний и организаций. Они позволяют оптимизировать затраты на оборудование, упростить управление ИТ-ресурсами и повысить гибкость и масштабируемость систем. Облака предоставляют возможность пользователям получать доступ к вычислительным мощностям и сервисам по запросу, что особенно важно для быстрого реагирования на изменения в бизнесе и для поддержки инноваций. Исследование основано на сравнительном анализе операционных систем и оболочек, с использованием практических примеров скриптов для создания и управления виртуальными серверами в облачной среде. Авторы приводят коды скриптов, что позволяет наглядно оценить возможности и особенности каждой системы и оболочки. Облачные вычисления становятся все более важными в современном ИТ-ландшафте. Правильный выбор операционной системы и командной оболочки может значительно повлиять на производительность, безопасность и удобство управления облачными ресурсами. Статья актуальна для администраторов систем, разработчиков и всех, кто работает с облачными технологиями. Научная новизна статьи заключается в структурированном подходе к сравнению операционных систем и командных оболочек в контексте их использования в облачных вычислениях. Авторы предоставляют конкретные примеры скриптов, что является ценным вкладом в практическое применение теоретических знаний. Стиль изложения научный, с акцентом на практическую применимость представленных данных. Структура статьи логична и последовательна: введение, описание операционных систем и оболочек, примеры использования, заключение. Содержание статьи полное и соответствует заявленной теме. Авторы делают вывод о значительном влиянии выбора операционной системы и командной оболочки на эффективность облачных вычислений. Приведенные примеры скриптов демонстрируют практическую ценность проведенного исследования. Статья представляет интерес для широкой аудитории, связанной с ИТ и облачными технологиями. Это включает системных администраторов, разработчиков, исследователей и студентов.

Программные системы и вычислительные методы

Правильная ссылка на статью:

Смирнов А.А., Подольский Е.А., Черенков А.В., Государев И.Б. Сравнительный анализ производительности сред исполнения JavaScript кода: Node.js, Deno и Bun // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.72206 EDN: KAZINO URL: https://nbpublish.com/library_read_article.php?id=72206

Сравнительный анализ производительности сред исполнения JavaScript кода: Node.js, Deno и Bun

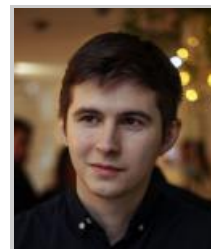
Смирнов Андрей Александрович

ORCID: 0009-0006-6322-6842

студент; факультет "Программная инженерия и компьютерные технологии"; Национальный исследовательский университет ИТМО

197101, Россия, г. Санкт-Петербург, ул. Кронверкский, 49

✉ smirnov.andrew.1999@yandex.ru



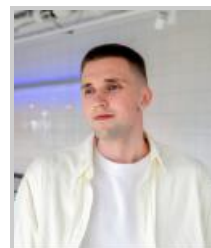
Подольский Егор Александрович

ORCID: 0009-0002-8311-8882

студент; факультет "Программная инженерия и компьютерные технологии"; Национальный исследовательский университет ИТМО

195112, Россия, г. Санкт-Петербург, ул. Заневский, 28-30-32, кв. 614

✉ egorpodolskij51@yandex.ru



Черенков Артем Вячеславович

студент; факультет "Программная инженерия и компьютерные технологии"; Национальный исследовательский университет ИТМО

197101, Россия, г. Санкт-Петербург, ул. Кронверкский, 49 лит. А

✉ art.cherenkov@gmail.com



Государев Илья Борисович

ORCID: 0000-0003-4236-5991

кандидат педагогических наук

доцент; факультет "Программная инженерия и компьютерные технологии"; Национальный исследовательский университет ИТМО

197046, Россия, г. Санкт-Петербург, Кронверкский пр-т, 49

✉ goss@itmo.ru



[Статья из рубрики "Операционные системы "](#)

DOI:

10.7256/2454-0714.2024.4.72206

EDN:

KAZINO

Дата направления статьи в редакцию:

04-11-2024

Аннотация: Предметом исследования послужила производительность исполнения программ на JavaScript в современных средах Node.js, Deno и Bun. Эти платформы используются для разработки серверных приложений и имеют значительные различия в архитектуре, функциональных возможностях и производительности. Node.js — это наиболее зрелое и широко распространенное решение, которое активно используется в большинстве современных веб-приложений. Deno — это более новая среда, разработанная создателем Node.js, предлагающая улучшенную безопасность, поддержку TypeScript и другие нововведения. Bun, в свою очередь, представляет собой современную и высокопроизводительную альтернативу, ориентированную на скорость работы с серверными приложениями. Цель исследования заключается в выявлении различий в производительности основных современных сред исполнения (Node.js, Deno и Bun) для дальнейшего применения этих сред в разработке серверной части веб-приложений. Для проведения исследования использовался метод компьютерного эксперимента с применением Docker-контейнеров и автоматизации процессов с помощью Ansible. Измерялось время выполнения различных сценариев в каждой из сред исполнения. Научная новизна данного исследования заключается в том что впервые предложена целостная и обоснованная методика измерения и сравнения производительности кода на JavaScript применительно к современным средам исполнения, которая позволит исследователям в дальнейших экспериментах опираться на предложенный подход и распространить его на новые среды исполнения. Результаты исследования показывают, что Bun демонстрирует наилучшую производительность в синхронных вычислениях (сортировка, обработка JSON), но уступает Node.js и Deno в проверке на простоту числа. Deno проявил высокую эффективность в асинхронных операциях, благодаря использованию Rust и библиотеки Tokio. Node.js, несмотря на более низкие результаты в синхронных задачах, показал стабильную производительность в тестах и остается надежным выбором для крупных проектов. В ходе исследования были разработаны рекомендации по выбору подходящей среды выполнения серверного JavaScript кода для различных задач.

Ключевые слова:

JavaScript, Node.js, Deno, Bun, Производительность, Компьютерный эксперимент, Бэкенд, Веб, Сервер, Docker

Введение

JavaScript был анонсирован в 1996 году. Целью языка было добавление интерактивности на статичные веб-страницы. С этого момента прошло много времени. JavaScript стал одним из наиболее популярных языков программирования, а область его применения расширилась. Ключевым расширением функциональности языка стало появление среды

исполнения серверного JavaScript кода - Node.js в 2009 году. Node.js позволила использовать один язык как для разработки клиента, так и для создания сервера. Это привело к активному развитию технологий на базе JavaScript, таких как серверные приложения, микросервисы и API.

Несовершенство Node.js и востребованность в средах выполнения серверного JavaScript кода подтолкнули индустрию к развитию и созданию новых решений. Создатель Node.js Райан Даль в 2018 представил новую технологию - Deno, которая должна была решить все проблемы предшественника. Вслед на Deno в 2022 году была анонсированная новая среда - Bun. Каждая из представленных технологий предлагает уникальные подходы к безопасности, производительности и удобству разработки. Однако, несмотря на растущее число исследований и статей о сравнении сред выполнения JavaScript, до сих пор не проводилось комплексного анализа всех трех технологий, учитывающего все ключевые критерии.

Объектом данного исследования является язык JavaScript и исполнение программного кода на этом языке.

Предметом исследования послужила производительность исполнения программ на JavaScript в современных средах.

Цель исследования – выявить различия в производительности основных современных сред исполнения (Node.js, Deno и Bun) для дальнейшего применения этих сред в разработке серверной части веб-приложений.

Научная новизна данного исследования заключается в том что впервые предложена целостная и обоснованная методика измерения и сравнения производительности кода на JavaScript применительно к современным средам исполнения, которая позволит исследователям в дальнейших экспериментах опираться на предложенный подход и распространить его на новые среды исполнения.

Перспективы

Область программных систем на веб-платформе развивается крайне динамично и сопровождается разработкой новых сред исполнения. До настоящего времени эти среды разрабатывались за рубежом. В русле импортозамещения планируется выработать систему критериев оценки производительности исполнения JavaScript-кода и использовать её для разработки отечественных решений в данной сфере. В числе прочего дальнейшие исследования будут направлены на разработку российских компонентов облачных инфраструктур, обеспечивающих безопасные и эффективные методы поддержки серверных решений, основанных на экосистеме JavaScript.

Анализ литературы

В научной литературе тема сравнения производительности сред исполнения JavaScript кода исследована недостаточно. Большинство работ фокусируются на сравнении производительности различных языков программирования или сравнивают только две среды исполнения [\[1, 2\]](#). Официальные сайты Node.js, Deno и Bun заявляют о превосходстве своих продуктов, но часто без подробных экспериментальных данных.

В работе [\[3\]](#) проведено сравнение Deno и Bun, где показано, что Bun выделяется в обработке массивов и рекурсивных вычислениях, а Deno — в асинхронных задачах. В статье [\[4\]](#) проведено сравнение Node.js и Bun, где отмечается, что Bun обещает

значительно более быстрый запуск, что делает его привлекательным для бессерверных функций.

В статье [5] отмечается, что хотя Bun все еще нов и требуется больше всесторонних испытаний, начальные тесты показывают многообещающие результаты. Легкая модель корутин и оптимизированный сборщик мусора в Bun могут обеспечить улучшенную производительность для нагрузки, связанной с вводом-выводом. Однако, относительная новизна Bun и меньшее сообщество по сравнению с Node.js могут создать препятствия для его широкого применения.

В работе [6] проведено сравнение производительности Node.js и Deno, показывающее приблизительно равный уровень производительности в задачах преобразования координат и поиска прямых линий в изображениях. Это связано с использованием одного и того же движка V8.

В книге [7] описываются основные требования к профессиональному бенчмаркингу: повторяемость, проверяемость и переносимость, принцип невмешательства, приемлемый уровень точности и честность. Авторы также рассматривают риски бенчмаркинга, такие как неточные замеры времени и оптимизация компилятора. Все эти требования и рекомендации учтены в данном исследовании.

В статье [8] рекомендуется использовать график "ящик с усами" для наглядного представления распределения данных. Этот график отображает распределение данных через их квартили и выявляет потенциальные выбросы, что особенно полезно для сравнения распределений между различными группами или условиями.

Исследуемая в работе [9] модель M/G/m queue основывается на том, что современные веб-серверы обладают очередью запросов и пулом потоков для обработки входящих запросов. В средах исполнения JavaScript вместо пула потоков используется механизм Event Loop, который обеспечивает асинхронность выполнения операций. Синхронные операции, однако, выполняются в основном потоке, который существует в единственном экземпляре. Если этот поток будет занят выполнением ресурсоемких вычислений, то время ожидания ответакратно увеличится.

В статье [10] проводится экспериментальное исследование с целью измерения и сравнения производительности Node.js и Jetty в контексте обработки REST-запросов. В рамках данного исследования сервер выполняет ряд операций, характерных для каждого запроса: парсинг JSON, создание объектов, их деструктуризация, обработка, сбор данных и формирование ответа в формате JSON. Эти операции, являющиеся неотъемлемой частью обработки каждого запроса, были выбраны для анализа производительности в данной работе.

Методология эксперимента

Для обеспечения воспроизводимости и переносимости экспериментальных вычислений все операции выполнялись в изолированной среде Docker-контейнеров. Создание Docker-образов осуществлялось на базе стандартных образов, доступных в репозитории Docker Hub (таблица 1). В качестве основы были выбраны Alpine-образы, которые отличаются минималистичностью и повышенной безопасностью.

Таблица 1 – Используемые базовые Docker-образы

Node.js	Deno	Bun
---------	------	-----

node:21-alpine	denoland/deno:alpine-1.42.0	oven/bun:1.1.2-alpine
----------------	-----------------------------	-----------------------

Исследования проводились в облачных вычислительных средах с использованием виртуальных машин (таблица 2). Такой подход позволяет приблизить условия эксперимента к реальным производственным условиям.

Таблица 2 – Характеристики виртуальной машины

Операционная система	Процессор	ОЗУ
ubuntu-22.04-live-server-amd64.iso	2x2.2 ГГц	2 Гб

Описание процесса эксперимента представлено на рисунке 1.

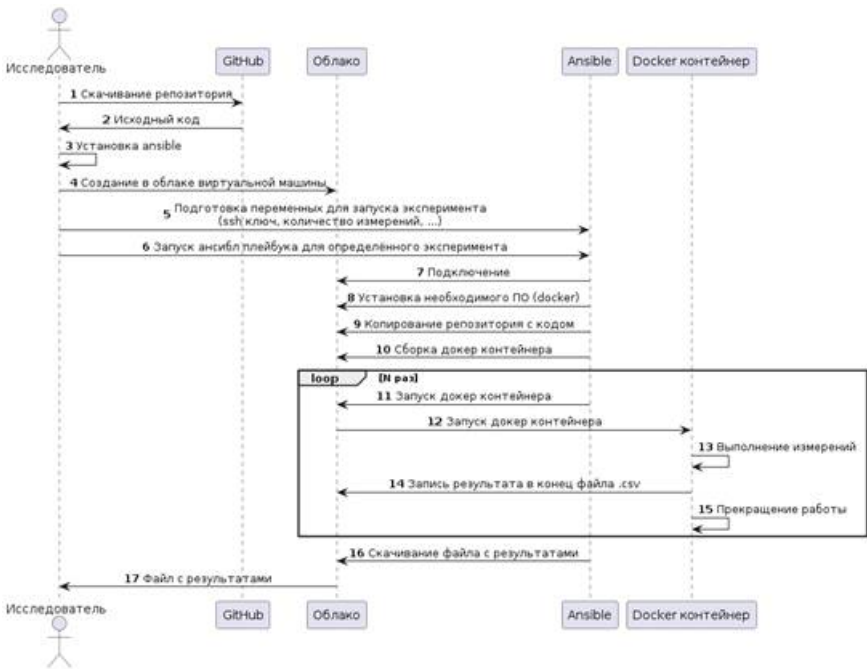


Рисунок 1 – Процесс проведения эксперимента

Эксперимент включал измерение времени выполнения следующих сценариев каждый из которых запускался 1000 раз для обеспечения достаточного количество элементов в выборке:

- Быстрая сортировка. Быстрая сортировка массива из 10000 чисел, сгенерированных с помощью функции Math.random().
- Обработка JSON. JSON.stringify, а затем JSON.parse объекта с 10000 ключами без вложенностей.
- Деструктуризация. Создание нового объекта из двух объектов, у каждого из которых по 10000 ключей, с помощью деструктуризации.
- Object.assign. Создание нового объекта из двух объектов, у каждого из которых по 10000 ключей, с помощью Object.assign.
- Проверка на простоту числа. Является ли число 5600748293801 простым.
- Обработка промисов. Создание и await пустого промиса в цикле 1 млн раз.

Результаты эксперимента

По результатам эксперимента составлены сравнительные таблицы с основными

характеристиками распределения и построены графики. В ходе эксперимента было зафиксировано, что средняя загрузка двухъядерного центрального процессора составила 58%. Это явление можно объяснить следующим образом: одно из ядер процессора испытывало максимальную нагрузку в связи с выполнением задач однопоточных движков, в то время как другое ядро обеспечивало выполнение фоновых процессов операционной системы.

В ходе эксперимента наблюдалось постепенное увеличение объема оперативной памяти, используемой для выполнения задач, с 400 Мб до 500 Мб. Выделение 2 Гб виртуальной памяти оказалось избыточным для выполнения задач.

Deno показал наилучшую производительность при выполнении быстрой сортировки массива, с наименьшим средним временем выполнения 17,3 мс и минимальным значением 11,5 мс. Bun занял второе место со средним временем 19,1 мс, а минимальным значением 13,2 мс. Node.js показал наихудшие результаты со средним временем 23,5 мс. Числовые показатели представлены в таблице 3, а их визуализация на рисунке 3.

Таблица 3 – Результаты быстрой сортировки массива

Время, мс	Node.js	Deno	Bun
Среднее	23,5	17,3	19,1
Медиана	21,5	16,8	18,3
Максимальное	51,4	42,0	33,5
Минимальное	16,3	11,5	13,2
Среднеквадратичное отклонение	4,6	2,7	3,0

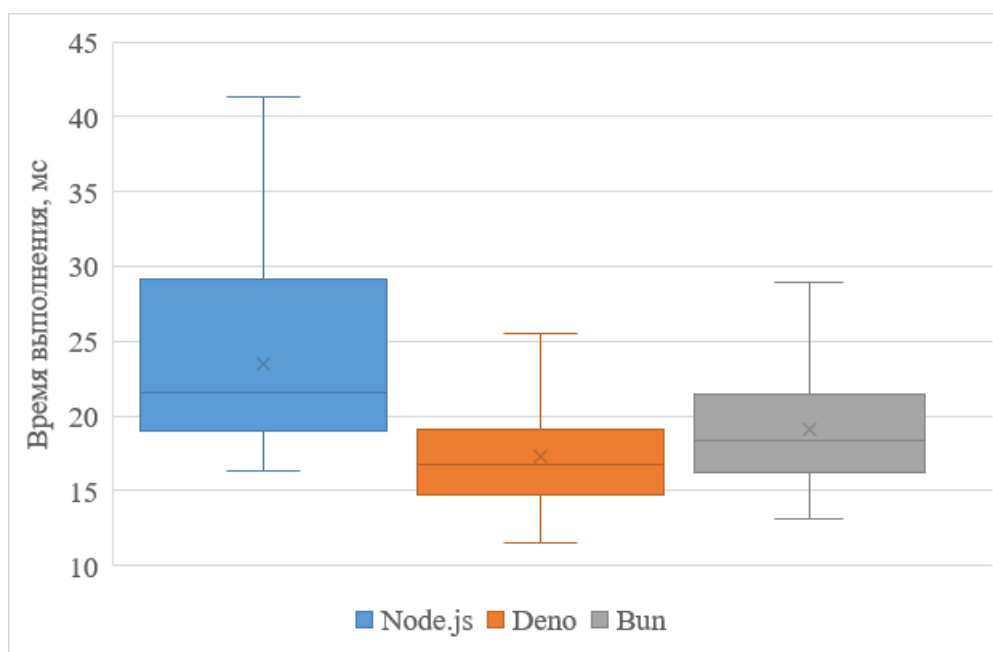


Рисунок 2 – Ящик с усами для результатов быстрой сортировки

Большое, относительно других экспериментов, среднеквадратичное отклонение объясняется тем, что время выполнения алгоритма быстрой сортировки зависит от того, как в массиве распределены элементы, которые в данном эксперименте генерировались случайным образом.

Для повышения наглядности масштаба основного распределения выбросы не отображены на графических иллюстрациях, поскольку их присутствие затрудняет визуальное восприятие распределения данных.

Bun продемонстрировал наилучшую производительность в обработке JSON, со средним временем выполнения 5,8 мс и минимальным значением 5,3 мс. Deno занял второе место со средним временем 13,4 мс, а Node.js показал наихудшие результаты со средним временем 15,0 мс. Среднеквадратичное отклонение было самым низким у Bun (0,3 мс), что указывает на высокую стабильность его производительности. Максимальное время больше всего отличается от медианы (на 56%) у Node.js, что свидетельствует о значительном выбросе. Числовые показатели представлены в таблице 4, а их визуализация на рисунке 4.

Таблица 4 – Результаты обработки JSON

Время, мс	Node.js	Deno	Bun
Среднее	15,0	13,4	5,8
Медиана	14,5	13,1	5,7
Максимальное	22,6	15,1	7,3
Минимальное	14,0	12,8	5,3
Среднеквадратичное отклонение	0,9	0,5	0,3

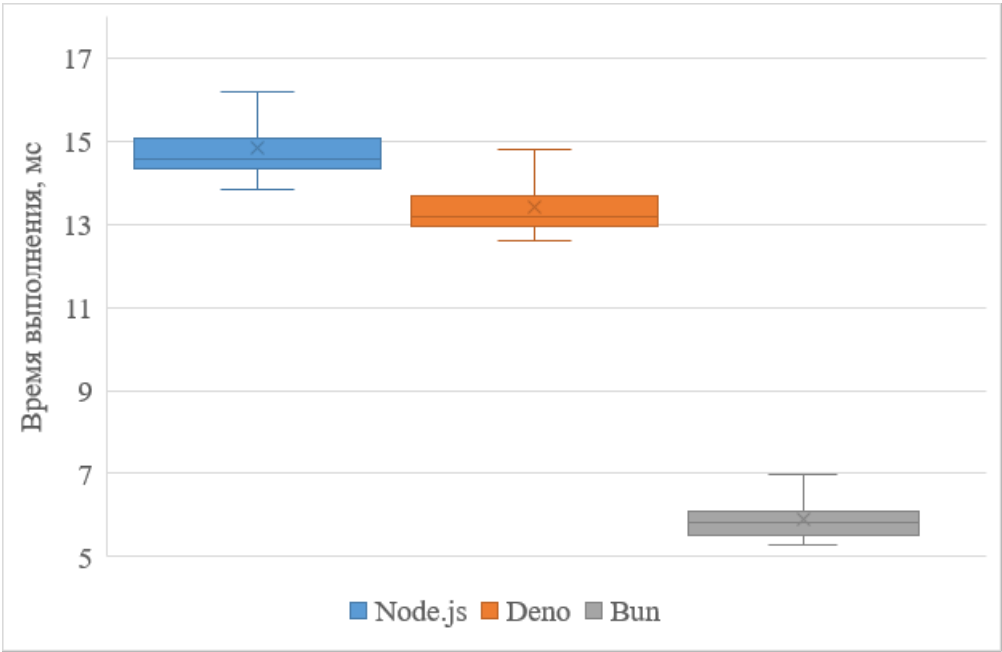


Рисунок 3 – Ящик с усами для результатов обработки JSON

Bun значительно превосходит Deno и Node.js в выполнении операций деструктуризации, со средним временем 3,5 мс и минимальным значением 3,1 мс. Deno занял второе место со средним временем 15,5 мс, а Node.js показал наихудшие результаты со средним временем 17,0 мс. Среднеквадратичное отклонение у Bun было самым низким (0,2 мс), что свидетельствует о стабильной производительности. Числовые показатели представлены в таблице 5, а их визуализация на рисунке 5.

Таблица 5 – Результаты деструктуризации

Время, мс	Node.js	Deno	Bun
Среднее	17,0	15,5	3,5
Медиана	16,7	15,3	3,4
Максимальное	45,2	33,1	7,1
Минимальное	15,8	14,2	3,1
Среднеквадратичное отклонение	0,7	0,7	0,2

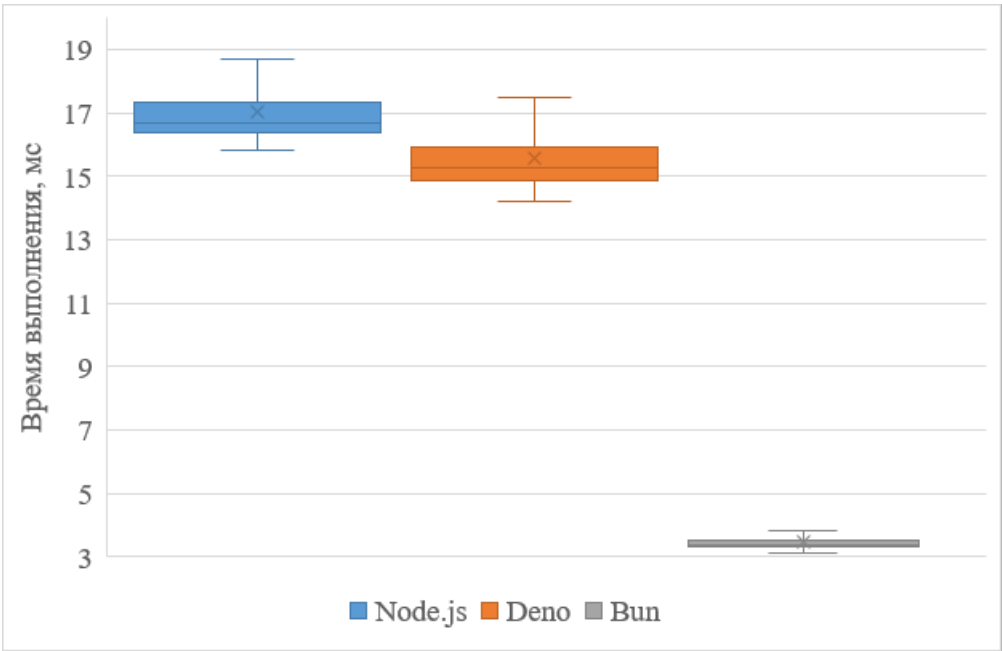


Рисунок 4 – Ящик с усами для результатов деструктуризации

Bun продемонстрировал наилучшую производительность в операции `Object.assign`, со средним временем 3,3 мс и минимальным значением 3,0 мс. Deno показал среднее время выполнения 16,9 мс, а Node.js – 18,3 мс. Среднеквадратичное отклонение у Bun было самым низким (0,2 мс), что указывает на стабильную производительность. Числовые показатели представлены в таблице 6, а их визуализация на рисунке 6.

Таблица 6 – Результаты `Object.assign`

Время, мс	Node.js	Deno	Bun
Среднее	18,3	16,9	3,3
Медиана	18,1	16,6	3,2
Максимальное	44,0	32,7	11,1
Минимальное	17,0	15,8	3,0
Среднеквадратичное отклонение	0,6	0,7	0,2

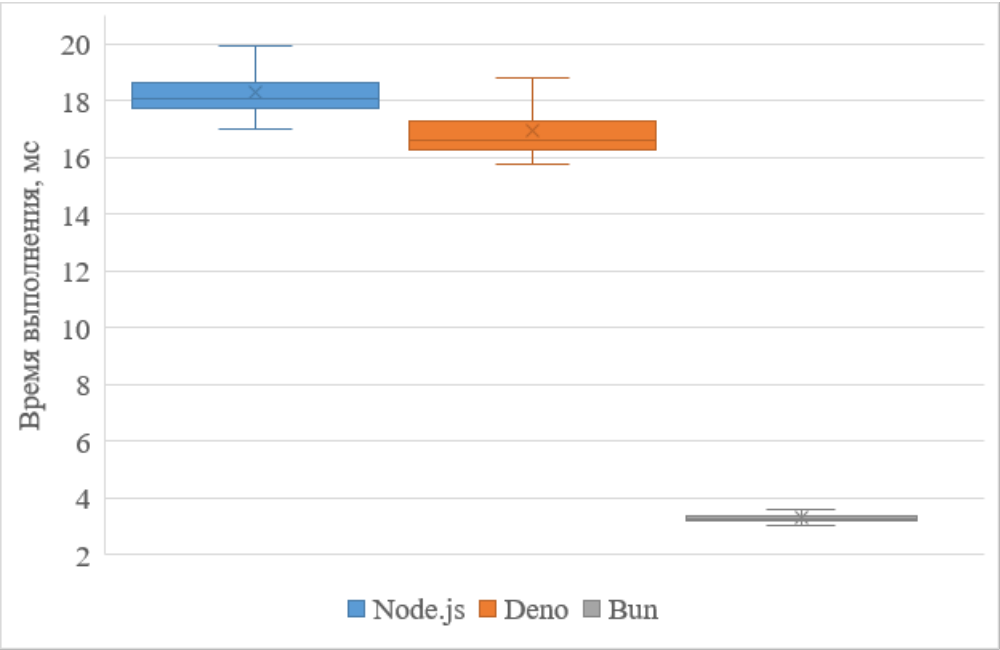


Рисунок 5 – Ящик с усами для результатов Object.assign

Node.js и Deno показали схожие результаты при проверке числа на простоту, со средними значениями 16,7 мс и 16,0 мс соответственно. Bun продемонстрировал значительно худшие результаты, со средним временем выполнения 119,5 мс. Среднеквадратичное отклонение у Bun было самым высоким (2,7 мс), что свидетельствует о меньшей стабильности. Числовые показатели представлены в таблице 7, а их визуализация на рисунке 7.

Таблица 7 – Результаты проверки на простоту числа

Время, мс	Node.js	Deno	Bun
Среднее	16,7	16,0	119,5
Медиана	16,3	15,7	119,7
Максимальное	50,8	27,0	150,8
Минимальное	15,7	14,9	113,1
Среднеквадратичное отклонение	0,8	0,8	2,7

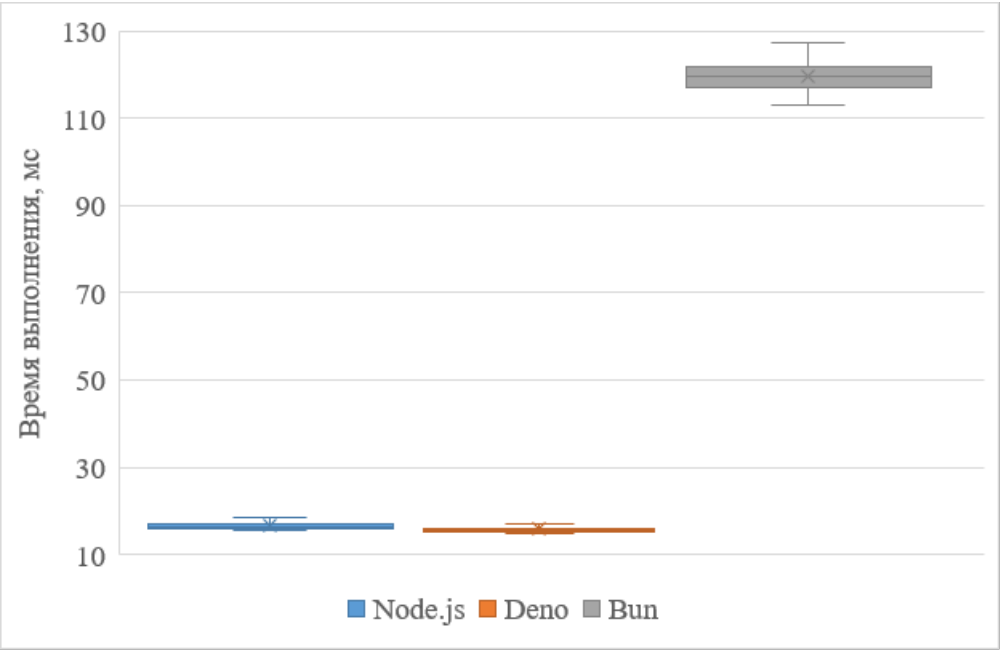


Рисунок 6 – Ящик с усами для результатов проверки на простоту числа

Таблица 8 – Результаты обработки промисов

Время, мс	Node.js	Deno	Bun
Среднее	195,5	173,1	230,1
Медиана	194,9	172,3	229,4
Максимальное	233,4	193,0	257,3
Минимальное	183,7	162,8	214,5
Среднеквадратичное отклонение	3,9	4,0	5,7

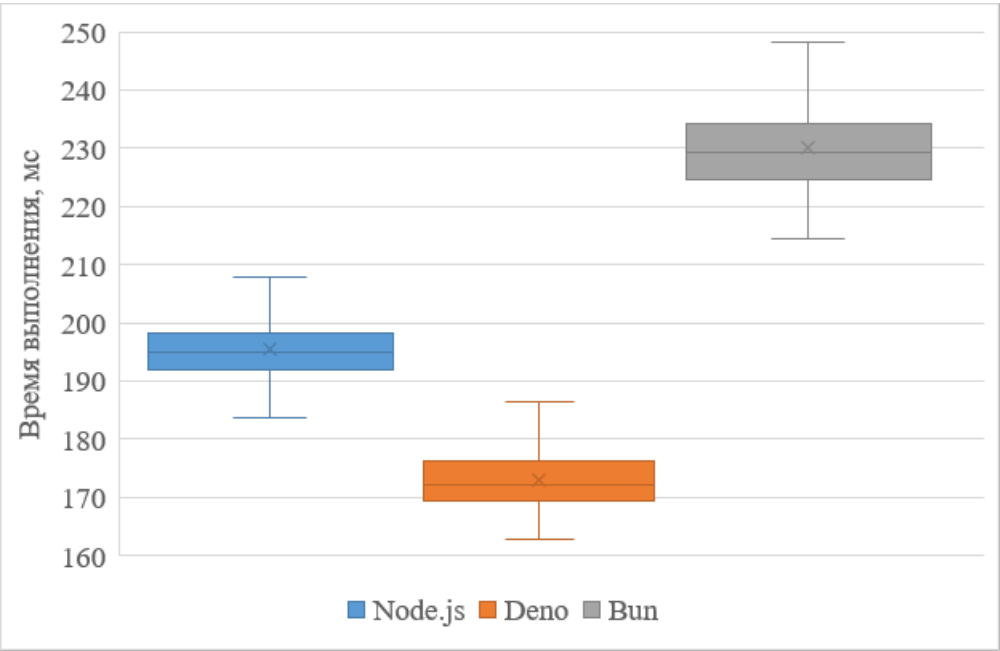


Рисунок 7 – Ящик с усами для результатов обработки промисов

Deno показал наилучшую производительность при обработке промисов, со средним временем выполнения 173,1 мс и минимальным значением 162,8 мс. Node.js занял второе место со средним временем 195,5 мс, а Bun показал наихудшие результаты со средним временем 230,1 мс.

Заключение

В данном исследовании проведён комплексный анализ производительности сред исполнения JavaScript кода: Node.js, Deno и Bun в различных сценариях, включая быструю сортировку, обработку JSON, деструктуризацию, Object.assign, проверку на простоту числа и обработку промисов. Эксперимент был выполнен в изолированной среде Docker-контейнеров с использованием инструмента Ansible для автоматизации процесса, что обеспечило воспроизводимость и переносимость результатов.

Результаты эксперимента показали, что Bun демонстрирует наилучшую производительность в большинстве синхронных вычислений, таких как быстрая сортировка, обработка JSON, деструктуризация и Object.assign. Однако, в сценарии проверки на простоту числа Bun значительно уступает Node.js и Deno, что может быть обусловлено особенностями реализации расчёта остатка от деления в Bun. Deno, в свою очередь, демонстрировал высокую эффективность в асинхронных операциях, таких как обработка промисов, благодаря тому, что он разработан на языке Rust и использует библиотеку Tokio для асинхронных задач, что способствовало его оптимизированной асинхронной архитектуре.

Node.js, несмотря на то, что он не выигрывает в большинстве синхронных сценариев, обеспечивает стабильную производительность во всех тестах и остается надежным выбором для крупных проектов, где важна проверенная временем стабильность и наличие специализированных библиотек. Более того, широкое распространение и большее количество разработчиков, знакомых с Node.js, делает его предпочтительным в тех случаях, когда требуется общая совместимость и поддержка сообщества.

Рекомендуется использовать Bun для задач, требующих высокой скорости сборки и эффективности в синхронных вычислениях, особенно в бессервисных функциях. Однако, при выборе Bun для задач, связанных с асинхронными операциями или специфическими вычислениями, следует проводить дополнительные исследования производительности, так как в некоторых сценариях он может оказаться менее эффективным, чем Node.js или Deno.

Deno, благодаря своей высокой производительности в асинхронных операциях и акценту на безопасности, является предпочтительным выбором для задач, где важен баланс между безопасностью и производительностью. Кроме того, Deno показал себя более быстрым, чем Node.js, во всех сценариях.

Предполагается, что похожие результаты производительности между Node.js и Deno могут быть обусловлены тем, что обе среды исполнения используют одинаковый движок V8, что подтверждает важность учёта движка при оценке производительности JavaScript сред.

В таблице 9 приведены относительные временные характеристики вычислений медиан, где минимальное время принято за 100%, для удобного сравнения результатов между собой.

Таблица 9 – Относительные результаты вычислений для медиан

Название эксперимента	Node.js	Deno	Bun
Быстрая сортировки массива	128%	100%	109%
Обработка JSON	254%	230%	100%
Деструктуризация	491%	450%	100%
Object.assign	566%	519%	100%
Проверка на простоту числа	104%	100%	762%
Обработки промисов	113%	100%	133%

В целом, выбор среды исполнения JavaScript должен основываться на конкретных требованиях проекта, таких как тип вычислений (синхронные или асинхронные), необходимость безопасности, стабильность. Результаты данного исследования предоставляют разработчикам обоснованные рекомендации для выбора наиболее подходящей среды в зависимости от специфики задачи.

Библиография

1. Martinsen, J., Grahn, H. A methodology for evaluating JavaScript execution behavior in interactive web applications. // В материалах 9-й Международной конференции по компьютерным системам и приложениям IEEE/ACS (AICCSA), 2011. С. 241-248. DOI: 10.1109/AICCSA.2011.6126611.
2. Родыгина И. В., Наливайко А. В. Сравнительный анализ технологий для разработки серверной части системы управления продажами: статья в журнале // Известия ЮФУ. Технические науки. № 4 (221). 2021. С. 256-266. DOI: 10.18522/2311-3103-2021-4-256-266.
3. Тихонов Д.С., Черенков А.В., Долгов А.В. Сравнительный анализ технологий серверной разработки на платформах deno и bun // Научно-технические инновации и веб-технологии. 2023. № 2. С. 55-59. URL: <https://elibrary.ru/item.asp?id=56409551>.
4. Kniazev I., Fitiskin A. Choosing the right javascript runtime: an in-depth comparison of node.js and bun: статья в журнале // Norwegian journal of development of the international science. 2023. С. 72-84. DOI: 10.5281/zenodo.7945166. URL: <https://elibrary.ru/item.asp?id=53844306>.
5. Koper D., Woda M. Performance Analysis and Comparison of Acceleration Methods in javascript Environments Based on Simplified Standard Hough Transform Algorithm //International Conference on Dependability and Complex Systems. Cham: Springer International Publishing, 2022. С. 131-142. DOI: 10.1007/978-3-031-06746-4_13.
6. Акиншин А. Профессиональный бенчмарк: искусство измерения производительности. спб.: Питер, 2022. 576 с.: ил. (Серия «Библиотека программиста»). ISBN 978-5-4461-1551-8.
7. Сальникова, К. В. Анализа массива данных с помощью инструмента визуализации "Ящик с усами" / К. В. Сальникова // Universum: экономика и юриспруденция. 2021. № 6(81). С. 11-17. DOI 10.32743/unilaw.2021.81.6.11778.
8. Gregg B. BPF Performance Tools (Addison-Wesley Professional Computing Series) // Update. Т. 517. С. 18. 2019.
9. Lu J., Gokhale S. S. performance analysis of a Web Server //International Journal of Information Technology and Web Engineering (IJITWE). – 2008. – Т. 3. – № 3. – С. 50-65.
10. Lundar J. A., Grønli T. M., Ghinea G. Performance evaluation of a modern web

architecture // International Journal of Information Technology and Web Engineering (IJITWE). – 2013. – Т. 8. – № 1. – С. 36-50.

11. Смирнов А. А., Черенков А. В., Подольский Е. А. Сравнительный анализ сред исполнения Javascript кода Node.js, Deno и Bun для разработки серверной части веб-приложения // Международный журнал информационных технологий и энергоэффективности. 2024. Т. 9, № 4(42). С. 100-106.

12. Lei, K., Ma, Y., & Tan, Z. (2014). Performance Comparison and Evaluation of Web Development Technologies in PHP, Python, and Node.js. // В материалах 17-й Международной конференции IEEE по вычислительной науке и инженерии (CSE), 2014. С. 661-668. DOI: 10.1109/CSE.2014.142.

13. Суворов Д. А. Измерение текущего быстродействия процессоров и качества программ. Способы оценки и повышения реальной производительности //Открытое образование. 2009. № 6. С. 59-65.

14. Пентковский В. М. и др. Методология оценки производительности вычислительных систем //Актуальные проблемы современной науки. 2012. № 6. С. 358-363.

15. Жуйков Р., Шарыгин Е. Методы предварительной оптимизации программ на языке JavaScript //Труды Института системного программирования РАН. 2015. Т. 27. № 6. С. 67-86.

16. B. Basumatary, N. Agnihotri BENEFITS AND CHALLENGES OF USING NODE.JS // International Journal of Innovative Research in Computer Science & Technology. 2022. № 10-3. С. 67-70. URL: <https://acspublisher.com/journals/index.php/ijircst/article/view/10433/9623>

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Представленная статья на тему «Сравнительный анализ производительности сред исполнения JavaScript кода: Node.js, Deno и Bun» соответствует тематике журнала «ПРОГРАММНЫЕ СИСТЕМЫ И ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ» и посвящена актуальному вопросу, связанному с ростом популярности JavaScript и его применения в веб-разработке, что приводит к выбору оптимальной среды исполнения. JavaScript, изначально предназначенный для динамического взаимодействия на веб-страницах имеет широкое использование в серверных приложениях, микросервисах и других высоконагруженных системах. В таких условиях разработчики сталкиваются с необходимостью оценить производительность различных сред исполнения, таких как Node.js, Deno и Bun, чтобы выбрать наиболее подходящую для конкретных задач.

В статье представлен достаточно широкий анализ литературных российских и зарубежных источников по вопросам сравнения Deno и Bun, сравнения производительности Node.js и Deno, экспериментального исследования с целью измерения и сравнения производительности Node.js и Jetty в контексте обработки REST-запросов.

Авторами самостоятельно проведен комплексный анализ производительности Node.js, Deno и Bun в различных сценариях исполнения JavaScript кода и др. параметров.

Стиль и язык изложения материала является достаточно доступным для широкого круга читателей.

Практическая значимость статьи четко обоснована. Статья по объему соответствует рекомендуемому объему от 12 000 знаков.

Статья достаточно структурирована - в наличии введение, заключение, внутреннее

членение основной части (анализ литературы, методология эксперимента, результаты эксперимента).

Авторами проведен эксперимент по воспроизводимости и переносимости вычислений, выполненных в изолированной среде Docker-контейнеров. Создание Docker-образов осуществлялось на базе стандартных образов, доступных в репозитории Docker Hub, результаты обобщены в виде таблицы. В качестве основы были выбраны Alpine-образы, которые отличаются минималистичностью и повышенной безопасностью.

Будет целесообразным добавить в водную часть статьи краткое рассмотрение истории вопроса применения JavaScript для подведения к современному положению дел и обоснованию актуальности рассматриваемой проблемы.

К недостаткам можно отнести следующие моменты: из содержания статьи не прослеживается научная новизна. Отсутствует четкое выделение предмета, объекта и цели исследования, сжатые выводы.

Рекомендуется четко обозначить научную новизну исследования, сформулировать предмет, объект, цель исследования и развернуто написать выводы (содержащие по необходимости рекомендации, оценки, предложения, гипотезы и др.). Также будет целесообразным добавить о перспективах дальнейшего исследования.

Статья «Сравнительный анализ производительности сред исполнения JavaScript кода: Node.js, Deno и Bun» требует доработки по указанным выше замечаниям. После внесения поправок рекомендуется к повторному рассмотрению редакцией рецензируемого научного журнала.

Результаты процедуры повторного рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Рецензируемая статья посвящена сравнительному анализу производительности различных сред исполнения JavaScript кода, предметом исследования в работе выступает производительность исполнения программ в современных средах: Node.js, Deno и Bun.

Методология исследования базируется на проведении экспериментальных вычислений в изолированной среде Docker-контейнеров в облачных вычислительных средах с использованием виртуальных машин, а также обобщении сведений из ранее опубликованных работ, посвященных решаемой проблеме.

Актуальность работы авторы связывают с тем, что JavaScript стал одним из наиболее популярных языков программирования, существующие среды исполнения серверного JavaScript кода: Node.js, Deno и Bun предлагают уникальные подходы к безопасности, производительности и удобству разработки, однако до сих пор не проводилось комплексного анализа всех трех технологий, учитывающего все ключевые критерии.

Научная новизна рецензируемого исследования состоит в том, что авторами предложена целостная и обоснованная методика измерения и сравнения производительности кода на JavaScript применительно к современным средам исполнения, которая позволит исследователям в дальнейших экспериментах опираться на предложенный подход и распространить его на новые среды исполнения.

В статье структурно выделены следующие разделы: Введение, Перспективы, Анализ литературы, Методология эксперимента, Результаты эксперимента, Заключение и Библиография.

Авторами приведен обзор литературы по теме публикации; детально отражен ход проведения эксперимента, описаны используемые базовые Docker-образы, приведены

характеристики виртуальной машины, представлена схема процесса проведения эксперимента; показаны результаты быстрой сортировки массива, обработки JSON, деструктуризации, операции `Object.assign`, проверки на простоту числа и обработки промисов в трёх средах. В Заключении сказано, что Bun демонстрирует наилучшую производительность в большинстве синхронных вычислений, таких как быстрая сортировка, обработка JSON, деструктуризация и `Object.assign`; Node.js обеспечивает стабильную производительность во всех тестах и остается надежным выбором для крупных проектов, где важна проверенная временем стабильность и наличие специализированных библиотек; Deno, благодаря своей высокой производительности в асинхронных операциях и акценту на безопасности, является предпочтительным выбором для задач, где важен баланс между безопасностью и производительностью. Полученные результаты исследования предоставляют разработчикам обоснованные рекомендации для выбора наиболее подходящей среды в зависимости от специфики задачи.

Библиографический список включает 16 источников – публикации отечественных и зарубежных ученых на русском и английском языках по теме статьи, на которые в тексте имеются адресные ссылки, подтверждающие наличие апелляции к оппонентам.

Статья отражает результаты проведенного авторами исследования, соответствует направлению журнала «Программные системы и вычислительные методы», содержит элементы научной новизны и практической значимости, может вызвать интерес у читателей, рекомендуется к опубликованию.

Программные системы и вычислительные методы

Правильная ссылка на статью:

Копышева Т.Н., Митрофанова Т.В., Смирнова Т.Н., Христофорова А.В. Применение платформ Neural Network Wizard и Loginom Community для решения задач аппроксимации и классификации в образовательном процессе // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.71438 EDN: KLWJBM URL: https://nbpublish.com/library_read_article.php?id=71438

Применение платформ Neural Network Wizard и Loginom Community для решения задач аппроксимации и классификации в образовательном процессе

Копышева Татьяна Николаевна

ORCID: 0000-0003-3392-1431

кандидат физико-математических наук

доцент; кафедра математического и аппаратного обеспечения информационных систем; Чувашский государственный университет имени И.Н. Ульянова

428015, Россия, республика Чувашия, г. Чебоксары, Московский пр-т, 15, оф. Б-305

✉ tn_pavlova@mail.ru



Митрофанова Татьяна Валерьевна

ORCID: 0000-0002-5750-7991

кандидат физико-математических наук

доцент; кафедра математического и аппаратного обеспечения информационных систем; Чувашский государственный университет имени И.Н. Ульянова

428015, Россия, республика Чувашия, г. Чебоксары, ул. Московский, 15, оф. Б-304

✉ mitrofanova_tv@mail.ru



Смирнова Татьяна Николаевна

ORCID: 0000-0001-6687-9415

кандидат физико-математических наук

доцент; кафедра математического и аппаратного обеспечения информационных систем; Чувашский государственный университет имени И.Н. Ульянова

428015, Россия, республика Чувашия, г. Чебоксары, Московский пр-т, 15, оф. Б-304

✉ smirnova-tanechka@yandex.ru



Христофорова Анастасия Владимировна

ORCID: 0000-0003-3534-8747

кандидат физико-математических наук

доцент; кафедра математического и аппаратного обеспечения информационных систем; Чувашский государственный университет имени И.Н. Ульянова

428015, Россия, республика Чувашия, г. Чебоксары, ул. Московский, 15, оф. Б-304

✉ dlya.nastenki@mail.ru



[Статья из рубрики "Образовательные программные системы"](#)

DOI:

10.7256/2454-0714.2024.4.71438

EDN:

KLWJBM

Дата направления статьи в редакцию:

09-08-2024

Аннотация: В Российской Федерации большое внимание уделяется развитию сквозных цифровых технологий, в том числе и технологиям искусственного интеллекта (ИИ). Указом Президента РФ от 10 октября 2019 г. № 490 «О развитии искусственного интеллекта в Российской Федерации» утверждена Национальная стратегия развития искусственного интеллекта на период до 2030 года. В соответствии с подпунктом (в) пункта 51.5 Стратегии одним из направлений повышения уровня компетенций в области ИИ и уровня информированности граждан о ИИ является развитие навыков использования технологий ИИ у выпускников образовательных организаций высшего образования посредством включения модулей по ИИ в каждую образовательную программу. Объектом исследования данной статьи является применение систем ИИ в ходе проведения лабораторных и практических занятий, а также самостоятельной работы обучающихся по дисциплинам «Системы искусственного интеллекта», «Основы искусственного интеллекта» и аналогичным дисциплинам. Предметом исследования являются способы решения задач аппроксимации функций и классификации данных средствами специализированных платформ Loginom Community и Neural Network Wizard. Подробно рассмотрены способы обучения нейронной сети, способы оценки качества обучения и обучающей выборки нейронной сети. Методология исследования основана на сочетании теоретического и практического подходов с применением методов анализа, сравнения, обобщения, синтеза, классификации, моделирования. Разработаны материалы, в ходе изучения которых студенты должны получить теоретические знания в области аппроксимации функций и классификации данных, ознакомиться с основными концепциями и методами искусственного интеллекта, а также их применением в различных областях обработки данных, закрепить практические навыки работы с нейронными сетями, а также специализированными платформами и инструментами Loginom Community и Neural Network Wizard. Исследование было проведено с обучающимися 1–2 курсов бакалавриата и специалитета УГСН 09.00.00, 10.00.00, 15.00.00 всех форм обучения. При реализации исследования соблюдались принципы последовательного изложения материала и «от простого к сложному». Результаты контрольных мероприятий показали, что на достаточном уровне были сформированы навыки использования технологий искусственного интеллекта у обучающихся направлений подготовки, не связанных с областью искусственного интеллекта.

Ключевые слова:

искусственный интеллект, нейронные сети, аппроксимация функций, классификация данных, обработка данных, обучение моделей, Loginom Community, Neural Network

Wizard, обучающая выборка, регрессия

Введение

В Российской Федерации большое внимание уделяется развитию сквозных цифровых технологий, в том числе и технологиям искусственного интеллекта. Цифровая трансформация представляет собой одну из пяти национальных целей Российской Федерации на период до 2030 года, определенную Указом Президента РФ от 21 июля 2020 г. № 474 «О национальных целях развития Российской Федерации на период до 2030 года». По этой причине Указом Президента РФ от 10 октября 2019 г. № 490 «О развитии искусственного интеллекта в Российской Федерации» утверждена Национальная стратегия развития искусственного интеллекта на период до 2030 года. В рамках реализации Национальной стратегии в 2019 году был разработан и утвержден федеральный проект «Искусственный интеллект», который призван создать условия, чтобы предприятия и граждане использовали продукты и услуги, созданные с применением преимущественно отечественных технологий искусственного интеллекта, обеспечивающих качественно новый уровень эффективности деятельности.

В настоящее время решение различных задач на основе систем искусственного интеллекта является распространенной темой и применяется в различных областях – в информационных технологиях [\[1\]](#), в финансах [\[2\]](#), спорте [\[3\]](#).

Много интересных работ связано с сельским хозяйством. В статье [\[4\]](#) дано описание разработанных моделей для оценки урожайности озимой пшеницы с использованием множественной линейной регрессии и нейронной сети. Вопросы отслеживания поведения сельскохозяйственных животных с помощью нейронных сетей отражены в работе [\[5\]](#), классификации белых грибов с целью автоматизации процесса их сортировки на основе модели нейронной сети – [\[6\]](#).

Помощи и поддержке лицам, страдающим депрессией, посвящено множество работ. Например, в статье [\[7\]](#) дано описание процесса создания эффективной системы автоматического обнаружения депрессии на основе аудио с применением сверточной нейронной сети. Применение глубокого обучения для определения уровня стресса по непрерывным звуковым сигналам в базе данных отражено в работе [\[8\]](#).

Вопросом улучшения генерации диалогов, основанных на знаниях, за счет двухэтапного отбора знаний и сети указателей, ориентированных на выбор знаний, занимались авторы работы [\[9\]](#). Разработке и применению интерактивной информационной системы в управлении персоналом колледжа посвящена статья [\[10\]](#).

Авторы данной статьи акцентировали внимание на подпункте (в) пункта 51.5 Национальной стратегии, в котором отмечено, что основными направлениями повышения уровня компетенций в области искусственного интеллекта и уровня информированности граждан о технологиях искусственного интеллекта являются «развитие навыков использования технологий искусственного интеллекта у выпускников образовательных организаций высшего образования посредством включения модулей по искусственному интеллекту в каждую образовательную программу (с учетом особенностей, связанных с отраслевой принадлежностью и направлениями подготовки)».

Проблема, на решение которой направлено настоящее исследование, заключается в дефиците адаптированных методик, позволяющих преподавать основы применения ИИ

студентам, не обладающим предварительной подготовкой в этой области. Стандартные подходы к обучению искусственному интеллекту и машинному обучению предполагают наличие у обучающихся углубленных знаний в области математики, программирования и моделирования. Однако студенты непрофильных направлений нуждаются в методике, позволяющей осваивать ИИ-инструменты поэтапно и в практико-ориентированном формате, который будет понятен и доступен.

Данная статья посвящена методике проведения лабораторных и практических занятий, а также организации самостоятельной работы обучающихся 1–2 курсов бакалавриата и специалитета по направлениям подготовки, не связанных непосредственно с ИИ [\[11\]](#).

Научная новизна заключается в разработке методики использования доступных платформ Neural Network Wizard и Loginom Community для решения задач аппроксимации и классификации. Особенность методики – адаптация подхода для обучения студентов, чьи направления подготовки не связаны с искусственным интеллектом. Это способствует развитию навыков работы с искусственным интеллектом, что важно в условиях цифровой трансформации и возросшего внимания к сквозным цифровым технологиям. Предложенные задания позволяют студентам, изучающим фундаментальные и прикладные науки, освоить применение методов искусственного интеллекта для анализа данных. Примеры использования включают такие задачи, как прогнозирование, анализ данных в бизнесе и экономике, оценка показателей эффективности и исследовательские работы, связанные с обработкой больших массивов данных. Предложенный подход позволяет проводить обучение ИИ в формате практических лабораторных работ и самостоятельного освоения принципов построения и применения моделей для классификации и аппроксимации, что способствует расширению профессиональных навыков и отвечает вызовам цифровой экономики. В рамках исследования были выработаны: цели обучения на каждом этапе, принципы взаимодействия с платформами и методические рекомендации, которые в совокупности обеспечивают логичную, поэтапную систему освоения ИИ-инструментов.

Основной целью методики является обучение студентов основам применения нейронных сетей и алгоритмов машинного обучения на примерах конкретных задач: аппроксимации (на платформе Neural Network Wizard) и классификации (на Loginom Community). Задачи включают:

- овладение навыками настройки и обучения нейронных сетей;
- закрепление практических знаний на основе выполнения типовых задач;
- развитие навыков анализа данных и оценки точности моделей.

Ожидаемые результаты обучения включают:

1. Освоение навыков построения нейронных сетей. Студенты смогут создавать и настраивать простые нейронные сети для задач аппроксимации и классификации, используя доступные платформы.
2. Приобретение опыта работы с ИИ-инструментами. Студенты будут способны подготовить данные для анализа, выполнить настройку параметров моделей и провести интерпретацию результатов.
3. Умение анализировать и корректировать модели. На основе анализа ошибок, студенты научатся корректировать параметры моделей, добавлять обучающие данные и улучшать точность прогнозов.

4. Применение знаний для решения прикладных задач. Студенты смогут применять базовые принципы машинного обучения для решения конкретных прикладных задач в их будущей профессиональной деятельности, таких как прогнозирование показателей или анализ экономических данных.

Описание методики использования Neural Network Wizard для аппроксимации функций

Аппроксимация функций является базовым примером использования ИИ в различных задачах. Методика предполагает, что студенты создают простую архитектуру нейронной сети, направленную на аппроксимацию функций с заранее подготовленной обучающей выборкой. В лабораторных условиях обучающая выборка формируется студентами для примера вычисления произведения двух множителей.

Процесс обучения представлен следующими этапами:

1. Подготовка данных. Создание текстового файла с тройками чисел (два множителя и их произведение).
2. Настройка нейронной сети. На платформе Neural Network Wizard студенты строят архитектуру сети, состоящую из одного или нескольких слоев.
3. Процесс обучения и интерпретация результатов. Выполняется обучение сети на примере произведения чисел, анализируется точность вычислений и способы её улучшения (дополнительное обучение, увеличение объема выборки, изменение параметров сети).

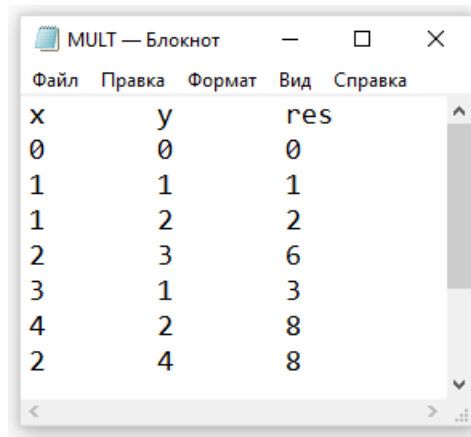
Аппроксимация функций с использованием нейронных сетей

Аппроксимация функций является распространенным случаем применения искусственных нейронных сетей. В работе [\[12\]](#) доказана теорема, в соответствии с которой с помощью линейных операций и каскадного соединения можно из произвольного нелинейного элемента получить устройство, вычисляющее любую непрерывную функцию с некоторой наперед заданной точностью.

Изучение материала предлагается начинать в Neural Network Wizard – среде визуального моделирования архитектуры сверточной нейронной сети и ее обучения.

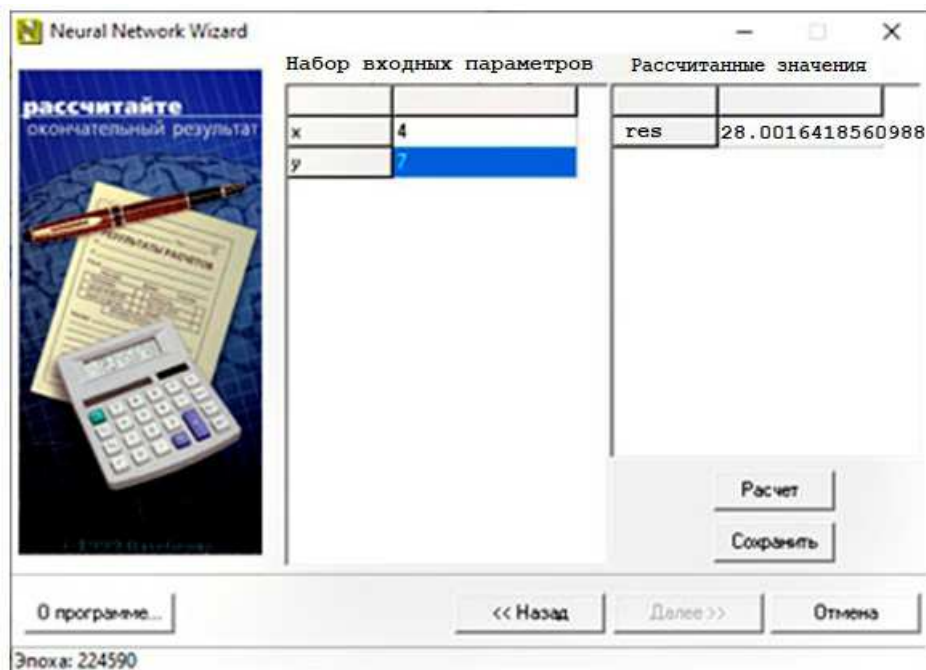
Применение Neural Network Wizard отражено, например, в работах для краткосрочного прогнозирования погоды [\[13\]](#), моделирования цен на легковые автомобили [\[14\]](#), построения моделей рынка смартфонов [\[15\]](#), прогнозирования некоторых котировок валют на ранке Forex [\[16\]](#).

Вначале рассматривается пример аппроксимации функции, предназначенной для вычисления произведения двух множителей. В среду Neural Network Wizard загружается файл формата txt с обучающей выборкой, содержащей тройки чисел – два множителя и произведение (рис. 1). Будем обучать нейронную сеть только на этой выборке. При подстановке пар чисел, не входящих в данное множество, ошибка может быть существенной.



x	y	res
0	0	0
1	1	1
1	2	2
2	3	6
3	1	3
4	2	8
2	4	8

Рис. 1. Файл с обучающей выборкой



Набор входных параметров		Рассчитанные значения	
x	4	res	28.0016418560988
y	7		

0 прогнано... << Назад Далее >> Отмена

Эпоха: 224590

Рис. 2. Результат умножения обученной сетью

Как видно из рис. 2, точность расчетов довольно высокая – отклонение от правильного ответа меньше одного процента.

Способы увеличения точности расчетов:

- 1) вернуться на один шаг назад и обучить сеть дополнительно;
- 2) изменить параметры обучения используемой нейросети;
- 3) увеличить обучающую выборку.

В качестве заданий требуется построить сети с одним, двумя, тремя нейронами на единственном внутреннем слое и обучить их и оценить точность расчетов.

Описание методики использования Loginom Community для задач классификации

Loginom Community применяется для решения задач классификации на основе анализа данных. Студентам предлагается задача определения рентабельности предприятий по историческим данным. Исходные данные включают значения двух показателей, разделяющих предприятия на рентабельные и нерентабельные группы.

Этапы выполнения задания:

- 1.Создание проекта. Студенты загружают исходные данные и настраивают компонент «Нейронная сеть (регрессия)».
- 2.Обучение сети. В настройках указывается количество рестартов (до 1000), проводится настройка параметров подбора, что позволяет автоматизировать процесс обучения.
- 3.Визуализация и анализ результатов. Студенты используют визуализаторы для оценки качества прогноза нейронной сети, проводят анализ ошибок и формируют выводы по точности классификации.

Применение платформы Loginom Community для аппроксимации функций

Платформа Loginom (разработчик – компания Loginom Company) внесена в реестр российских программ и предназначена для проведения расширенной аналитики бизнес-пользователями. В учебных целях рассмотрена настольная редакция для некоммерческого использования Loginom Community, которую можно скачать бесплатно. Visual builder позволяет настраивать весь процесс анализа: интеграцию, подготовку данных, моделирование и визуализацию. В результате сократится время между проверкой гипотез и созданием рабочего бизнес-процесса.

Платформа Loginom Community применяется для решения разнообразных задач. Например, обработки больших массивов данных при решении транспортных задач [\[17\]](#), анализа товарного ассортимента [\[18\]](#), разработки модели социального рейтинга [\[19\]](#), анализа клиентской базы магазина [\[20\]](#).

Для обучения нейронной сети будем использовать исходную обучающую выборку (рис. 1). Добавим в проект компонент «Нейронная сеть (регрессия)» (рис. 3). Укажем количество рестартов обучения нейронной сети, равное 1000. Обратим внимание на параметры настроек окна «Настройка автоматического подбора параметров Нейросети». При увеличении сложности датасета придется ими воспользоваться для поиска оптимальной нейросети и ее настроек. Далее необходимо обучить нейронную сеть, воспользовавшись пунктом контекстного меню «Переобучить узел». Процесс обучения может быть длительным на компьютерах с низкой производительностью.

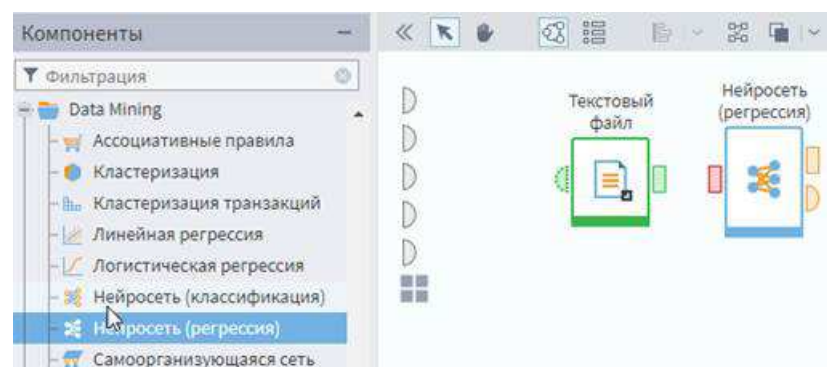


Рис. 3. Добавление компонента «Нейронная сеть (регрессия)»

12 res Прогноз	12 x	12 y	12 res
0	0	0	0
1	1	1	1
2	1	2	2
6	2	3	6
3	3	1	3
8	4	2	8
8	2	4	8
15	5	3	15
12	6	2	12
7	1	7	7
46	5	8	40

Рис. 4. Анализ результатов

Далее необходимо проанализировать эффективность обучения компонента «Нейронная сеть (регрессия)» с применением визуализатора. Имеется возможность добавления до пяти различных визуализаторов выхода нейронной сети («Диаграмма», «Качество данных», «Куб», «Статистика», «Таблица»), на основании которых можно провести анализ качества обучения нейронной сети.

Был выбран визуализатор «Таблица», который отображает исходную обучающую выборку (столбцы x , y , res), а также прогноз нейронной сети (столбец $rez|Прогноз$). Анализ показал, что более 90% прогнозов верные, однако имеются и ошибочные (рис. 4). Такой результат свидетельствует либо об ошибках в обучающей выборке, либо о недостаточной обученности нейронной сети. Улучшить качество работы нейронной сети можно, добавив в обучающую выборку большее количество элементов, а также удалив строки, содержащие неверные значения обучающей выборки.

Аппроксимация двух функций с использованием одной нейронной сети

Рассмотрим пример обучения для выполнения двух операций – сложения и умножения чисел x и y на отрезке [\[0; 10\]](#). Так как в обучающей выборке планируется использование двух различных формул, то в отличие от предыдущей задачи необходимо рассмотреть не две, а три входных переменных (x , y , знак операции). С помощью средств программирования (рис. 5) заполним таблицу Excel, содержащую обучающую выборку (рис. 6).

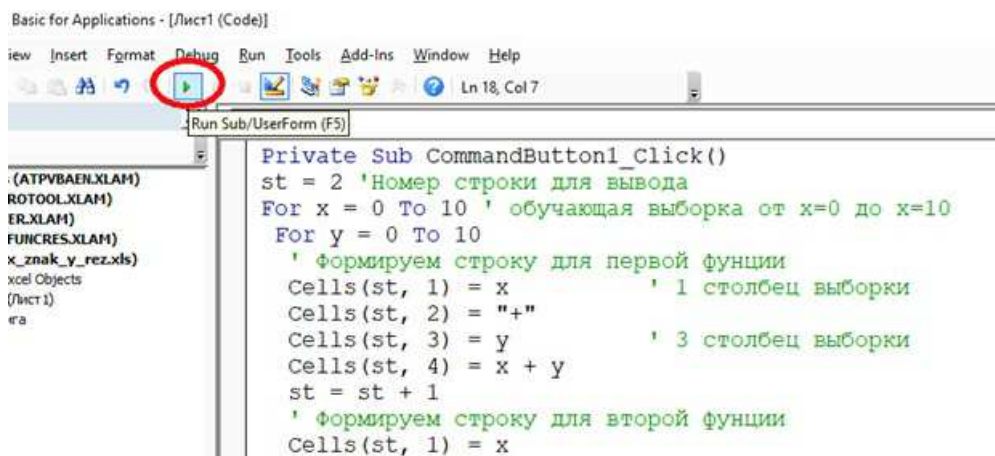


Рис. 5. Фрагмент кода программы

	A	B	C	D
76	3	+	4	7
77	3	*	4	12
78	3	+	5	8
79	3	*	5	15
80	3	+	6	9
81	3	*	6	18

Рис. 6. Фрагмент обучающей выборки

Создадим новый проект в Loginom Community. Соединим выход компонента «Excel файл» с компонентом «Нейронная сеть (регрессия)». Выполним настройки нейросети согласно рис. 7. Параметры разбиения множества: обучающее – 80%, тестовое – 20%. Количество рестартов – 5000. Далее проверим работу нейросети на произвольной выборке (рис. 8).

Настройка входных столбцов

Фильтрация			
Метка	Имя	Вид данных	Назначение
12 x	x	Непрерывный	Входное
ab znak	znak	Дискретный	Входное
12 y	y	Непрерывный	Входное
12 rez	rez	Непрерывный	Выходное

Рис. 7. Фрагмент окна настроек

Loginom Community

Таблица						
Пакеты Пример3 Мод						
#	12 rez Прогноз	12 x	ab znak	12 y	12 rez	
206	27	9	*	3	27	
207	13	9	+	4	13	
208	36	9	*	4	36	
209	14	9	+	5	14	
210	45	9	*	5	45	
211	15	9	+	6	15	

Рис. 8. Фрагмент визуализатора

Такую же задачу требуется решить средствами Neural Network Wizard.

В качестве самостоятельной работы предлагается решить следующие задания средствами программных продуктов Loginom Community и Neural Network Wizard:

1. Доработать проект для сложения и умножения двух вещественных чисел на отрезке [\[0; 10\]](#) (либо на произвольном отрезке).
2. Создать и обучить нейронную сеть для выполнения расчетов по четырем формулам (рис. 9). Для создания обучающей выборки использовать интервал значений x, y, z на отрезке [\[0; 10\]](#). Погрешность расчетов должна быть не более 1–2%.

№	Формула 1	Формула 2	Формула 3	Формула 4
1	$x+y+z$	$x+y-z$	$x-y-z$	$x+y-z$
2	$x+y*z$	$x-y+z$	$x-y-z$	$x-y+z$
3	$x+y+z$	$x*y+z$	$x-y-z$	$x*y+z$
4	$x+y+z$	$x+y*z$	$x-y-z$	$x+y*z$
5	$x+y+z$	$x*y-z$	$x-y-z$	$x*y-z$
6	$x+y+z$	$x-y*z$	$x-y-z$	$x-y*z$

Рис. 9. Фрагмент таблицы с вариантами заданий

Классификационный анализ с использованием нейронных сетей

Изучение данного вопроса отведено на самостоятельную работу, в результате выполнения которой студенты должны научиться использовать нейронные сети для решения классификационных задач.

Приведем один из вариантов формулировки задания. По результатам предыдущего года известны значения двух показателей и выявлены две группы предприятий: группа А – рентабельные, группа В – нерентабельные. Несколько предприятий выполнили прогноз своего развития на текущий год и представили его в аналогичной форме (два показателя, рис. 10). Используя значения прошлого года в качестве обучающей выборки, выполнить классификацию предприятий по признаку рентабельности средствами программных продуктов Loginom Community и Neural Network Wizard.

Рентабельные		Нерентабельные		Прогноз	
2059,091	140,3525	1921,818	254,7346	2745,455	254,4877
1990,454	143,0035	1166,818	255,0123	1784,545	226,1412
823,6364	50,20237	3431,818	288,8432	2196,364	313,4629
274,5456	57,46803	2882,727	293,178	1510	56,25671
1098,182	101,6007	411,8182	182,757	2127,727	179,7001
1029,545	103,1735	960,909	187,9823	2608,182	105,8285
1921,818	108,9705	480,4545	189,4364		
1372,727	110,243	137,2726	207,4699		

Рис. 10. Фрагмент таблицы с исходными данными

Заключение

Исследование было проведено с обучающимися 1–2 курсов бакалавриата и специалитета УГСН 09.00.00, 10.00.00, 15.00.00. При реализации исследования соблюдались принципы последовательного изложения материала и «от простого к сложному». Результаты контрольных мероприятий показали, что на достаточном уровне были сформированы навыки использования технологий искусственного интеллекта у обучающихся направлений подготовки, не связанных с областью искусственного интеллекта.

Авторская методика предоставляет студентам возможность изучить базовые принципы работы с нейронными сетями и инструментами ИИ, адаптированными под образовательные нужды. В отличие от существующих подходов, предложенная методика:

1. Ориентирована на непрофильные специальности и рассчитана на студентов, не обладающих глубокими знаниями в математике и программировании.
2. Включает поэтапное обучение с переходом от простых к более сложным задачам, что снижает порог входа и позволяет развивать навыки работы с данными в удобном темпе.

3.Предлагает практико-ориентированные задания с акцентом на конкретные задачи аппроксимации и классификации, что даёт возможность студентам освоить применимость методов ИИ на реальных примерах.

Таким образом, авторская методика закрывает пробел в образовательной практике, предоставляя преподавателям и студентам простой и эффективный инструмент для начального освоения методов анализа данных и работы с ИИ, что имеет прикладное значение для студентов широкого спектра направлений подготовки.

Полученные результаты могут быть использованы для построения практико-ориентированных учебных курсов и задач в различных областях, не связанных напрямую с искусственным интеллектом, в которых необходимо знание базовых методов анализа данных и алгоритмов искусственного интеллекта.

Предложенный подход позволяет проводить обучение ИИ в формате практических лабораторных работ и самостоятельного освоения принципов построения и применения моделей для классификации и аппроксимации, что способствует расширению профессиональных навыков и отвечает вызовам цифровой экономики. В рамках исследования были выработаны: цели обучения на каждом этапе, принципы взаимодействия с платформами и методические рекомендации, которые в совокупности обеспечивают логичную, поэтапную систему освоения ИИ-инструментов.

Основной целью методики является обучение студентов основам применения нейронных сетей и алгоритмов машинного обучения на примерах конкретных задач: аппроксимации (на платформе Neural Network Wizard) и классификации (на Loginom Community). Задачи включают:

- овладение навыками настройки и обучения нейронных сетей;
- закрепление практических знаний на основе выполнения типовых задач;
- развитие навыков анализа данных и оценки точности моделей.

Ожидаемые результаты обучения включают:

1. Освоение навыков построения нейронных сетей. Студенты смогут создавать и настраивать простые нейронные сети для задач аппроксимации и классификации, используя доступные платформы.
2. Приобретение опыта работы с ИИ-инструментами. Студенты будут способны подготовить данные для анализа, выполнить настройку параметров моделей и провести интерпретацию результатов.
3. Умение анализировать и корректировать модели. На основе анализа ошибок, студенты научатся корректировать параметры моделей, добавлять обучающие данные и улучшать точность прогнозов.
4. Применение знаний для решения прикладных задач. Студенты смогут применять базовые принципы машинного обучения для решения конкретных прикладных задач в их будущей профессиональной деятельности, таких как прогнозирование показателей или анализ экономических данных.

Библиография

1. Компьютерное зрение в теории и на практике / В. А. Артемьев, Т. Н. Копышева, Т. В. Митрофанова [и др.]. // Инновации в информационных технологиях, машиностроении и автотранспорте : Сборник материалов V Международной научно-практической конференции, Кемерово, 19–20 октября 2021 года / Редколлегия: Д.М. Дубинкин (отв. ред.) [и др.]. Кемерово: Кузбасский государственный технический университет имени

- Т.Ф. Горбачева, 2021. С. 47–50. EDN UQSPNB.
2. Лаврентьев, Л. Ф. Финансовое прогнозирование на основе аппарата нейронных сетей / Л. Ф. Лаврентьев, В. П. Филиппов // Вестник Российского университета кооперации. 2014. № 2(16). С. 122–127. EDN QNPSGO.
3. Особенности прогнозирования спортивных событий на основе использования аппарата нейронных сетей / М. С. Портнов, А. В. Речнов, В. П. Филиппов [и др.]. // Вестник Российского университета кооперации. 2019. № 2(36). С. 76–79. EDN KIZOXH.
4. Wheat Yield Estimation Based on Unmanned Aerial Vehicle Multispectral Images and Texture Feature Indices / Y. Kang, Ya. Wang, Ya. Fan [et al.] // Agriculture. 2024. Vol. 14, No. 2. P. 167. DOI: 10.3390/agriculture14020167. EDN PRFCIE.
5. Vayssade, J. A. Wizard: Unsupervised goats tracking algorithm / J. A. Vayssade, X. Godard, M. Bonneau // Computers and Electronics in Agriculture. 2023. Vol. 209. P. 107831. DOI: 10.1016/j.compag.2023.107831. EDN TIHIOK.
6. Кафиев, И. Р. К вопросу автоматизации процесса сортировки белых грибов с использованием нейронных сетей / И. Р. Кафиев, П. С. Романов, И. П. Романова // Вестник НГИЭИ. 2024. № 4(155). С. 34–49. DOI: 10.24412/2227-9407-2024-4-34-49. EDN DENZQG.
7. Audio based depression detection using Convolutional Autoencoder / S. Sardari, B. Nakisa, M. N. Rastgoo, P. Eklund // Expert Systems with Applications. 2022. Vol. 189. P. 116076. DOI: 10.1016/j.eswa.2021.116076. EDN JXRGN.
8. Faye, G. Localized states in an unbounded neural field equation with smooth firing rate function: a multi-parameter analysis / G. Faye, Ja. Rankin, P. Chossat // Journal of Mathematical Biology. 2013. Vol. 66, No. 6. P. 1303–1338. DOI: 10.1007/s00285-012-0532-y. EDN GPURTX.
9. Improving knowledge-based dialogue generation through two-stage knowledge selection and knowledge selection-guided pointer network / M. Liu, P. Zhao, J. Liu [et al.] // Journal of Intelligent Information Systems. 2022. Vol. 59, No. 3. P. 591–611. DOI: 10.1007/s10844-022-00709-5. EDN ULVRYK.
10. Minsheng, L. Application of interactive information system in college personnel management by using BP neural network algorithm / L. Minsheng // Soft Computing-A Fusion of Foundations, Methodologies and Applications. 2023. DOI: 10.1007/s00500-023-08617-8. EDN CTVCYU.
11. Применение систем искусственного интеллекта для обработки данных : практикум / М. С. Портнов, А. В. Речнов, Т. Н. Смирнова [и др.]. Чебоксары: Изд-во Чу-ваш. ун-та, 2023. 52 с.
12. Горбань, А. Н. Обобщенная аппроксимационная теорема и вычислительные возможности нейронных сетей / А. Н. Горбань // Сибирский журнал вычислительной математики. 1998. Т. 1, № 1. С. 11–24. EDN PRZOML.
13. Прогнозирование среднесуточной температуры воздуха в Бишкеке с помощью нейронных сетей в среде Neural Network Wizard / А. А. Насиров, Э. М. Имамалиев, Т. Т. Талантбеков, А. Ж. Андакулов // Наука и инновационные технологии. 2022. № 2(23). С. 138–143. DOI: 10.33942/sititpr202265. EDN OVGMTA.
14. Ступников, А. В. Прогнозирование цены легковых автомобилей с помощью нейронных сетей в среде Neural Network Wizard / А. В. Ступников, Р. И. Баженов // Современная техника и технологии. 2015. № 7(47). С. 3–10. EDN UHWDBB.
15. Козич, П. А. Применение регрессионного анализа и нейронных сетей для построения моделей рынка смартфонов Honor / П. А. Козич, Р. И. Баженов // Постулат. 2019. № 1–1(39). С. 23. EDN ZAHJUT.
16. Клинский, С. Д. Прогнозирование котировок валют на финансовой бирже / С. Д. Клинский // Постулат. 2019. № 12(50). С. 9. EDN TAFBLL.

17. Маркевич, Д. В. Интеграция систем бизнес-аналитики с системами управления базами данных на транспорте / Д. В. Маркевич, В. В. Харланова, А. Д. Хомоненко // Научные технологии в космических исследованиях Земли. 2023. Т. 15, № 2. С. 41-48. DOI: 10.36724/2409-5419-2023-15-2-41-48. EDN XAPPBD.
18. Евдокимова, С. А. Анализ товарного ассортимента запасных частей дилерского предприятия автомобильного сервиса с помощью алгоритма FP-Growth / С. А. Евдокимова, К. В. Фролов, А. И. Новиков // Моделирование систем и процессов. 2022. Т. 15, № 4. С. 24-33. – DOI: 10.12737/2219-0767-2022-15-4-24-33. EDN JCNGHB.
19. Попова, С. А. Разработка модели социального рейтинга и ее нейросетевая реализация в Loginom Community / С. А. Попова, Д. Б. Владимирова // Устойчивое развитие общества: новые научные подходы и исследования : Сборник научных трудов по материалам V Международной научно-практической конференции, Москва, 10 апреля 2024 года. – Москва: Центр развития образования и науки, 2024. С. 151-159. EDN TSVPHX.
20. Евдокимова, С. А. Применение алгоритмов кластеризации для анализа клиентской базы магазина / С. А. Евдокимова, А. В. Журавлев, Т. П. Новикова // Моделирование систем и процессов. 2021. Т. 14, № 2. С. 4-12. DOI: 10.12737/2219-0767-2021-14-2-4-12. EDN DXGWQN

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Рецензируемая статья посвящена применению систем искусственного интеллекта для обработки данных в учебном процессе, в ней рассматривается методика проведения лабораторных и практических занятий, а также организации самостоятельной работы обучающихся 1–2 курсов бакалавриата и специалитета по направлениям подготовки, не связанных непосредственно с искусственным интеллектом.

Методология исследования базируется на демонстрации возможностей среды визуального моделирования Neural Network Wizard и платформы Loginom Community в аппроксимации функций с использованием нейронных сетей и решении задач классификации.

Актуальность работы авторы связывают с тем, что В Российской Федерации большое внимание уделяется развитию сквозных цифровых технологий, в том числе и технологиям искусственного интеллекта, а принятые на государственном уровне программные документы предусматривают развитие навыков использования технологий искусственного интеллекта у выпускников образовательных организаций высшего образования посредством включения модулей по искусственному интеллекту в каждую образовательную программу.

Научная новизна рецензируемого исследования, к сожалению, авторами в публикации четко не сформулирована, по мнению рецензента, к новым результатам можно отнести изложение методики использования среды визуального моделирования Neural Network Wizard и платформы Loginom Community для решения задач с применением методов искусственного интеллекта.

Структурно в публикации выделены следующие разделы: Введение, Аппроксимация функций с использованием нейронных сетей, Применение платформы Loginom Community для аппроксимации функций, Аппроксимация двух функций с использованием одной нейронной сети, Классификационный анализ с использованием нейронных сетей, Заключение, Благодарность и Библиография.

Представляется, что отраженные в публикации учебные примеры использования искусственных нейронных систем для решения задач классификации и аппроксимации функций с использованием популярных и общедоступных инструментов могут быть полезны широкому кругу читателей на начальном этапе ознакомления с возможностями методов искусственного интеллекта.

Библиографический список включает 20 источников – публикации отечественных и зарубежных авторов по теме статьи, на которые в тексте имеются адресные ссылки, подтверждающие наличие апелляции к оппонентам.

Из замечаний стоит отметить, во-первых, наименования разделов статьи охватывают лишь часть элементов системы интеллекта – заголовок шире, чем отраженные в публикации материалы; чтобы у читателей не возникло ложное представление об искусственном интеллекте авторам предлагается либо уточнить название публикации, либо расширить освещение систем искусственного интеллекта, которые не сводятся только лишь к аппроксимации функций с использованием нейронных сетей и решению задач классификации; во-вторых, хотелось бы увидеть более четкие формулировки полученного приращения научного знания и возможных сфер практического применения полученных результатов исследования.

Статья соответствует направлению журнала «Программные системы и вычислительные методы», содержит ознакомительный материал о применении систем искусственного интеллекта для обработки данных в учебном процессе, может вызвать интерес у читателей, но формулировки научной новизны и практической значимости нуждается в уточнении, материал необходимо доработать в соответствии с высказанными замечаниями.

Результаты процедуры повторного рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Представленная статья на тему «Применение платформ Neural Network Wizard и Loginom Community для решения задач аппроксимации и классификации в образовательном процессе» соответствует тематике журнала «ПРОГРАММНЫЕ СИСТЕМЫ И ВЫЧИСЛИТЕЛЬНЫЕ МЕТОДЫ» и посвящена актуальному направлению «развитие навыков использования технологий искусственного интеллекта у выпускников образовательных организаций высшего образования посредством включения модулей по искусственному интеллекту в каждую образовательную программу».

Актуальность статьи не вызывает сомнений, так как в последние годы в России большое внимание уделяется развитию сквозных цифровых технологий, в том числе и технологиям искусственного интеллекта. В соответствии с Национальной стратегией развития искусственного интеллекта на период до 2030 года предприятия и граждане должны быть обеспечены условиями использования продуктов и услуг, созданных с применением преимущественно отечественных технологий искусственного интеллекта, обеспечивающих качественно новый уровень эффективности деятельности. Авторы акцентировали внимание на подпункте (в) пункта 51.5 Национальной стратегии, в котором отмечено, что необходимо повышать уровень компетенций в области искусственного интеллекта и уровня информированности граждан о технологиях искусственного интеллекта, в том числе развивать навыки использования технологий искусственного интеллекта у выпускников образовательных организаций высшего образования.

Статья достаточно структурирована - в наличии введение, заключение, внутреннее

членение основной части.

В статье представлены примеры применения платформ Neural Network Wizard и Loginom Community на лабораторных и практических занятиях, и организации самостоятельной работы бакалавров и специалистов по направлениям подготовки, не связанных непосредственно с искусственным интеллектом. Авторы статьи провели аналитический обзор российской и зарубежной актуальной литературы.

К недостаткам можно отнести: качество языка не отвечает требованиям научной статьи. Статья очень средняя по содержанию, смыслу, отсутствуют выводы и т.п. (с точки зрения ВАКовских требований). Чувствуется «студенческая рука». Также, практическая значимость статьи на среднем уровне. Из содержания статьи слабо прослеживается научная новизна.

Авторами заявлена разработка методики использования доступных платформ Neural Network Wizard и Loginom Community для решения задач аппроксимации и классификации, однако в статье не раскрыто содержание методики использования данных платформ (отсутствуют цели обучения; принципы обучения; результаты обучения).

Рекомендуется более четко обозначить проблему исследования и авторский вклад. Рекомендуемый объем статьи от 12000 знаков, однако представленная статья не отвечает данным требованиям.

Статья «Применение платформ Neural Network Wizard и Loginom Community для решения задач аппроксимации и классификации в образовательном процессе» требует доработки по указанным выше замечаниям. После внесения поправок рекомендуется к повторному рассмотрению редакцией рецензируемого научного журнала.

Результаты процедуры окончательного рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Предметом исследования данной статьи является разработка методики преподавания основ искусственного интеллекта (ИИ) и машинного обучения (МО) для студентов, не обладающих предварительными знаниями в данной области. Основной фокус направлен на использование платформ Neural Network Wizard и Loginom Community для решения задач аппроксимации и классификации в образовательном процессе.

Методология, предложенная авторами, основана на проведении лабораторных занятий, практических упражнений и организации самостоятельной работы студентов. В статье описывается поэтапное обучение, которое включает настройку и обучение простых нейронных сетей, анализ данных и оценку точности моделей. Уникальность подхода состоит в его доступности для студентов непрофильных направлений, позволяя освоить сложные концепции ИИ и МО без необходимости углубленных знаний в математике или программировании.

Актуальность данного исследования обусловлена значимостью цифровой трансформации и внедрением технологий ИИ в различные отрасли. Программа обучения, направленная на студентов, не имеющих профильной подготовки, позволяет расширить доступ к знаниям о современных цифровых технологиях и адаптирует обучение под актуальные потребности рынка. Исследование важно и для образовательных учреждений, заинтересованных в подготовке специалистов с базовыми навыками работы с ИИ, что необходимо в условиях развития национальной стратегии по искусственному интеллекту. Научная новизна исследования заключается в адаптации платформ Neural Network Wizard и Loginom Community для обучения ИИ и МО студентов, чьи направления

обучения не связаны с данной областью. Авторы разработали комплекс заданий, подходящих для использования в учебной среде, что позволяет студентам последовательно освоить инструменты анализа данных, необходимые для решения прикладных задач. Данный подход уникален и способен служить полезным дополнением к учебным программам технических и гуманитарных вузов.

Статья написана в доступном для понимания стиле, структура работы логична и последовательна. Каждый раздел посвящен отдельному аспекту методики, что позволяет читателю легко ориентироваться в материалах. Содержание статей носит систематизированный характер, представленные данные иллюстрируются примерами и графическими материалами, что способствует лучшему пониманию изложенного материала. Авторы охватывают как теоретические аспекты, так и практические задачи, предлагая готовые решения для применения в образовательной практике.

Выводы работы демонстрируют эффективность предложенной методики и указывают на успешное усвоение студентами непрофильных направлений базовых принципов ИИ и МО. Результаты исследования могут представлять интерес для преподавателей, специалистов в области образования, а также для тех, кто разрабатывает учебные программы по внедрению цифровых технологий.

Для дальнейшего развития данной работы рекомендуется уделить внимание нескольким направлениям, которые могут повысить эффективность предложенной методики и ее практическую ценность. Во-первых, следует рассмотреть возможность более глубокого анализа и расширения спектра задач, решаемых с использованием платформ Neural Network Wizard и Loginom Community. Включение примеров, связанных с более сложными сценариями, например, анализа временных рядов или многомерных данных, позволит студентам лучше понять универсальность и гибкость применения нейронных сетей в различных областях.

Во-вторых, полезным было бы дополнить методику заданиями, ориентированными на междисциплинарное применение технологий искусственного интеллекта. Это может включать разработку проектов, которые интегрируют знания студентов из других предметных областей, таких как биология, экономика, психология, и требуют применения ИИ для решения практических проблем в этих дисциплинах. Такой подход не только повысит мотивацию студентов, но и продемонстрирует реальные возможности использования ИИ за пределами технических направлений.

Третьим аспектом, который заслуживает внимания, является развитие системы обратной связи, позволяющей преподавателям и студентам отслеживать прогресс обучения и выявлять области, требующие улучшений. Введение инструментов автоматизированного тестирования и адаптивного обучения позволит оценивать уровень усвоения материала каждым студентом и персонализировать процесс обучения, что особенно важно в контексте обучения непрофильных студентов.

Наконец, перспективным направлением является интеграция предложенной методики с другими цифровыми платформами и инструментами анализа данных, такими как Python или R, что откроет студентам доступ к более широкому спектру возможностей и сделает их навыки более востребованными на современном рынке труда.

Программные системы и вычислительные методы

Правильная ссылка на статью:

Ишанхонов А.Ю., Пшиченко Д.В., Можаровский Е.А., Алуев А.С. Роль больших языковых моделей в интегрированных средах разработки нового поколения // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.72022 EDN: KMTOBG URL: https://nbpublish.com/library_read_article.php?id=72022

Роль больших языковых моделей в интегрированных средах разработки нового поколения

Ишанхонов Азизхон Юнусхон

ORCID: 0009-0009-8934-6289

магистр; кафедра металловедения цветных металлов; Национальный исследовательский технологический университет "МИСИС"

119049, Россия, г. Москва, Ленинский пр-т, 4, стр. 1

✉ m180119@edu.misis.ru



Пшиченко Дмитрий Викторович

ORCID: 0009-0006-8866-8057

независимый исследователь

119049, Россия, г. Москва, ул. Шаболовка, 26-28

✉ dmitry.pshychenko@rambler.ru



Можаровский Евгений Александрович

ORCID: 0009-0005-9957-1632

независимый исследователь

119991, Россия, г. Москва, ул. Ленинские горы, 1

✉ mozharovsky_ea@rambler.ru



Алуев Андрей Сергеевич

ORCID: 0009-0001-6737-7545

магистр; Уральский Федеральный Университет

620062, Россия, г. Екатеринбург, ул. Мира, 19

✉ aluev_andrei@rambler.ru



[Статья из рубрики "Методы, языки и модели человеко-машинного взаимодействия"](#)

DOI:

10.7256/2454-0714.2024.4.72022

EDN:

KMTOBG

—

дата направления статьи в редакцию:

18-10-2024

Аннотация: В статье рассматривается роль больших языковых моделей (Large Language Models, LLM) в интегрированных средах разработки (Integrated Development Environment, IDE) нового поколения. Изучаются инструменты, такие как GitHub Copilot, IntelliCode и Alice Code Assistant, в контексте их использования в программировании. Авторы исследуют, каким образом LLM позволяют автоматизировать ключевые задачи разработки, включая автодополнение кода, выявление ошибок, рефакторинг и генерацию фрагментов программного кода, и как автоматизация приводит к повышению эффективности разработки и улучшению качества конечного продукта. Особое внимание уделяется влиянию использования LLM на когнитивные процессы разработчиков, их способность к решению творческих задач, а также на мотивацию и профессиональные навыки. Так же обсуждаются этические аспекты внедрения LLM. Обзор существующих интегрированных сред разработки, в которых применяются большие языковые модели. Оценивались функциональные возможности LLM для автодополнения кода, генерации фрагментов, выявления и исправления ошибок. Применялись сравнительные методы для оценки эффективности LLM по сравнению с традиционными средствами разработки. Новизна исследования заключается в комплексном анализе применения LLM в современных IDE, а также в выявлении их потенциала для повышения продуктивности разработчиков и улучшения качества программного кода. Сделан вывод о том, что интеграция LLM в IDE позволяет не только ускорить процесс создания кода, но и существенно повысить его качество за счет интеллектуальной поддержки и автоматизации рутинных задач. Однако выявлены и ограничения, связанные, в частности, с когнитивной нагрузкой, этическими вопросами и необходимостью сохранения баланса между автоматизацией и развитием навыков программистов. Авторы отмечают, что для успешной интеграции LLM необходим продуманный и ответственный подход, предполагающий баланс между автоматизацией и сохранением творческого потенциала программистов.

Ключевые слова:

большие языковые модели, интегрированные среды разработки, автоматизация программирования, улучшение кода, искусственный интеллект, автодополнение кода, программные системы, машинное обучение, оптимизация процессов разработки, анализ данных

Введение

Большие языковые модели (Large Language Models, LLM), основанные на архитектуре трансформеров, демонстрируют значительный прогресс в области искусственного интеллекта (artificial intelligence, AI) и обработки естественного языка. Изначально разработанные для решения задач, связанных с текстовой обработкой, таких как машинный перевод, генерация текстов и ответы на вопросы, они быстро нашли широкое применение в различных отраслях, включая программирование и разработку программного обеспечения (ПО). В современных интегрированных средах разработки

(Integrated Developer Environment, IDE) LLM являются важным инструментом, способным существенно автоматизировать процессы, повысить эффективность разработки и улучшить качество кода.

Интеграция LLM в IDE предоставляет возможность автоматизации множества задач, включая автодополнение кода, обнаружение и исправление ошибок, а также генерацию фрагментов ПО. Эти функции позволяют разработчикам сократить время, затрачиваемое на выполнение рутинных операций, снизить количество ошибок и сосредоточиться на решении более сложных и творческих задач. Кроме того, LLM способствуют интеллектуальной поддержке, предоставляя рекомендации по оптимизации и улучшению кода, что может значительно повысить производительность и качество работы.

Однако внедрение LLM в процессы разработки ПО порождает ряд вопросов и вызовов. Одним из ключевых аспектов является влияние таких моделей на когнитивные процессы разработчиков, их способность к обучению и самосовершенствованию, а также на профессиональные компетенции. Возникает необходимость оценки того, как постоянное взаимодействие с LLM сказывается на нагрузке программистов, их мотивации и способности к творческому мышлению. Этические аспекты применения LLM в программировании, включая вопросы безопасности данных, защиты интеллектуальной собственности и возможные риски, связанные с чрезмерной зависимостью от технологий AI, являются особенно актуальными.

Целью данной статьи является анализ роли LLM в IDE нового поколения. Делается акцент на когнитивных и социальных аспектах взаимодействия разработчиков с LLM, а также на оценке их влияния на эффективность программирования, на изменение методов работы и взаимодействие в командах.

Основная часть. Эволюция LLM и их применение в IDE

Первые LLM, созданные в 1990-е гг., основывались на статистических подходах, таких как *n*-граммы и скрытые марковские модели. **N-граммы** – это последовательности из *n* элементов (слов, символов или других единиц), которые встречаются в тексте или данных [1]. В контексте обработки естественного языка (NLP) *n*-граммы используются для моделирования текста и предсказания следующих элементов на основе предыдущих. Например, для предложения «Машинное обучение помогает решать задачи» биграммы (последовательности из двух элементов): «машинное обучение», «обучение помогает», «помогает решать», «решать задачи».

Скрытые марковские модели основаны на предположении, что вероятность появления каждого элемента кода или текста зависит только от нескольких предыдущих элементов, а не от всей последовательности [2]. Такие модели вычисляют вероятности слов в тексте или следующего шага при написании кода на основе наблюдений из тренировочных данных, где оцениваются частоты появления элементов (рис.1).

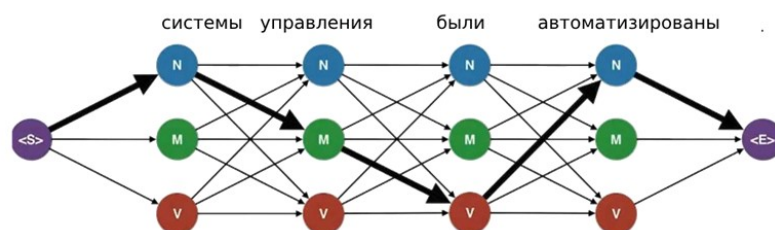


Рисунок 1. Схема марковской модели

Несмотря на свою популярность и широкое применение на ранних этапах разработки интеллектуальных инструментов для IDE, такие технологии, как n-граммы и марковские модели, обладают существенными ограничениями. Эти методы основаны на анализе локального контекста, что не позволяет учитывать долгосрочные зависимости между элементами кода или текста. В IDE это означает, что такие модели могут упускать важные связи между переменными, функциями или блоками кода, особенно в крупных проектах с длинными зависимостями. Игнорирование более глубоких семантических связей ограничивает точность автодополнения, поиска ошибок и анализа кода, так как марковские модели и n-граммы не могут эффективно учитывать глобальную структуру программы.

Существенным этапом в развитии LLM стало появление **Recurrent Neural Network** (RNN) в начале 2000-х гг. [3]. Эти модели способны сохранять контекст предыдущих элементов в коде или тексте на протяжении более длинных последовательностей, что позволяет IDE лучше понимать структуру и логику программ (рис.2).

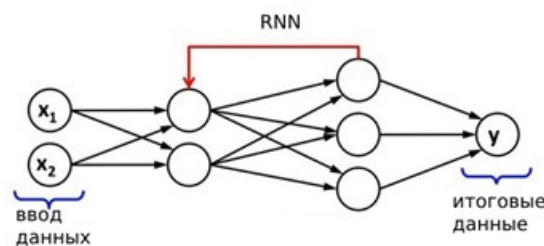


Рисунок 2. Схема RNN

Однако даже более продвинутые варианты RNN, такие как LSTM (The Long Short-Term Memory) и GRU (Gated Recurrent Unit), сталкиваются с проблемой обработки длинных кодов, поскольку в них наблюдается постепенное снижение количества важной информации по мере роста длины последовательностей.

В 2017 году появилась архитектура трансформеров, которая изменила подход к языковому моделированию. **Трансформеры** используют механизм внимания (attention), который позволяет моделям сосредоточиться на значимых частях текста, независимо от того, где они находятся в последовательности (рис.3).

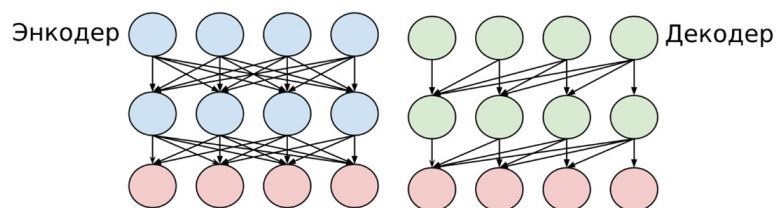


Рисунок 3. Схема архитектуры трансформеров

В IDE блок энкодера играет важную роль в преобразовании исходного кода или текстовых данных в промежуточные представления, которые затем передаются через многослойную архитектуру внимания. Это позволяет IDE анализировать взаимосвязи между различными элементами кода, выявляя ключевые структуры и зависимости, независимо от их местоположения в кодовой базе. Механизм внимания в энкодере обеспечивает взаимодействие между любыми элементами последовательности, будь то переменные, функции или блоки кода, что позволяет эффективно обрабатывать долгосрочные зависимости, такие как связи между определением переменной и ее использованием в других частях программы.

Такой подход значительно улучшает работу IDE по сравнению с предыдущими моделями, такими как RNN и LSTM, которые испытывали трудности с учетом удаленных зависимостей в коде [4]. Это нововведение стало важным шагом в развитии инструментов для автодополнения, анализа ошибок и рефакторинга кода. Блок декодера использует промежуточные представления, созданные энкодером, для генерации новой последовательности кода, например, для автодополнения функции, генерация тестов или предсказания следующего шага разработки.

Модели на основе трансформеров, такие как BERT (Bidirectional Encoder Representations from Transformers) и GPT (Generative Pre-trained Transformer), широко применяются в различных сферах, включая задачи, связанные с программированием. **BERT**, благодаря своей двунаправленной архитектуре, обрабатывает текст, учитывая контекст как слева, так и справа от каждого слова, что делает его особенно полезным в задачах анализа и рефакторинга кода, автодополнения и поиска ошибок [5]. В IDE это позволяет более точно понимать намерения разработчиков и предлагать оптимальные решения для завершения кода или улучшения его структуры. **GPT** с его однонаправленным подходом к генерации, особенно полезен в задачах, связанных с автодополнением и генерацией кода на основе текстовых описаний [6]. В интегрированных средах разработки GPT может использоваться для написания новых фрагментов кода, предсказания следующего шага в программировании, создания шаблонов и автогенерации тестов.

Роль LLM в трансформации современных IDE

Внедрение LLM в современные IDE является значимым этапом эволюции инструментов программирования. LLM обладают способностью анализировать код, предсказывать следующие фрагменты и предлагать исправления, что значительно ускоряет процесс разработки и улучшает его качество. Это не только облегчают рутинные задачи, но и предоставляют разработчикам интеллектуальную поддержку на всех этапах создания ПО (таблица 1).

Таблица 1. Влияние LLM на IDE [7, 8]

Аспект применения	Описание	Преимущества для разработки	Примеры инструментов
Автодополнение кода	Модели прогнозируют следующий фрагмент кода на основе текущего контекста.	Ускорение написания кода, снижение количества синтаксических ошибок.	GitHub Copilot, IntelliCode
Анализ кода в реальном времени	Автоматическая проверка кода на наличие ошибок; предложения по оптимизации во время написания.	Улучшение структуры и качества кода, уменьшение числа ошибок на этапе разработки.	DeepCode, CodeGuru
Обнаружение ошибок и исправление синтаксиса	Модели выявляют логические и синтаксические ошибки с предложением их исправления.	Повышение надежности кода, сокращение времени на отладку.	Tabnine, Kite
Прогнозирование и автоматизация рутинных задач	Автоматизация повторяющихся операций, таких как генерация шаблонного кода или создание тестов.	Снижение когнитивной нагрузки разработчиков, ускорение выполнения стандартных задач.	Codex, Replit

Применение LLM в IDE значительно преобразует процесс программирования, предлагая

новые возможности для автоматизации и интеллектуальной поддержки. Эти технологии не только повышают эффективность разработки за счет ускорения рутинных операций, но и влияют на качество создаваемого ПО, снижая вероятность ошибок на ранних этапах [9]. Внедрение LLM способствует более интуитивному взаимодействию с кодом, позволяя разработчикам сосредоточиться на решении творческих и сложных задач, минимизируя при этом время, затрачиваемое на поиск и исправление ошибок.

Успешная интеграция LLM требует учета возможных ограничений, таких как зависимость от качества обучающих данных и необходимость контроля со стороны разработчиков для предотвращения внедрения неэффективных или небезопасных решений. Важно также учитывать этические аспекты использования моделей, особенно в отношении интеллектуальной собственности и конфиденциальности данных.

Когнитивное взаимодействие разработчиков с LLM

Интеграция LLM в процессы разработки ПО оказывает значительное влияние на когнитивное взаимодействие разработчиков с кодом и инструментами разработки. Одним из ключевых аспектов является **снижение когнитивной нагрузки на разработчика**. За счет автоматизации рутинных задач, таких как автодополнение кода и исправление синтаксических ошибок, LLM позволяют разработчикам сосредотачиваться на более сложных и творческих аспектах программирования.

Кроме того, взаимодействие с LLM **изменяет процесс принятия решений**. Модели предоставляют множество возможных вариантов решения задач, что расширяет выбор разработчика и может ускорить процесс нахождения правильного решения. Это помогает сократить количество усилий, связанных с поиском информации, и делает процесс принятия решений более интуитивным и эффективным.

Важным аспектом когнитивного взаимодействия с LLM является **поддержка творческого процесса**. LLM способны предлагать нестандартные решения и помогать разработчикам выйти за рамки привычных шаблонов. Еще один важный аспект – **адаптация LLM к личному стилю программирования разработчика**. По мере использования модели, LLM могут запоминать стиль и предпочтения конкретного сотрудника, предлагая решения, которые соответствуют его привычкам и практикам. Это позволяет сократить время на выполнение задач и сделать взаимодействие с моделью более интуитивным.

Применение LLM в IDE: анализ современных решений в крупных компаниях

Международные компании активно внедряют LLM в IDE, чтобы автоматизировать процессы программирования, улучшить качество кода и повысить производительность разработчиков. Одним из самых известных инструментов является **GitHub Copilot**, разработанный корпорацией Microsoft на базе модели OpenAI Codex [10]. Он внедрен в популярные IDE, такие как Visual Studio Code, и помогает разработчикам за счет автодополнения кода, генерации целых фрагментов программ на основе текстовых описаний и предложений по улучшению кода. GitHub Copilot обучен на миллиардах строк кода из открытых репозиториях, что позволяет ему предоставлять контекстуально точные подсказки для многих языков программирования. Этот инструмент широко используется в американских компаниях для ускорения разработки и снижения нагрузки на программистов. Например, он на 25 % увеличил скорость разработки платформы по изучению языка Duolingo, а среднее время выполнения проверки кода сократилось на 67%.

Еще один продукт от Microsoft, **IntelliCode**, представляет собой интеллектуальную систему автодополнения кода, интегрированную в Visual Studio и Visual Studio Code. IntelliCode использует AI для анализа кода, предлагая рекомендации на основе лучших практик и кода, используемого в реальных проектах [\[11\]](#). Так, при написании функции на языке Python, IntelliCode может предложить оптимальное использование стандартных библиотек и улучшения для повышения эффективности кода. Рассмотрим следующий пример:

```
def process_data(data):  
  
    result = []  
  
    for item in data:  
  
        result.append(item * 2)  
  
    return result  
  
data = \[1, 2, 3, 4\]  
  
print(process_data(data)), (1)
```

При использовании IntelliCode система может предложить оптимизированный код с использованием встроенных методов Python, таких как `map()` и `lambda`, что ускоряет выполнение программы:

```
def process_data(data):  
  
    return list(map(lambda x: x * 2, data))  
  
data = \[1, 2, 3, 4\]  
  
print(process_data(data)), (2)
```

В отличие от GitHub Copilot, который предлагает широкие рекомендации, IntelliCode больше ориентирован на улучшение качества кода за счет следования стандартам и анализу предыдущих действий разработчика. Этот инструмент активно используется американскими и международными компаниями для повышения эффективности разработки ПО.

Компания «Яндекс», один из лидеров российского IT-рынка, разработала собственные решения на основе LLM для поддержки разработчиков. В рамках своей экосистемы LLM были интегрированы в среду разработки через **Alice Code Assistant**, который помогает программистам за счет автодополнения кода, предложения тестов и автоматического исправления ошибок [\[12\]](#). При написании функции на языке Python для обработки данных, Alice Code Assistant может предложить автодополнение и автоматическую генерацию тестов для функции. Рассмотрим пример:

```
def process_data(data):  
  
    processed_data = []  
  
    for item in data:  
  
        processed_data.append(item.lower())
```

```
return processed_data

data = ["Hello", "World", "Alice"]

print(process_data(data)), (3)
```

Alice Code Assistant может предложить оптимизацию этой функции, используя более эффективные встроенные методы Python, такие как list comprehension, а также предложить сгенерировать тесты для функции:

```
# Оптимизированный код с рекомендацией Alice Code Assistant

def process_data(data):

    return [item.lower() for item in data]

# Автоматически сгенерированный тест

def test_process_data():

    assert process_data(["Hello", "World", "Alice"]) == ["hello", "world", "alice"]

    assert process_data([]) == []

# Запуск теста

test_process_data(), (4)
```

Этот инструмент используется в различных проектах компании Яндекс, в том числе в развитии поисковых и рекламных технологий. Alice Code Assistant обеспечивает поддержку различных языков программирования и активно применяется внутри Яндекса для ускорения разработки новых сервисов и приложений.

Проблемы и вызовы интеграции LLM в интегрированные среды разработки нового поколения

Интеграция LLM в IDE нового поколения приносит множество преимуществ, таких как автоматизация рутинных задач, улучшение качества кода и повышение производительности. Однако этот процесс сопряжен с рядом проблем, включая зависимость от качества данных, недостаточную адаптацию моделей к специфическим проектам, вопросы интеллектуальной собственности и этические риски. Чтобы обеспечить успешное внедрение LLM, необходимо учитывать эти вызовы и разрабатывать эффективные стратегии их решения (таблица 2).

Таблица 2. Проблемы интеграции LLM и способы их решения [\[13, 14\]](#)

Проблема	Описание	Способы решения
Качество предвзятость данных	Модели могут обучаться на данных с ошибками или предвзятостью, что снижает точность рекомендаций.	Постоянное обновление обучающих данных, использование специализированных данных для конкретных задач.
Отсутствие контекста	LLM не всегда могут учитывать уникальные требования конкретных проектов.	Разработка и внедрение моделей, обученных на данных, связанных с конкретной областью или проектом.
Проблемы интеллектуальной	Модели могут использовать данные, защищенные	Применение моделей на основе закрытых корпоративных данных и

собственностью	авторскими правами, или нарушать конфиденциальность.	внедрение строгих механизмов управления доступом.
Зависимость от технологий	Чрезмерное использование LLM может привести к снижению навыков разработчиков.	Ограничение автоматизации в ключевых этапах разработки, стимулирование обучения и улучшения навыков разработчиков.
Этические вопросы безопасность	Использование LLM может создавать риски для безопасности данных.	Разработка и внедрение этических стандартов и протоколов безопасности при использовании LLM.

Анализ проблем внедрения LLM в интегрированные среды разработки показывает, что успешное внедрение этих технологий требует комплексного подхода и внимательного контроля. Хотя LLM могут значительно повысить продуктивность и автоматизировать рутинные задачи, их применение должно сопровождаться адаптацией к специфике проекта и улучшением данных, на которых обучены модели. Это позволит избежать возможных предвзятостей и ошибок в работе моделей. Кроме того, важно обеспечивать баланс между автоматизацией и развитием навыков разработчиков, чтобы не допустить снижения их квалификации.

Этические вопросы также требуют особого внимания. Использование LLM должно осуществляться в рамках строгих стандартов, касающихся безопасности и защиты данных, чтобы предотвратить утечку конфиденциальной информации и нарушение прав на интеллектуальную собственность.

Заключение

Применение LLM в IDE существенно изменяет процессы программирования, способствуя автоматизации рутинных задач и повышению общей производительности. Использование LLM в IDE позволяет разработчикам не только ускорить создание кода, но и улучшить его качество за счет интеллектуальной поддержки и автодополнения. Однако, несмотря на очевидные преимущества, применение таких технологий требует решения ряда этических и технических вопросов. В частности, важно контролировать зависимость от автоматизации, чтобы поддерживать профессиональные навыки разработчиков, а также обеспечивать соблюдение норм безопасности и конфиденциальности данных. Для успешной интеграции LLM необходим продуманный и ответственный подход, предполагающий баланс между автоматизацией и сохранением творческого потенциала программистов.

Библиография

1. Иванов К. Н., Захарова О. И. Обработка естественного языка. Применение языковых моделей // Актуальные проблемы информатики, радиотехники и связи. – 2023. – С. 155-156.
2. Korostin O. Comparative analysis of NLP algorithms for optimizing communications in the maritime industry // Journal of science. Lyon. – 2024. – № 56. – С. 19-22.
3. Qin Z., Yang S., Zhong Y. Hierarchically gated recurrent neural network for sequence modeling // Advances in Neural Information Processing Systems. – 2024. – V. 36.
4. Узких Г. Ю. Применение трансформеров в обработке естественного языка // Вестник науки. – 2024. – Т. 4. – № 8 (77). – С. 186-189.
5. Gweon H., Schonlau M. Automated classification for open-ended questions with BERT // Journal of Survey Statistics and Methodology. – 2024. – V. 12. – № 2. – P. 493-504.
6. Liukko V., Knappe A., Anttila T., Hakala J. ChatGPT as a Full-Stack Web Developer // Generative AI for Effective Software Development. – Cham: Springer Nature Switzerland. –

2024. – P. 197-215.

7. Ponomarev E. Optimizing android application performance: modern methods and practices // Sciences of Europe. – 2024. – № 149. – С. 62-64.

8. Макарьян О. С. Разработка программного обеспечения с использованием искусственного интеллекта // Вестник магистратуры. – 2024. – С. 23.

9. Бобунов А. Ю. Сравнение практик автоматизации тестирования в традиционных банках и финтех-компаниях // Дневник науки. 2024. № 8 [Электронный ресурс]. URL: <http://www.dnevniknauki.ru/images/publications/2024/8/technics/Bobunov.pdf>

10. Koyanagi K., Wang D., Noguchi K., и т. д. Exploring the effect of multiple natural languages on code suggestion using github copilot // 2024 IEEE/ACM 21st International Conference on Mining Software Repositories (MSR). – 2024. – P. 481-486.

11. Oh S., Lee K., Park S., Kim D. Poisoned chatgpt finds work for idle hands: exploring developers' coding practices with insecure suggestions from poisoned ai models // 2024 IEEE Symposium on Security and Privacy (SP). – 2024. – P. 1141-1159.

12. Жикулина К. П., Перфильева Н. В., Мань Л. Цифровой страт парадигмы языка // Вестник Российского университета дружбы народов. Серия: Теория языка. Семиотика. Семантика. – 2024. – Т. 15. – № 2. – С. 364-375.

13. Пекарева, В. В. Семантический анализ дефиниции «информация» в целях систематизации подходов и факторов обеспечения информационной безопасности в условиях цифровизации / В. В. Пекарева, Ю. И. Фроловская // Аграрное и земельное право. – 2024. – № 3(231). – С. 89-92. – DOI: 10.47643/1815-1329_2024_3_89. – EDN PQGQDB

14. Verner D. Integration of artificial intelligence in backend development // Annali d'Italia. – 2024. – № 59. – P. 88-91.

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Предметом исследования в рецензируемой публикации выступают большие языковые модели (LLM), в работе раскрывается их роль в интегрированных средах разработки (IDE) нового поколения.

Методология исследования базируется на изучении и обобщении научных публикаций по рассматриваемой теме и анализе применения больших языковых моделей в практической работе команд ИТ-разработчиков.

Актуальность работы авторы связывают с тем, что большие языковые модели, основанные на архитектуре трансформеров, демонстрируют значительный прогресс в области искусственного интеллекта и обработки естественного языка, быстро нашли широкое применение в различных отраслях, включая программирование и разработку программного обеспечения, являются важным инструментом, способным существенно автоматизировать процессы, повысить эффективность разработки и улучшить качество кода в современных интегрированных средах разработки.

Научная новизна рецензируемого исследования состоит в представленных результатах анализа роли больших языковых моделей в современных интегрированных средах разработки, в отраженных авторами когнитивных и социальных аспектах взаимодействия разработчиков с большими языковыми моделями, оценках изменения методов работы и взаимодействия в командах, а также повышения эффективности программирования.

В тексте статьи выделены следующие разделы: Введение, Основная часть. Эволюция LLM и их применение в IDE, Роль LLM в трансформации современных IDE, Когнитивное

взаимодействие разработчиков с LLM, Применение LLM в IDE: анализ современных решений в крупных компаниях, Проблемы и вызовы интеграции LLM в интегрированные среды разработки нового поколения, Заключение и Библиография.

В статье проанализировано, как постоянное взаимодействие с большими языковыми моделями сказывается на нагрузке программистов, их мотивации и способности к творческому мышлению, на безопасности данных, защите интеллектуальной собственности и рисках чрезмерной зависимости от технологий искусственного интеллекта. Авторами отмечена важность обеспечения баланса между автоматизацией и развитием навыков разработчиков, чтобы не допустить снижения их квалификации. В публикации освещены проблемы интеграции больших языковых моделей: качество и предвзятость данных, отсутствие контекста, проблемы с интеллектуальной собственностью, зависимость от технологий, этические вопросы и безопасность. Текст статьи иллюстрирован двумя таблицами и тремя рисунками, содержит несколько фрагментов программных кодов.

Библиографический список включает 14 источников – научные публикации на русском и иностранных языках, на которые в тексте приведены адресные отсылки, что подтверждает наличие апелляции к оппонентам.

Рецензируемый материал соответствует направлению журнала «Программные системы и вычислительные методы», отражает результаты проведенной авторами работы, может вызвать интерес у читателей, поскольку содержит интересные сведения о роли больших языковых моделей в интегрированных средах разработки нового поколения, а также потенциальных угрозах и рисках их массового распространения. Статья рекомендуется к опубликованию.

Программные системы и вычислительные методы

Правильная ссылка на статью:

Шейнман В., Стариков Д.Д., Тюменцев Д.В., Вавилов Г.Д. Повышение эффективности процессов разработки программного обеспечения: контейнерные технологии // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.72015 EDN: JXRTJC URL: https://nbpublish.com/library_read_article.php?id=72015

Повышение эффективности процессов разработки программного обеспечения: контейнерные технологии

Шейнман Веред

ORCID: 0009-0003-5977-4211

независимый исследователь

3498838, Израиль, г. Хайфа, ул. Абба Хоши, 199

✉ vered.sheinman@gmail.com



Стариков Дмитрий Дмитриевич

ORCID: 0009-0009-8788-709X

студент; институт № 8 «Компьютерные науки и прикладная математика»; Московский авиационный институт (национальный исследовательский университет)

125993, Россия, г. Москва, Волоколамское шоссе, 4

✉ dd.starikov@gmail.com



Тюменцев Денис Викторович

ORCID: 0009-0003-5275-3223

независимый исследователь

670013, Россия, республика Бурятия, г. Улан-Удэ, ул. Ключевская, 40В

✉ tyumencev_dv@rambler.ru



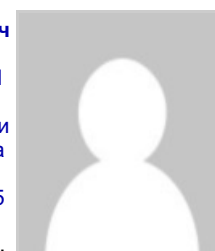
Вавилов Георгий Давидович

ORCID: 0000-0002-5951-3201

студент; отделение радиотерапии; Национальный медицинский исследовательский центр имени академика Е.Н. Мешалкина

630055, Россия, г. Новосибирск, ул. Речуновская, 15

✉ Frost20@narod.ru



[Статья из рубрики "Показатели качества и повышение надежности программных систем"](#)

DOI:

10.7256/2454-0714.2024.4.72015

EDN:

JXRTJC

Дата направления статьи в редакцию:

17-10-2024

Аннотация: В статье авторы рассматривают влияние контейнерных технологий на процессы разработки программного обеспечения (ПО). Основное внимание уделяется роли контейнеризации в оптимизации процессов развертывания и управления приложениями, а также в повышении гибкости и масштабируемости программных систем. В исследовании анализируются ключевые аспекты применения контейнеров, включая изоляцию приложений, повышение переносимости ПО между различными средами, а также снижение затрат на эксплуатацию благодаря оптимизации использования вычислительных ресурсов. Рассматриваются такие современные инструменты, как Docker и Kubernetes, которые позволяют стандартизировать и автоматизировать процессы развертывания и управления инфраструктурой. Авторы обсуждают примеры практического применения контейнерных технологий в крупных российских и иностранных компаниях, где контейнеризация значительно улучшила процессы разработки и эксплуатации ПО. Для анализа эффективности контейнерных технологий использованы методы сравнительного анализа, позволяющие оценить их влияние на гибкость инфраструктуры и производительность программных систем. Источниками данных служили научные публикации. Новизна исследования заключается в рассмотрении применения контейнерных технологий в контексте современных практик разработки ПО, что позволяет значительно ускорить процессы разработки, тестирования и развертывания программных продуктов. Полученные результаты показывают, что контейнеризация способствует улучшению производительности систем, упрощает управление приложениями и снижает затраты на эксплуатацию. Примеры практического использования Docker и Kubernetes в крупных компаниях демонстрируют, что контейнеризация значительно повышает гибкость инфраструктуры и масштабируемость решений, позволяя разработчикам легко адаптироваться к меняющимся условиям и требованиям рынка. В заключение подчеркивается, что контейнерные технологии играют ключевую роль в современных процессах разработки ПО, и их дальнейшее развитие будет способствовать еще более значительным улучшениям в области автоматизации и управления инфраструктурой программных систем.

Ключевые слова:

контейнеризация, разработка программного обеспечения, Docker, Kubernetes, масштабируемость, оптимизация ресурсов, эксплуатация программного обеспечения, автоматизация, платформы, изоляция процессов

Введение

Контейнерные технологии играют важную роль в разработке программного обеспечения (ПО), предоставляя эффективные решения для управления приложениями и инфраструктурой. В условиях роста сложности программных систем и стремительного темпа изменений в ИТ-индустрии контейнеризация обеспечивает необходимые инструменты для оптимизации процессов создания, тестирования и развертывания

элементов ПО.

Контейнерные платформы, такие как Docker, а также системы оркестрации контейнеров, включая Kubernetes, стали стандартом в области разработки. Эти технологии позволяют изолировать приложения и их зависимости в независимые, легковесные среды, что способствует повышению гибкости и переносимости ПО между различными вычислительными средами. Это устраняет проблемы, связанные с несовместимостью конфигураций и инфраструктуры, и упрощает управление жизненным циклом программных продуктов. Цель данной статьи – проанализировать, каким образом использование контейнерных технологий способствует повышению эффективности процессов разработки ПО.

Основная часть. Понятие и особенности контейнеризации

Методология создания и развертывания программных приложений в изолированных средах называется контейнеризацией. Контейнеры предоставляют средства для упаковки ПО и всех его зависимостей в самодостаточные, изолированные от хост-системы единицы, что позволяет значительно упростить переносимость и воспроизводимость приложений между различными вычислительными платформами [1]. В отличие от традиционной виртуализации, контейнеры используют общее ядро операционной системы, обеспечивая при этом полную изоляцию процессов, файловой системы и сетевых ресурсов для каждого элемента.

Контейнеры используют механизмы изоляции, такие как cgroups и namespaces, которые позволяют ограничить доступ к ресурсам и создать изолированные пространства имен для процессов, сетевых интерфейсов и систем [2]. Это гарантирует, что файлы, работающие в контейнерах, не будут влиять друг на друга, обеспечивая высокий уровень безопасности и изоляции без необходимости в полноценной виртуализации.

Контейнеризация предоставляет множество преимуществ для разработки и эксплуатации ПО. Это различные аспекты работы с приложениями – от повышения гибкости до оптимизации использования ресурсов (таблица 1).

Таблица 1. Преимущества контейнеризации [3, 4]

Преимущество	Описание	Применение на практике
Переносимость	Контейнеры обеспечивают возможность запуска продуктов на различных платформах без изменения кода.	Программисты могут разрабатывать ПО локально и затем без изменений развертывать его в облаке.
Масштабируемость	Легкость горизонтального масштабирования приложений путем добавления новых контейнеров.	Использование контейнеров для динамического увеличения числа экземпляров приложения при высокой нагрузке.
Изоляция	Полная изоляция процессов, файловых систем и сетевых ресурсов, что предотвращает конфликты.	Одновременный запуск нескольких файлов на одном сервере без риска конфликтов или помех.
Эффективное	Контейнеры используют	Повышение плотности

эффективное использование ресурсов	контейнеры используют общее ядро операционной системы, снижая затраты на ресурсы по сравнению с виртуальными машинами.	повышение плотности развертывания на физических серверах и снижение затрат на инфраструктуру.
Гибкость	Упрощение разработки и тестирования благодаря возможности быстро создавать и изменять контейнеры.	Быстрое тестирование и развертывание новых функций без прерывания основной работы продукта.

Преимущества контейнеризации, представленные в таблице, подчеркивают ее важность для современного процесса разработки и эксплуатации ПО. Более эффективное использование ресурсов и быстрота развертывания способствуют оптимизации инфраструктуры и уменьшению затрат на поддержание приложений. В совокупности, это делает контейнеризацию мощным инструментом для ускорения инноваций и повышения общей эффективности в ИТ-индустрии.

Основные технологии и инструменты контейнеризации

Согласно онлайн-опросу, проведенному американской компанией Statista в 2024 году, одной из самых популярных и востребованных технологий для контейнеризации является **Docker** – о его использовании сообщили 59% разработчиков [5]. Он предоставляет средства для создания, управления и развертывания контейнеров, позволяя легко упаковывать программные продукты вместе с их зависимостями в стандартные контейнерные образы (рис.1).

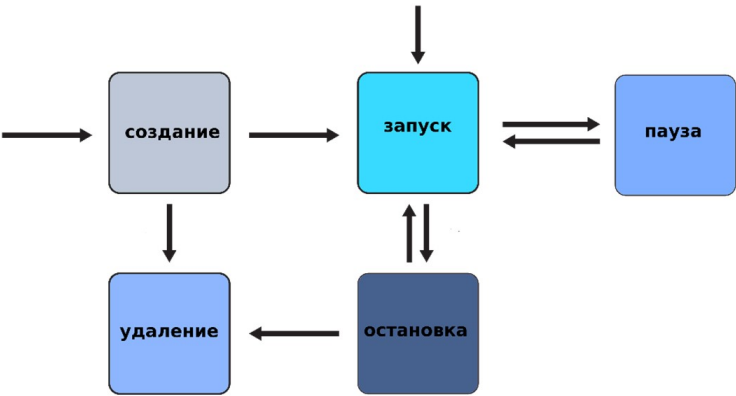


Рисунок 1. Основные состояния контейнеров в Docker

Компоненты Docker включают Docker Engine, Docker Hub и Docker Compose [6]. **Docker Engine** является ключевым элементом платформы, отвечающим за создание и выполнение контейнеров и взаимодействие с операционной системой. **Docker Hub** представляет собой облачную платформу для хранения и распространения контейнерных образов. Она позволяет разработчикам делиться готовыми элементами ПО с другими пользователями, а также загружать необходимые продукты из общего хранилища. **Docker Compose** является инструментом, который упрощает работу с многоконтейнерными приложениями, предоставляя возможность описывать конфигурации в виде YAML-файлов и управлять процессами их запуска и взаимодействия.

Встроенная в Docker система оркестрации контейнеров называется **Docker Swarm** [7].

Она обеспечивает простую интеграцию с существующими Docker-средами и позволяет распределять контейнеры между несколькими узлами, поддерживая балансировку нагрузки и автоматическое восстановление.

Kubernetes – это система оркестрации контейнеров с открытым исходным кодом, предназначенная для автоматизации процессов развертывания, управления и масштабирования программных продуктов, построенных на контейнерах. Она разработана для эффективной работы с крупными распределенными системами и предлагает мощные инструменты для управления большим числом контейнеров в различных вычислительных окружениях [8]. Это делает Kubernetes незаменимым инструментом для компаний, стремящихся к высокому уровню автоматизации и гибкости в разработке ПО (рис.2).

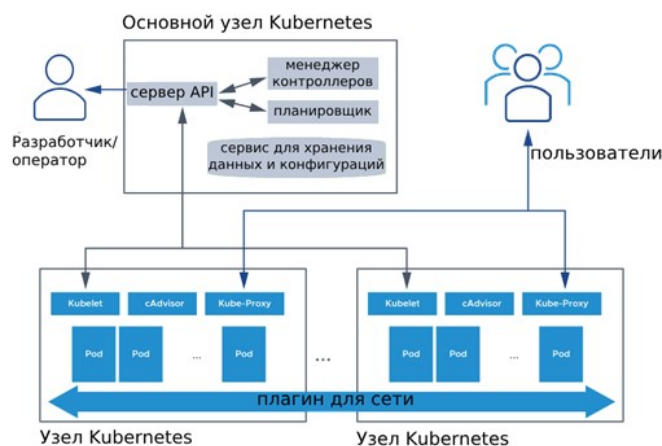


Рисунок 2. Архитектура системы Kubernetes

Разработчики и операторы взаимодействуют с системой через API сервер для управления контейнерными приложениями. Рабочие узлы содержат поды (Pod), которые представляют собой группы контейнеров, а также компоненты для мониторинга и управления – Kubelet, cAdvisor и Kube-Proxy. Узлы связаны через сетевые плагины, обеспечивающие взаимодействие контейнеров в кластере.

Согласно исследованиям Statista, в 2023 году 38% разработчиков отмечали повышение производительности, доступности и отказоустойчивости систем как главное преимущество Kubernetes. Ключевые возможности технологии включают автоматическое масштабирование, оркестрацию сетей и хранилищ, автовосстановление и балансировку нагрузки [9]. **Автоматическое масштабирование** позволяет системе динамически изменять количество контейнеров в зависимости от текущей нагрузки на приложение, что способствует эффективному использованию ресурсов. **Оркестрация сетей и хранилищ** обеспечивает управляемое взаимодействие между контейнерами и предоставляет доступ к внешним хранилищам данных, упрощая работу с распределенными файлами. **Функция автовосстановления** автоматически перезапускает или заменяет контейнеры в случае их выхода из строя, поддерживая бесперебойную работу системы. **Балансировка нагрузки** распределяет входящий трафик между контейнерами, что позволяет достичь оптимальной производительности продуктов, даже при высокой нагрузке.

OpenShift –это платформа контейнеризации, основанная на Kubernetes. Она предоставляет дополнительные инструменты для управления безопасностью, разработкой и эксплуатацией файлов в контейнерах (рис.3).



Рисунок 3. Схема OpenShift

OpenShift ориентирован на корпоративный сегмент и включает средства для разработки приложений [10]. Пользователь инициирует резервное копирование через команду, которая взаимодействует с Kubernetes API. Контроллер резервного копирования управляет процессом, сохраняя конфигурации в хранилище и загружая данные в облачный провайдер с созданием снимков дисков. Технология предлагает улучшенные механизмы управления безопасностью контейнеров и процессами аутентификации, а также интеграцию с CI/CD (непрерывной интеграцией и развертыванием), что делает ее эффективным инструментом для крупных организаций с комплексными требованиями к инфраструктуре.

CRI-O – это проект с открытым исходным кодом, специально разработанный для использования с Kubernetes. Он предоставляет минималистичную, но эффективную среду выполнения контейнеров, что позволяет Kubernetes напрямую управлять контейнерами без необходимости использования более сложных решений, таких как Docker (рис.4)

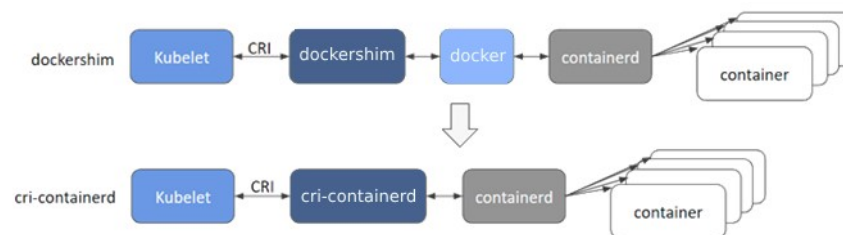


Рисунок 4. Сравнение взаимодействия Kubernetes с контейнерами через dockershim и cri-containerd

На схеме показаны два различных подхода взаимодействия Kubernetes с контейнерами. В первом случае используется dockershim – промежуточный слой, который позволяет Kubernetes взаимодействовать с контейнерами через Docker. В этой архитектуре Kubernetes с помощью компонента Kubelet отправляет команды через интерфейс CRI (Container Runtime Interface) к Docker, который управляет контейнерами через низкоуровневый компонент containerd. Контейнеры создаются и управляются Docker, что добавляет лишний уровень абстракции.

Во втором подходе используется CRI-O. В этом случае Kubelet взаимодействует напрямую с CRI-O через тот же интерфейс CRI, что исключает необходимость использования Docker. Вместо этого CRI-O напрямую управляет контейнерами через containerd, что упрощает архитектуру, улучшает производительность и устраняет избыточные слои. CRI-O сосредоточен на минимизации функциональности для выполнения только тех задач, которые необходимы для работы с Kubernetes. Это делает его более легковесной и эффективной альтернативой для Kubernetes-кластеров [11].

Podman – это инструмент с открытым исходным кодом, обладающий функциональностью, схожей с Docker, но с некоторым отличием: Podman не требует наличия фонового

процесса для управления контейнерами. Это делает его более безопасной альтернативой Docker, особенно в средах с повышенными требованиями к безопасности (рис.5)

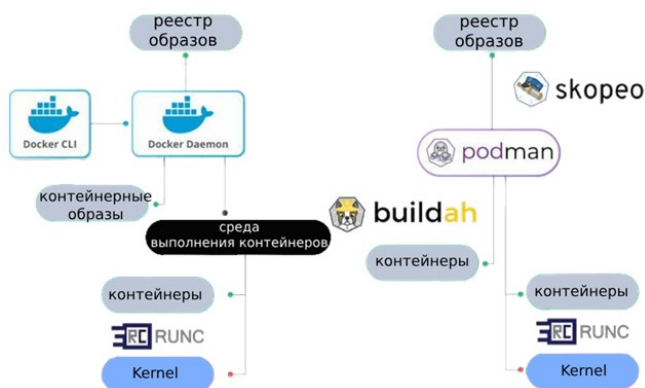


Рисунок 5. Сравнение архитектуры Docker и Podman

Podman интегрируется с другими инструментами, такими как Buildah для создания контейнерных образов и Skopeo для их управления, что делает его надежным инструментом для работы с контейнерами в средах с высокими требованиями к безопасности.

Согласно исследованию, проведенному в Национальном научно-вычислительном центре энергетических исследований (NERSC) в сотрудничестве с американским производителем ПО Red Hat, технология Podman адаптирована для использования в области высокопроизводительных вычислений (HPC). Podman предлагает такие преимущества, как возможность запуска контейнеров без привилегий root, что повышает уровень безопасности, а также возможность сборки контейнерных образов непосредственно на узлах суперкомпьютера Perlmutter [\[12\]](#). Результаты тестирования различных режимов работы Podman, включая podman-exec и Podman container-per-process, продемонстрировали производительность, сопоставимую с непосредственным использованием аппаратных ресурсов, что делает Podman перспективным инструментом для использования.

Применение контейнеров на различных этапах разработки ПО

Контейнеризация предоставляет разработчикам возможность эффективно управлять программными системами на всех стадиях их жизненного цикла – от разработки до эксплуатации. Контейнеры помогают создавать изолированные среды, которые можно быстро развертывать и тестировать, что особенно важно для крупных компаний [\[13\]](#).

На этапе разработки контейнеры используются для создания стандартизированных сред, которые обеспечивают одинаковую работу приложения на различных устройствах. Например, американская компания **Google** использует контейнеризацию для разработки своих сервисов, включая Google Search и Gmail [\[14\]](#). Контейнеры позволяют разработчикам создавать приложения в идентичных продуктивных средах, что исключает проблемы, связанные с различиями в конфигурации операционных систем и библиотек. Это существенно упрощает процесс написания кода и его последующее тестирование. Google Cloud предоставляет экосистему, необходимую для более быстрой разработки и внедрения ПО без ущерба для безопасности. Согласно отчетам, 98 % клиентов компаний, которые используют инструменты Google для контейнеризации, развертывают приложение с первой попытки менее чем за 5 минут.

Тестирование ПО – это важнейший этап разработки, на котором контейнерные технологии позволяют создавать изолированные и воспроизводимые тестовые среды. В американском стриминговом сервисе **Netflix** контейнеры играют ключевую роль в ускорении тестирования и улучшении качества новых версий приложений и сервисов. Для этих целей Netflix использует Docker и Kubernetes [\[15\]](#). С помощью Docker компания создает контейнеры, в которых изолируются различные версии приложений, что позволяет проводить тестирование в идентичных средах, избегая проблем, связанных с различиями в конфигурации. Kubernetes помогает Netflix масштабировать тестовые среды для имитации различных сценариев работы потокового сервиса под разными уровнями нагрузки. Это позволяет тестировать производительность и устойчивость приложений к высоким нагрузкам, быстрее выявлять потенциальные ошибки и снижать риски, связанные с обновлениями приложений.

Российская компания «**Яндекс**» применяет контейнеризацию на этапе тестирования для проверки новых функциональных возможностей своих продуктов. Для этих целей Яндекс использует такие инструменты, как Docker и Kubernetes. Docker позволяет создавать изолированные контейнеры, в которых разрабатываются и тестируются отдельные компоненты сервисов [\[16\]](#). Эти контейнеры помогают воспроизводить тестовые среды, идентичные реальным, что снижает вероятность возникновения ошибок.

Kubernetes используется для автоматизации развертывания контейнеров и управления ими на кластерах серверов. Это позволяет Яндексу масштабировать тестовые окружения и проводить нагрузочные тесты для проверки производительности новых функций под разными уровнями нагрузки. Благодаря такой интеграции Docker и Kubernetes, компания может параллельно тестировать множество версий своих продуктов, что ускоряет цикл разработки и снижает риски внедрения обновлений в рабочие среды.

На этапе **развертывания и эксплуатации** ПО контейнеризация предоставляет значительные преимущества, обеспечивая гибкость и масштабируемость приложений. Примером этого может служить практика **American Express (США)**, которая активно использует контейнеризацию в процессах разработки и эксплуатации ПО [\[17\]](#). Компания применяет Docker и Kubernetes для автоматизации развертывания финансовых сервисов, что позволяет им гибко масштабировать приложения в зависимости от изменяющихся требований и нагрузки на систему.

После развертывания контейнеры продолжают играть важную роль в **эксплуатации и масштабировании ПО**, обеспечивая автоматическое управление ресурсами и гибкость в изменении конфигураций.

Заключение

Контейнерные технологии способствуют повышению эффективности разработки ПО за счет изоляции приложений и их зависимостей, что облегчает переносимость между различными средами. Это позволяет устранить проблемы совместимости, ускоряя процессы разработки, тестирования и развертывания. Использование контейнеров оптимизирует распределение ресурсов, снижая нагрузку на инфраструктуру и улучшая масштабируемость программных продуктов. Такие платформы, как Docker и Kubernetes, обеспечивают автоматизацию и гибкость управления файлами, что сокращает время на внедрение новых функций и минимизирует затраты на поддержание инфраструктуры.

Финансирование. Исследование не имело спонсорской поддержки.

Вклад авторов. Все авторы внесли равный вклад в написание настоящей статьи.

Конфликт интересов. Авторы заявляют об отсутствии конфликта интересов.

Funding. The study had no sponsorship.

Authors contributions. All authors contributed equally to this article.

Conflict of interest. The authors declare no conflict of interest.

Библиография

1. Белодед Н.И., Демиденко К.Г. Развитие и применение технологии контейнеризации в разработке программного обеспечения // Актуальные проблемы научных исследований: теоретические. – 2023. – С. 57.
2. Bondarenko A.S., Zaytsev K.S. Using container management systems to build distributed cloud information systems with microservice architecture // International Journal of Open Information Technologies. – 2023. – V. 11. – № 8. – P. 17-23.
3. Можаровский Е.А. Разработка мобильных приложений: от идеи до рынка // Современные научные исследования и инновации. – 2024. – № 1.
4. Aluev A. Scalable web applications: a cost-effectiveness study using microservice architecture // Cold Science. – 2024. – № 8. – С. 32-38.
5. Muzumdar P, Bhosale A., Basyal G., Kurian G. Navigating the Docker ecosystem: a comprehensive taxonomy and survey // arXiv preprint arXiv:2403.17940. – 2024.
6. Christudas B.A. Introducing Docker // Java Microservices and containers in the Cloud: with Spring Boot, Kafka, PostgreSQL, Kubernetes, Helm, Terraform and AWS EKS. – Berkeley, CA: Apress, 2024. – P. 281-343.
7. Higgins T., Jha D.N., Ranjan R. Swarm Storm: an automated chaos tool for docker swarm applications // Proceedings of the 33rd International Symposium on High-Performance Parallel and Distributed Computing. – 2024. – P. 367-369.
8. Миронов Т.О. Построение информационной архитектуры системы автоматизации цикла выпуска программного обеспечения // Инжиниринг предприятий и управление знаниями. – С. 280.
9. Дудак А.А. Сравнительный анализ инструментов разработки для систем управления проектами: преимущества стек-технологий TypeScript и React // Новая наука: от идеи к результату. – 2024. – № 9. – С. 32-40.
10. Глумов К.С. Шаблоны, лежащие в основе java, kubernetes и современных распределенных систем // Хлебопечение России. – 2024. – Т. 68. – № 1. – С. 6-12.
11. Poggiani L., Puliafito C., Virdis A., Mingozzi E. Live Migration of Multi-Container Kubernetes Pods in Multi-Cluster Serverless Edge Systems // Proceedings of the 1st Workshop on Serverless at the Edge. – 2024. – P. 9-16.
12. Stephey L., Canon S., Gaur A., Fulton D., Younge A. Scaling Podman on Perlmutter: Embracing a community-supported container ecosystem // 2022 IEEE/ACM 4th International Workshop on Containers and New Orchestration Paradigms for Isolated Environments in HPC (CANOPIE-HPC). – IEEE, 2022. – P. 25-35.
13. Sidorov D. Leveraging web components for scalable and maintainable development // Sciences of Europe. – 2024. – № 150. – P. 87-89.
14. Чередников К. А., Лаврова Е. Д., Марухленко А. Л. Современный взгляд на контейнеризацию // Современные информационные технологии и информационная безопасность. – 2023. – P. 115-119.
15. Erdenebat B., Bud B., Kozsik T. Challenges in service discovery for microservices deployed in a Kubernetes cluster – a case study // Infocommunications Journal. – 2023. – V. 15. – № SI. – P. 69-75.

16. Макарова Н.В., Савичев Д.Е. Применение методов искусственного интеллекта при эксплуатации программного обеспечения // Актуальные проблемы экономики и управления. – 2023. – № 1. – С. 17.

17. Косарев В.Е., Добридникова С.Л. Практические аспекты разработки и внедрения цифрового рубля в банковские информационные системы // Инновации и инвестиции. – 2023. – № 2. – С. 143-149.

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Рецензируемая работа посвящена Повышению эффективности процессов разработки программного обеспечения за счет применения контейнерных технологий.

Методология исследования базируется на изучении и обобщении научных публикаций по рассматриваемой теме и практики применения различных инструментов контейнеризации.

Актуальность работы обусловлена увеличением количества разрабатываемых компьютерных программ и ростом их сложности, что требует повышения эффективности процессов разработки программного обеспечения, которое может быть обеспечено в частности за счет применения контейнерных технологий в разработке программного обеспечения.

Научная новизна рецензируемого исследования состоит в выводах о том, что контейнерные технологии способствуют повышению эффективности разработки ПО за счет изоляции приложений и их зависимостей, что облегчает переносимость между различными средами, это позволяет устранить проблемы совместимости, ускоряя процессы разработки, тестирования и развертывания, оптимизирует распределение ресурсов, снижая нагрузку на инфраструктуру и улучшая масштабируемость программных продуктов.

В тексте статьи выделены следующие разделы: Введение, Основная часть. Понятие и особенности контейнеризации, Основные технологии и инструменты контейнеризации, Применение контейнеров на различных этапах разработки ПО, Заключение и Библиография.

В статье под контейнеризацией понимается методология создания и развертывания программных приложений в изолированных средах; отмечено, что контейнеры предоставляют средства для упаковки программного обеспечения и всех его зависимостей в самодостаточные, изолированные от хост-системы единицы, что позволяет значительно упростить переносимость и воспроизводимость приложений между различными вычислительными платформами; в отличие от традиционной виртуализации, контейнеры используют общее ядро операционной системы, обеспечивая при этом полную изоляцию процессов, файловой системы и сетевых ресурсов для каждого элемента. В публикации детально рассмотрены преимущества контейнеризации: переносимость, масштабируемость, изоляция, эффективное использование ресурсов, гибкость. Авторами освещены особенности применения таких инструментов контейнеризации как Docker со встроенной системой оркестрации контейнеров Docker Swarm, а также Kubernetes – системы оркестрации контейнеров с открытым исходным кодом, предназначенной для автоматизации процессов развертывания, управления и масштабирования программных продуктов, построенных на контейнерах, а также платформы контейнеризации OpenShift и CRI-O. Также в статье отражены возможности Контейнеризации для эффективного управления программными

системами на всех стадиях их жизненного цикла – от разработки до эксплуатации.

Библиографический список включает 17 источников – научные публикации на русском и английском языках, на которые в тексте приведены адресные отсылки, что подтверждает наличие апелляции к оппонентам.

Рецензируемый материал соответствует направлению журнала «Программные системы и вычислительные методы», отражает результаты проведенной авторами работы, может вызвать интерес у читателей, поскольку содержит интересные сведения о применении контейнерных технологий в разработке программного обеспечения, а поэтому рекомендуется к опубликованию.

Программные системы и вычислительные методы

Правильная ссылка на статью:

Пекунов В.В. Объектно-транзакционные модели программ на алгоритмических языках // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.69228 EDN: JWFWBE URL: https://nbpublish.com/library_read_article.php?id=69228

Объектно-транзакционные модели программ на алгоритмических языках

Пекунов Владимир Викторович

доктор технических наук

Инженер-программист, ОАО "Информатика"

153000, Россия, Ивановская область, г. Иваново, ул. Ташкентская, 90

✉ pekunov@mail.ru



[Статья из рубрики "Системный анализ, поиск, анализ и фильтрация информации"](#)

DOI:

10.7256/2454-0714.2024.4.69228

EDN:

JWFWBE

Дата направления статьи в редакцию:

05-12-2023

Аннотация: Данная работа посвящена вопросу представимости программ, написанных на алгоритмических языках, с помощью формализмов, базирующихся на идее применения предельной частично транзакционной памяти, включающей единственную транзакционную ячейку и множество обычных ячеек. При этом предполагается, что такие формализмы основаны на понятии сети объектов, представляющих как основные, так и вспомогательные элементы решающей задачи. Объекты функционируют в памяти указанного вида, выполняя методы, содержащие исключительно ветвящийся код, лишенный циклов. Циклы при таком подходе заменяются многократным специальным пересогласованием сети объектов, схожим с реализованным в классической транзакционной памяти. Исходя из самых общих представлений о процессе решения задачи в некоторой предметной области, впервые вводится концепция объектно-транзакционной модели и формулируются их основные свойства. При формулировании структуры и основных принципов функционирования объектно-транзакционных моделей используются методы дискретной математики, теории алгоритмов. Вводится понятие предельной частично транзакционной памяти, содержащей единственную

транзакционную ячейку со специальным согласованием. Описываются особенности согласования такой памяти в контексте предлагаемых моделей. Выдвигается гипотеза о реализуемости произвольных алгоритмов с помощью объектно-транзакционных моделей. Описаны основные принципы функционирования таких моделей, сформулированы их основные свойства. Вводятся понятия предельных непараллельной и параллельной моделей. Доказывается, что предельная непараллельная модель способна выполнить произвольный разрешимый по Тьюрингу алгоритм. Доказывается, что предельная параллельная модель из $K+2$ узлов эквивалентна системе из K параллельно работающих машин Тьюринга и, соответственно, способна выполнить произвольный разрешимый по Тьюрингу алгоритм, подразумевающий наличие K параллельных ветвей. Таким образом, выдвинутая в работе гипотеза о реализуемости произвольных алгоритмов доказана.

Ключевые слова:

последовательный алгоритм, модель программы, транзакционная память, специальное согласование, предельные теоремы, объектно-транзакционная модель, машина Тьюринга, теория, параллельный алгоритм, реализуемость алгоритмов

Транзакционная память [1, 2] – достаточно хорошо изученный формализм для построения параллельных программ, отличающийся простотой распараллеливания, связанной с отсутствием необходимости в явных синхронизациях [3]. Теоретически интересным является вопрос, можно ли с помощью некоторого формализма, базирующегося на той или иной предельной модификации транзакционной памяти, строить произвольные (последовательные или параллельные) программы, используя единственную алгоритмическую конструкцию — ветвление. При этом явные циклы можно заменить многократным условным пересогласованием транзакции. Достаточно очевидно, что такая замена требует работы с данными, сохраняющими свои значения при откате транзакции [4], а значит и перехода от полной изоляции транзакций к их частичной изоляции. Такой подход к реализации транзакционной памяти известен [5], в данной работе используется авторский вариант такой памяти [6] – предельная частично транзакционная память, включающая единственную специальную транзакционную ячейку.

Таким образом, целью данной работы является исследование проблемы описания последовательных и параллельных алгоритмов на базе формализмов, основанных на применении предельной частично транзакционной памяти в условиях единственной алгоритмической конструкции ветвления. Такая постановка проблемы является новой. Поставим задачу разработки и исследования свойств такого формализма, исходя из самых общих представлений о концептах, используемых для построения алгоритма, решающего некоторую задачу.

Общая постановка проблемы

Пусть решается некоторая задача в произвольной предметной области. Такая область может быть описана в терминах некоторых понятий, то есть мыслей, обобщающих и выделяющих предметы некоторых классов по общим признакам. Следовательно, предметная область может быть определена как иерархия классов (понятий), описывающих общие свойства объектов, подпадающих под данные понятия. Предположим, что для задачи известен общий, разрешимый по Тьюрингу алгоритм ее решения, причем он может быть декомпозирован на серию многократно пересогласуемых

последовательных и параллельных ациклических процессов (методов объектов, относящихся к классам соответствующей предметной области), последовательность активизации которых определяется по схеме сетевого графика работ.

Тогда формальным описанием процесса решения исходной задачи будет сеть взаимосвязанных объектов, являющихся экземплярами классов соответствующей предметной области, взаимодействующих, например, через общую память. Данное описание может быть как предельным, включающим единственный объект, исчерпывающим образом описывающий процесс решения задачи, так и неопредельным, включающим более одного объекта.

Можно предположить, что в самом элементарном случае полное решение задачи выполняется путем однократного запуска процессов сети в требуемом порядке. Однако очевидно, что в общем случае может потребоваться управляемый многократный перезапуск сети, например, в результате пересогласования по некоей специальной транзакционной ячейке. Интересными следствиями наличия такого механизма являются следующие:

- а) если пересогласование по транзакционной памяти не требуется, то сеть всегда завершает работу за конечное время;
- б) с помощью пересогласования в условиях предельной частично транзакционной памяти может быть реализовано неоднократное исполнение одного и того же алгоритма, что является приемлемой альтернативой циклам.

Введем теперь понятие предельной частично транзакционной памяти. Пусть она представляет собой ассоциативный кортеж $R_K = (V_K, T_K)$, ключами K которого являются некие идентификаторы, а значениями V_K могут быть произвольные структуры данных, причем каждая ячейка может иметь собственный тип T_K со значениями из множества $\{a, t\}$. Ячейки типа a являются элементарными ячейками с атомарным доступом, *единственная* ячейка типа t относится к транзакционной памяти, локальные значения которой, фактически, хранятся в векторе $V_K = (L_1, \dots, L_N)$. Каждый i -й элемент такого вектора является двойкой (S_i, C_i) , где S_i – метка времени первой модификации ячейки в i -ом потоке, C_i – локальная копия ячейки для i -го потока (всего N параллельных потоков – параллельных путей исполнения).

Пусть C – множество классов. Класс S представляет собой тройку (N_c, F, M) , где N_c – идентификатор (имя) класса, F – поля, M – методы. Определим отношение иерархии $\text{parent}: C \rightarrow C$, заданное функцией

$$\forall a \in C \exists \text{parent}(a) = \begin{cases} (\varepsilon, \emptyset, \emptyset), & \text{если } a \text{ есть корень иерархии;} \\ c \in C, & \text{если } c \text{ есть предок } a, \end{cases}$$

где ε – пустая цепочка. Пусть каждый метод множества M представляет собой ветвящийся алгоритм, не содержащий циклов, имеющий полный доступ ко всем полям узла-объекта и ко всем ячейкам некоторой общей предельной частично транзакционной памяти, если таковая присутствует.

Гипотеза. Модель, включающая сетевой граф (P, E) , содержащий множество P , включающее объекты классов C текущей предметной области (соединенные дугами из множества E), вызываемые в определенном выше порядке [с учетом того, что методы объектов, между которыми не определено отношение следования (по структуре модели),

всегда выполняются в параллель], функционирующие по ветвящимся алгоритмам в условиях предельной частично транзакционной памяти со специальным согласованием, способна выполнить произвольный разрешимый по Тьюрингу последовательный или параллельный алгоритм.

Назовем такие модели объектно-транзакционными (ОТМ).

Непараллельные ОТМ

Введем понятие *предельной непараллельной объектно-транзакционной модели*. Пусть такая модель является двойкой (R, N) , где R – ассоциативный кортеж предельной частично транзакционной памяти, N – сеть из единственного узла-объекта A , относящегося к некоторому классу.

Введем специальный механизм согласования единственной транзакционной ячейки. Пусть она имеет идентификатор RESTART и хранит ссылку на текущий исполняемый метод объекта A (сразу заметим, что в случае ОТМ из нескольких узлов это будет список пар вида $(A, M[A])$, где A – ссылка на узел, $M[A]$ -- ссылка на метод узла A), отличающаяся тем, что

- а) в начале исполнения модели данная ячейка хранит ссылку на специальный метод, всегда вызываемый первым;
- б) любое присваивание данной ячейке ссылки на произвольный метод неминуемо приводит к необходимости пост-пересогласования текущей транзакции (всей сети, по окончании ее текущей интерпретации) с вызовом указанного метода;
- в) перед пост-пересогласованием текущей транзакции определяются подлежащие вызову методы объектов – это методы, ссылки на которые занесены в RESTART. Далее ячейка RESTART очищается и начинается пост-пересогласование транзакции (всей сети с самого начала).

Отметим, что все ячейки памяти типа а не влияют на процесс согласования.

Теорема 1. Предельная непараллельная ОТМ способна реализовать произвольный разрешимый по Тьюрингу алгоритм.

Доказательство. Рассмотрим произвольную программу на алгоритмическом языке высокого уровня C (с расширением, реализующим требуемый механизм согласования предельной частично транзакционной памяти). Если такая программа не содержит циклов, то она может быть реализована произвольным методом (реализующим произвольный ветвящийся алгоритм) единственного объекта предельной непараллельной ОТМ по определению.

Пусть программа содержит циклы. Эквивалентно трансформируем ее по следующим правилам:

- а) свяжем с каждым оператором некую условную метку;
- б) введем нетранзакционную переменную CURLABEL, хранящую имя метки текущего оператора;
- в) в начале программы CURLABEL содержит метку начального оператора;
- г) каждый оператор с меткой предваряем конструкцией вида

```
[else] if (CURLABEL == метка_текущего_оператора) {
```

д) в каждом операторе с меткой производим соответствующие присваивания переменной CURLABEL меток операторов, выполняемых следующими;

е) в конце каждого оператора ставим закрывающий блок знак «}», перед которым (за исключением операторов, исполняемых последними), ставим конструкцию рассогласования текущей транзакции (приводящую к ее последующему повтору по завершению текущего сеанса обработки сети) путем присваивания переменной RESTART ссылки на текущий метод;

ж) сначала преобразуем все циклы в цикл while;

з) каждый цикл while вида;

```
Метка1: while(условие) { Оператор_A; ... Оператор_B; } Оператор_C;
```

заменим на условные операторы:

```
if (CURLABEL == Метка1 && условие) { CURLABEL = Метка_A; RESTART = ...; }
```

```
else if (CURLABEL == Метка1 && !условие) { CURLABEL = Метка_C; RESTART = ...; }
```

```
else if (CURLABEL == Метка_A) { Оператор_A; CURLABEL = ...; RESTART = ...; }
```

```
...
```

```
else if (CURLABEL == Метка_B) { Оператор_B; CURLABEL = Метка_1; RESTART = ... }
```

```
else if (CURLABEL == Метка_C) { Оператор_C; ... }
```

и) вся программа заключается в транзакционный блок;

к) все переменные программы оформляем как ячейки типа а предельной частично транзакционной памяти.

Получили эквивалентную программу, которая может быть исполнена произвольным методом ОТМ по определению. Итак, предельная непараллельная ОТМ способна реализовать произвольную программу (например, на алгоритмическом языке высокого уровня С), которой, как известно, может быть программа, реализующая алгоритм функционирования машины Тьюринга (способной реализовать произвольный разрешимый по Тьюрингу алгоритм), следовательно, такая ОТМ способна реализовать произвольный разрешимый по Тьюрингу алгоритм. Также отметим несомненную прямую структурную аналогию между полученной после преобразования программой на С и кодом для машины Тьюринга.

Доказано.

Следствие. Поскольку предельная непараллельная ОТМ способна реализовать произвольный, разрешимый по Тьюрингу алгоритм, следовательно, и непредельная непараллельная ОСМ способна реализовать такой алгоритм (например, используя один свой произвольный узел).

Параллельные ОТМ

Введем понятие *предельной параллельной (К-поточной) объектно-транзакционной модели*. Пусть такая модель является двойкой (R, N), где R – ассоциативный кортеж

предельной частично транзакционной памяти, N – сеть, включающая узлы с пустыми методами (исток и сток), а также K рабочих узлов с непустыми методами, в каждый из которых входит дуга из истока и из каждого исходит дуга в сток. Пусть модель выполняет транзакционный блок по всем описанным выше правилам (возможно, с многократным пост-пересогласованием), учитывая, что процессы узлов, находящихся в ветвях, для которых не определено отношение следования, выполняются параллельно.

Теорема 2. Предельная параллельная (K -поточная) ОТМ способна реализовать произвольный разрешимый по Тьюрингу алгоритм с K параллельными ветвями.

Доказательство. Выше было показано, что одноузловая непараллельная ОТМ способна реализовать произвольный, разрешимый по Тьюрингу алгоритм. Здесь мы имеем систему из K узлов (которые могут работать параллельно), каждый из которых можно рассматривать как отдельную ОТМ, взаимодействующую с прочими ОТМ через общую память. Таким образом, всю систему можно рассматривать как K параллельно работающих машин Тьюринга, обменивающихся данными, например, через общую ленту. Соответственно, данная система способна выполнить произвольный разрешимый по Тьюрингу алгоритм с K параллельными ветвями.

Следствие. Поскольку предельная параллельная (K -поточная) ОТМ способна реализовать произвольный параллельный K -поточный алгоритм, разрешимый по Тьюрингу, следовательно и неопредельная параллельная ОСМ с не менее чем K параллельными ветвями способна реализовать такой алгоритм.

Основные свойства ОТМ

В заключение сформулируем некоторые основные свойства объектно-транзакционных моделей, предложенных в данной работе. Данные свойства вытекают либо из основных структурно-функциональных особенностей моделей, либо из доказанных выше положений. Итак, основные свойства:

1. **Предельная единичность.** Каждой уникальной программе на некотором алгоритмическом языке высокого уровня соответствует не менее одной эквивалентной ей ОТМ. Данное свойство вытекает из потенциальной реализуемости с помощью ОТМ произвольного разрешимого по Тьюрингу алгоритма (теорема 1). Следовательно, эквивалентом может быть как минимум предельная непараллельная ОТМ, реализующая машину Тьюринга на заданном языке высокого уровня, выполняющую указанный алгоритм. Кроме того, не исключено построение иной реализующей ОТМ, например, подобной по структуре основному дереву графа блок-схемы данного алгоритма и содержащей один и более узлов с методами на заданном языке высокого уровня.
2. **Алгоритмическая полнота по Тьюрингу** (следует из теоремы 1).
3. **Естественное описание параллелизма** (параллельные и непараллельные процессы четко определяются структурой модели).
4. **Атранзакционная конечность.** Если механизм согласования единственной ячейки транзакционной памяти не задействуется, то ОТМ всегда завершает работу за конечное время, поскольку имеет сетевую структуру, что в данном случае делает заикливание принципиально невозможным.
5. **Принципиальная неоднородность используемых видов памяти** при избыточной (исключающей простое дублирование тел циклов) реализации алгоритмов, содержащих циклы со счетчиком или межвитковыми зависимостями. Следует из того факта, что

модель имеет сетевую структуру, ее методы также имеют ветвящуюся структуру, поэтому такие циклы избыточно могут быть реализованы только за счет пересогласования модели (требуется ячейка RESTART предельной частично транзакционной памяти) с сохранением изменяющегося контекста [счетчиков или иных общих для витков цикла переменных (требуется ячейки типа а частично транзакционной памяти)].

Выводы

В данной работе рассмотрена проблема представимости параллельных и непараллельных алгоритмов с помощью объектно-транзакционных моделей, базирующихся на идее предельной частично транзакционной памяти, содержащей единственную транзакционную ячейку со специальным согласованием, в условиях наличия единственной алгоритмической конструкции – ветвления. Впервые сформулировано понятие таких моделей, описаны основные принципы их функционирования, сформулированы их основные свойства. Доказаны теоремы о разрешимости такими моделями произвольных последовательных алгоритмов, вычислимых по Тьюрингу, а также о разрешимости произвольных алгоритмов с K параллельными ветвями, вычислимых системой из K параллельно работающих машин Тьюринга, взаимодействующих через общую ленту.

Библиография

1. Черняк, Л. Транзакционная память-первые шаги / Л. Черняк // Открытые системы. СУБД. – 2007. – № 4. – С. 12-15.
2. Marathe, V.J., Scherer, W.N., Scott, M.L. (2005). Adaptive Software Transactional Memory. In: Fraigniaud, P. (eds) Distributed Computing. DISC 2005. Lecture Notes in Computer Science, vol 3724. Springer, Berlin, Heidelberg.
https://doi.org/10.1007/11561927_26
3. M. Herlihy and J. E. B. Moss. Transactional memory: Architectural support for lock-free data structures. In A. J. Smith, editor, Proceedings of the 20th Annual International Symposium on Computer Architecture. San Diego, CA, May 1993, pages 289–300. ACM, 1993.
4. V. Luchangco and V. J. Marathe. Transaction communicators: Enabling cooperation among concurrent transactions. In Proceedings of the 16th ACM Symposium on Principles and Practice of Parallel Programming, PPoPP '11, pages 169-178, New York, NY, USA, 2011. ACM.
5. Miculan, M., Peressotti, M. Software Transactional Memory with Interactions. Proceedings of the 21st Italian Conference on Theoretical Computer Science, Ischia, Italy, September 14-16, 2020, pp. 67-80.
6. Пекунов В.В. Сверхоптимистичные вычисления: концепция и апробация в задаче о моделировании электростатической линзы // Программные системы и вычислительные методы. – 2020. № 2. – С. 37-44. DOI: 10.7256/2454-0714.2020.2.32232. URL: https://e-notabene.ru/ppsvm/article_32232.htm

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Представленная на рецензирование научная статья на тему: «Объектно-транзакционные модели программ на алгоритмических языках» представляет собой актуальное исследование.

В рецензируемой статье авторами определена цель исследования - исследование

проблемы описания последовательных и параллельных алгоритмов на базе формализмов, основанных на применении предельной частично транзакционной памяти в условиях единственной алгоритмической конструкции ветвления.

Поставлены также задачи исследования. В частности, задача разработки и исследования свойств формализма, исходя из самых общих представлений о концептах, используемых для построения алгоритма, решающего некоторую задачу. В работе вводятся понятия предельной частично транзакционной памяти, предельной непараллельной объектно-транзакционной модели.

Рассмотрена проблема представимости параллельных и непараллельных алгоритмов с помощью объектно-транзакционных моделей, базирующихся на идее предельной частично транзакционной памяти, содержащей единственную транзакционную ячейку со специальным согласованием, в условиях наличия единственной алгоритмической конструкции – ветвления. По мнению авторов, впервые сформулировано понятие таких моделей, описаны основные принципы их функционирования, сформулированы их основные свойства. Сказанное позволяет сделать вывод о наличии новизны исследования.

В работе авторами сформулированы и представлены результаты исследования. В частности, доказаны теоремы о разрешимости такими моделями произвольных последовательных алгоритмов, вычислимых по Тьюрингу, а также о разрешимости произвольных алгоритмов с K параллельными ветвями, вычислимых системой из K параллельно работающих машин Тьюринга, взаимодействующих через общую ленту.

Проведенный анализ статьи также показал, что при подготовке статьи использованы различные источники и научная литература. Однако, список использованных источников и литературы достаточно скромнен. Библиографический список составляет 6 источников, среди которых представлены научные публикации зарубежных и отечественных исследователей. В этой связи, полагаем, что источниковая база исследования не позволила авторам статьи определить уровень разработанности проблемы в отечественной науке. В статье также, по нашему мнению, не удалось развернуть полноценную научную дискуссию. Однако данное обстоятельство, в целом не влияет на ее научность, глубину исследовательской концепции и научные результаты, отраженные в статье.

Статья структурирована. Положительно следует оценить наличие методологической базы исследования, подходов. В частности, в данной работе используется авторский вариант – предельная частично транзакционная память, включающая единственную специальную транзакционную ячейку.

Статья интересная. В статье представлены формулы, теоремы и их доказательства. Статья способна вызвать читательский интерес.

Считаем, что рецензируемая научная статья на тему: «Объектно-транзакционные модели программ на алгоритмических языках» соответствует необходимым требованиям, предъявляемым к такому виду научных работ. Она способна вызвать читательский интерес и рекомендуется к опубликованию в искомом научном журнале.

Программные системы и вычислительные методы

Правильная ссылка на статью:

Хейфиц А.Е., Янчус В.Э., Борович Е.В. Методика проведения экспериментального исследования по восприятию визуальной информации в области периферийного зрения человека // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.44101 EDN: JBYQNP URL: https://nbpublish.com/library_read_article.php?id=44101

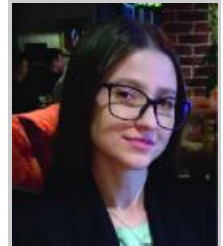
Методика проведения экспериментального исследования по восприятию визуальной информации в области периферийного зрения человека

Хейфиц Антонина Евгеньевна

ассистент, Высшая школа дизайна и архитектуры, Санкт-Петербургский политехнический университет
Петра Великого

195251, Россия, Ленинградская область, г. Санкт-Петербург, ул. Политехническая, 29

✉ antoni.t-h@mail.ru



Янчус Виктор Эдмундасович

кандидат технических наук

доцент, Высшая школа дизайна и архитектуры, Санкт-Петербургский политехнический университет
Петра Великого

195251, Россия, Ленинградская область, г. Санкт-Петербург, ул. Политехническая, 29

✉ victorimop@mail.ru



Борович Екатерина Владиславовна

ассистент, Высшая школа дизайна и архитектуры, Санкт-Петербургский политехнический университет
Петра Великого

195251, Россия, Ленинградская область, г. Санкт-Петербург, ул. Политехническая, 29

✉ plasma5210@mail.ru



[Статья из рубрики "Компьютерная графика, обработка изображений и распознавание образов"](#)

DOI:

10.7256/2454-0714.2024.4.44101

EDN:

JBYQNP

Дата направления статьи в редакцию:

22-09-2023

Аннотация: В данной статье описана методика проведения эксперимента в рамках исследования восприятия визуальной информации в области периферийного зрения человека. Авторами описана методика подготовки и проектирования стимульного материала, экспериментальная установка, методика проведения эксперимента и обработка полученных данных. В ходе проектирования эксперимента ставится вопрос расширения зоны чтения визуальной информации при работе с компьютерной системой. Поскольку анализ поступающей информации производился с постепенным усложнением задачи, разработка возможного решения в ходе работы стало производственной необходимостью. Любой монитор, вне зависимости от размера ограничивает анализ зрительного восприятия. Некорректное увеличение или уменьшение объектов в формате дисплея приводит к ухудшению данных. Корректное изображение снижает уровень усталости при чтении информации, повышает качество ее восприятия. Зона периферийного зрения остается незатронутой в рабочих процессах, являясь важной частью зрения человека. Исследование восприятия графических изображений в периферийной области человеческого зрения потенциально расширит эффективность работы интерфейса. Результаты, полученные на основе экспериментальных данных рационально использовать при разработке интерфейсов человеко-компьютерного взаимодействия. Методика включает в себя разработку и рассмотрение факторов цвета, размера и удаленности в стимульном материале. Стимульный материал спроектирован с помощью программного модуля в дальнейшем отвечающего за случайное и независимое расположение стимульного материала. Для фиксации параметров шаблона рассматривания используется программно-аппаратный комплекс ай-трекинга.

Ключевые слова:

Визуальное восприятие, Область периферийного зрения, Графический интерфейс, Ай-Трекинг, Стимульный материал, Статистический анализ, Интеллектуальный интерфейс, Человеко-компьютерное взаимодействие, Ай - трекер, Зрительная система

1. Введение

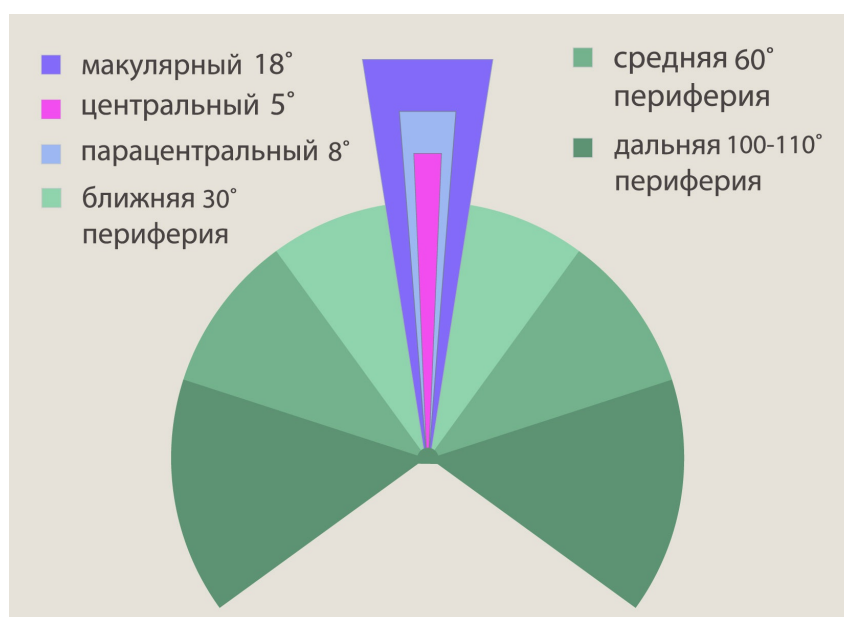
Прогресс технологических средств обработки и визуализации информации позволяет автоматизировать рабочие места операторов, контролирующих состояние сложных технических систем с динамически меняющимся состоянием. При управлении динамической системой в реальном времени оператор вынужден длительное время наблюдать за массивом одновременно меняющихся параметров на экранах мониторов, что приводит к высокой степени утомляемости и высоким психическим нагрузкам [1]. Компьютерные средства позволяют систематизировать динамическую информацию о наблюдаемом объекте и представлять ее в графическом виде и в виде анимационных диаграмм. В отличие от текстового представления данных, графический способ визуализации с помощью средств инфографики предлагает более эффективный подход к анализу оператором многочисленных параметрических данных о состоянии объекта [2].

Однако, применение компьютерных средств в создании систем управления ведет к тому, что работа оператора из реальной среды переходит в среду виртуальную со своим интерфейсом и правилами работы. Написано много научного материала о принципах взаимодействия оператора с компьютерной системой, вводится понятие иммерсивной среды [3], предлагаются принципиально новые методы анализа этого взаимодействия.

Но, следует отметить, что сфера взаимодействия оператора от пространства в 360 градусов сужается до размеров монитора (или нескольких мониторов) и, соответственно, происходит пространственное сжатие информации в соответствующей пропорции. Похожая ситуация происходит при переходе действия со сцены театра на экран кинотеатра. Чтение информации с экрана монитора происходит, в основном, за счет центрального зрения. Вопрос расширения зоны чтения визуальной информации при работе с компьютерной системой является малоизученным и требует проведения исследований.

2. Теоретическая модель

Рассмотрим систему считывания информации человеческим глазом. Человеческий глаз способен успешно считывать информацию, которая находится в пределах видимости центральной ямки, или фовеи (2 угловых градуса зрительного поля), так как именно в этой области достигается максимальная четкость изображения. Немного хуже воспринимается информация в парафовеи (5° по обе стороны от точки фиксации), а все, что дальше 5° (периферия), воспринимается плохо (рис. 1). Для языков с написанием слева и алфавитом это означает, что мы можем извлечь информацию из трех-четырех букв слева от фиксации и 14-15 знаков (включая пробелы) справа от фиксации [\[4,5\]](#). Восприятие информации происходит во время фиксации (медленного движения центральной точки зрения) (рис.2), которые составляют от 150 до 350 миллисекунд и зависят от решаемой наблюдателем задачи при восприятии информации.



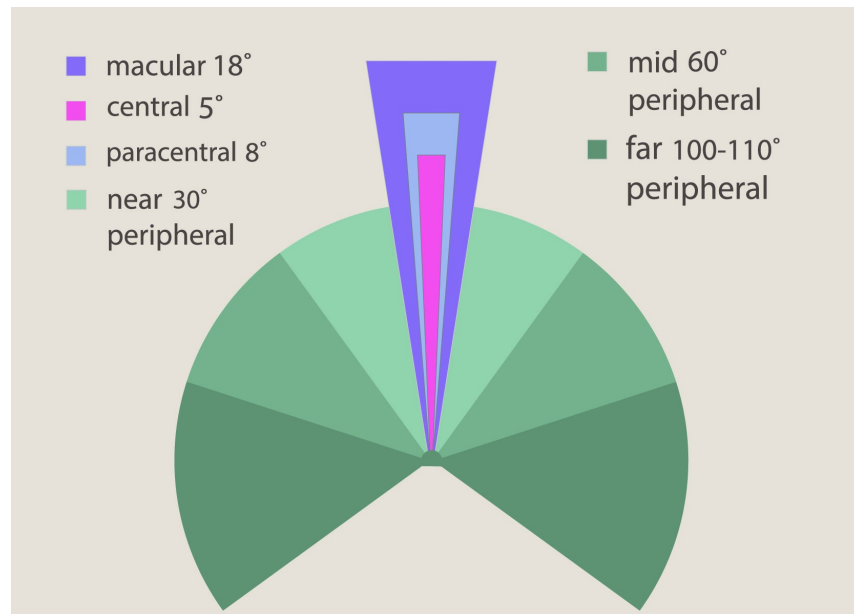


Рис. 1. Зоны зрения человеческого глаза

Однако, возможности периферийного зрения до конца не исследованы. Опыт спортсменов игровых видов спорта, показывает, что в результате тренировок у них развивается способность воспринимать периферийным зрением значительно большие объемы информации в сравнении с нетренированным человеком.

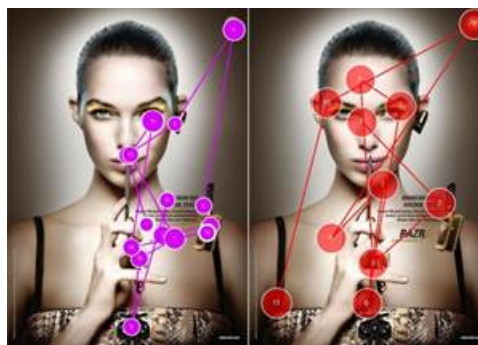


Рис. 2. Пример траектории движения взгляда

При восприятии текстовой информации задействована центральная область зрения, где изображение наиболее резкое [6]. В качестве гипотезы исследования принимается следующее утверждение: графические лаконичные изображения, представляющие собой цветные пятна определенного размера и воспринимаемые человеком как единый образ, в отличие от текста, могут восприниматься в периферийной области человеческого зрения. Исследование данного вопроса потенциально расширит эффективность работы интерфейса и определит информацию, которая может фиксироваться периферийной областью зрения (рис.1).

3. Экспериментальная установка

Исходя из технических возможностей программно-аппаратного комплекса SMIREД 250 [7], мы определяем ближнюю периферийную зону зрения как объект исследования. В экспериментальной установке расстояние от испытуемого до монитора (рис. 3) составляет 60 см при размерах монитора 64 на 40 см. Выбранные размеры и расстояния определяют максимальный угол обзора испытуемого, который составляет плюс-минус 30°, он перекрывает ближнюю периферийную зону зрения (см. рис.1).

В разрабатываемом эксперименте исследуется влияние трех факторов: цвет, размер и удаленность объекта от центра стимула. В соответствии с выбранными факторами разрабатывался стимульный материал эксперимента:

- выбор цвета выполнен в соответствии с теорией Геринга – чёрно-белый канал и цветные каналы: красно-зеленый, жёлто-синий [8];
- выбраны удобочитаемые за счет формы пиктограммы/иконки для максимального удобства оператора [9];
- удаленность от точки первичной фокусировки зрения в стимулах составила $\pm 15^\circ$, $\pm 20^\circ$, $\pm 25^\circ$, что соответствует расстоянию в 300 px, 570 px, 840 px от центра монитора.

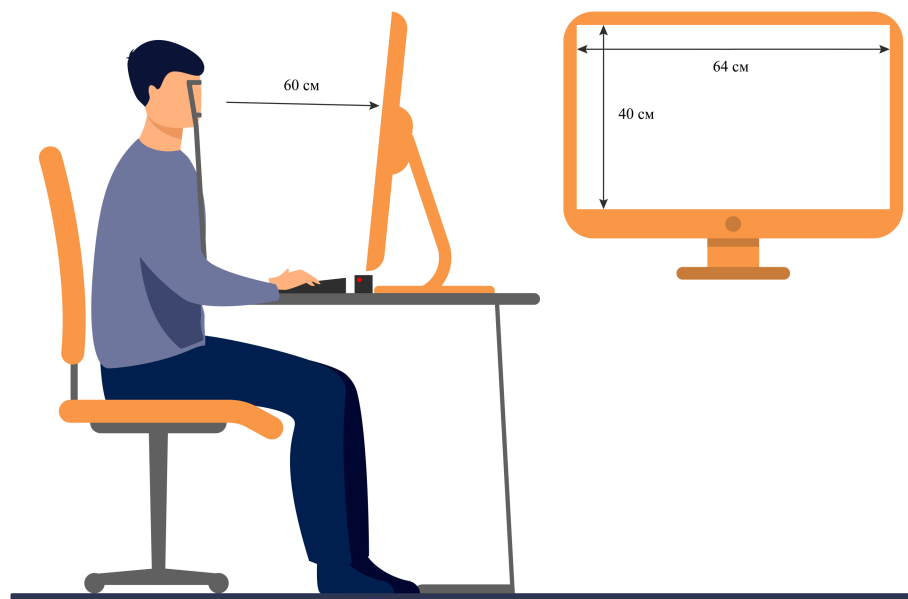


Рис. 3.Схема расположения экрана испытуемого в эксперименте

Для заполнения стимулов пиктограммами случайным образом из созданного набора был использован программный модуль, специально разработанный на языке процессинг. Всего было подготовлено 54 стимула.

Для создания стимульного материала было необходимо создать набор изображений, содержащих по 6 иконок расположенных по кругу и равноудалённых от иконки, расположенной в центре. Была создана база из 12-ти пиктограмм (рис. 4).

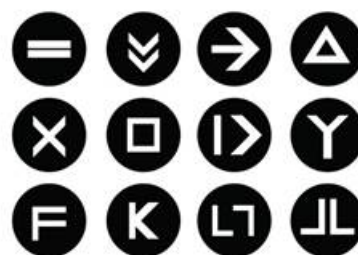


Рис. 4. Набор пиктограмм используемых в эксперименте

В рамках данного эксперимента иконки на разных стимулах должны быть выполнены в различных **цветах**: красный «r» (ED1E2E), зелёный «g» (019C59), синий «b» (1C68B1),

оранжевый «о» (F68522), чёрный «к» (000000), светло-серый«w» (E0E7F5), фон стимулов серый (B3B5B5). Кроме этого, иконки имеют разный**размер**: большой «big» 93px, средний «normal» 66px, маленький «little» 42px.**Удалённость от центра** принимает три значения: Closenearly (Low) – радиус 300px, Middle – радиус 570px, Farther (high) – радиус 840px (рис.5). Пример стимульного материала изображен на рисунке 6.

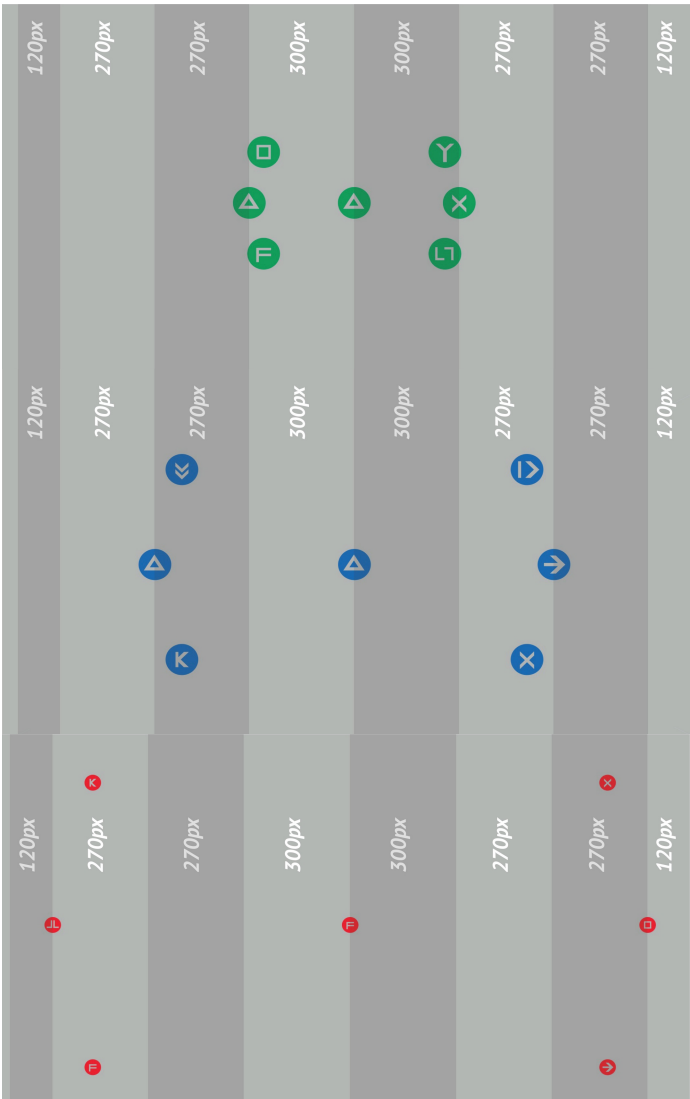


Рис. 5.Удаленность пиктограмм от центра стимула. Размер стимулов составляет 1920*1080px



Рис.6. Пример стимульного материала

Далее Алгоритм для создания стимулов с заданным значением радиуса удалённости иконок от центра представлен на рисунке 6.

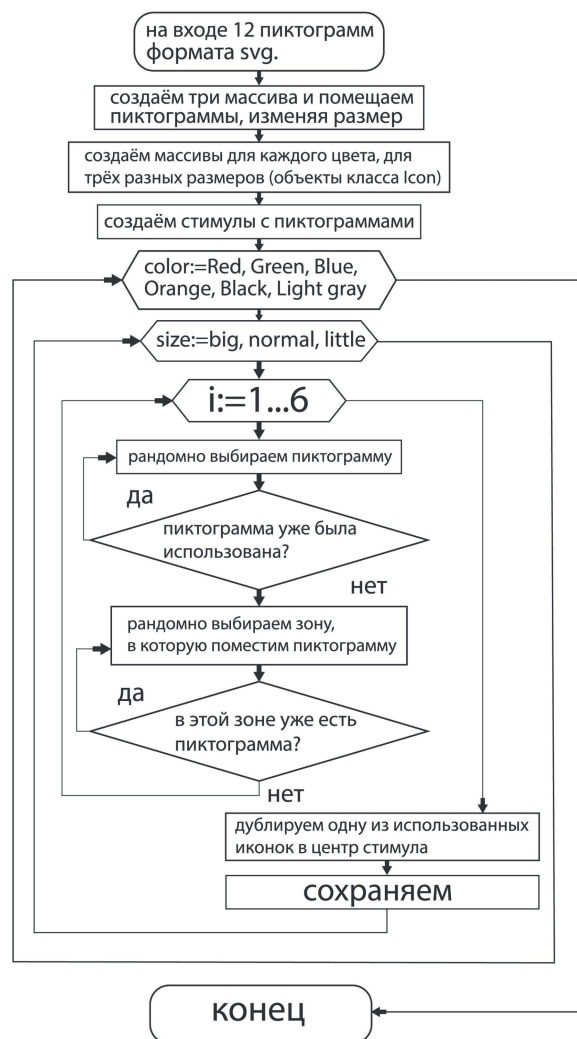


Рис. 6. Блок-схема алгоритма создания стимульного материала

На входе имеем базу из векторных файлов с пиктограммами типа svg,

создаём три массива иконок класса PShape, в которых задаём разный размер с помощью функции scale (1) для большого размера иконок, scale (0.7) - для среднего и scale (0.5) - для маленького.

Далее создадим новый класс объектов, который назовём Icon, чтобы хранить, помимо самого изображения, информацию о том, была ли пиктограмма уже использована в стимуле информации о зоне, в которой она будет помещена. Создаём массивы иконок для каждого размера и цвета (всего 6 цветов). Всего создано 18 массивов. Для расположения иконок существует шесть зон, равноудалённых от центра.

Перед испытуемым в эксперименте ставилась следующая задача: посмотреть в центр стимула, запомнить находящуюся там пиктограмму, затем, найти идентичную пиктограмму на окружности вне центра стимула и отметить ее наведением курсора мыши и нажатием кнопки. Координаты нажатия кнопки компьютерной мыши фиксируется в базе эксперимента, после чего происходит переход к следующему стимулу.

4. Результаты эксперимента

В эксперименте участвовало 22 человека от 18 до 25 лет из числа студентов Санкт-Петербургского политехнического университета Петра Великого. Было собрано 7930 фиксаций и 10720 саккад. Статистическая обработка результатов эксперимента

производилась посредством дисперсионного анализа ANOVA [10]. Анализировались следующие параметры шаблона рассматривания: время наблюдения стимула, средняя длительность фиксаций, средняя длительность фиксаций при наблюдении одного стимула, среднее количество фиксаций при наблюдении одного стимула, среднее время саккад и среднее количество саккад при наблюдении одного стимула. Задачей анализа было выявить влияние факторов размера пиктограмм, их удаленности от центра и цвета диаграмм, а также факторов гендера, наличия/отсутствия художественной подготовки и типа образования (гуманитарное - техническое). Значение критерия значимости p-value для принятия гипотезы было выбрано 0,05. Значения p-value, полученные в результате выполнения вычислительной процедуры представлены, в таблице 1.

Таблица.1. Вычисленные значения p-value. Красным выделены значения, позволяющие принять гипотезу о влиянии фактора.

фактор	p-value
Цвет	0.000003
Размер	0.000001
Удалённость от центра	0.000001
Гендер	0.000352
Образование	0.001502
Художественная подготовка	0.110483

График плотности распределения времени рассматривания стимула в зависимости от факторов цвета, размера, удаленности от центра, наличия/отсутствия художественной подготовки и гендерного признака представлены на рисунках 7,8.9.

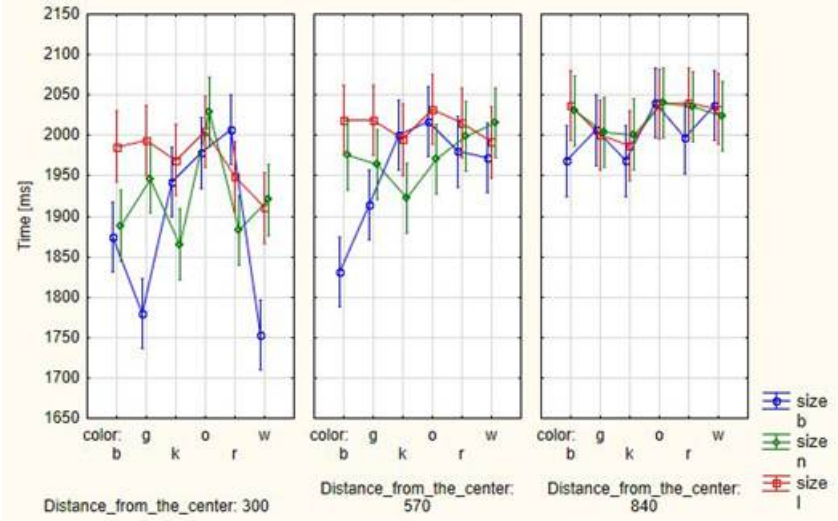


Рис.7. Графики плотности распределения времени рассматривания стимула в зависимости от факторов цвета (г – красный, g – зелёный, b – синий, о – оранжевый, k – чёрный, w – светло-серый), размера и удаленности

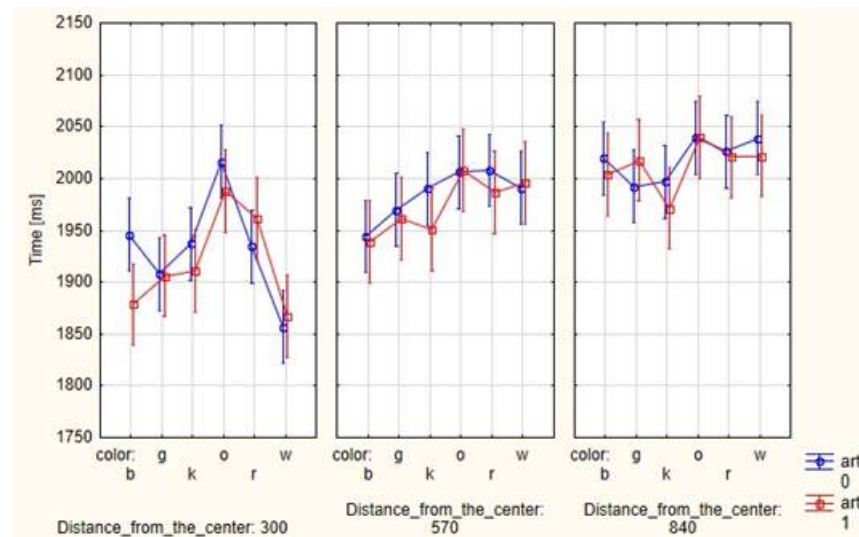


Рис. 8. График плотности распределения времени рассматривания стимула в зависимости от факторов наличия художественной подготовки (0 – отсутствует подготовка, 1 – наличествует художественная подготовка), цвета и удаленности от центра

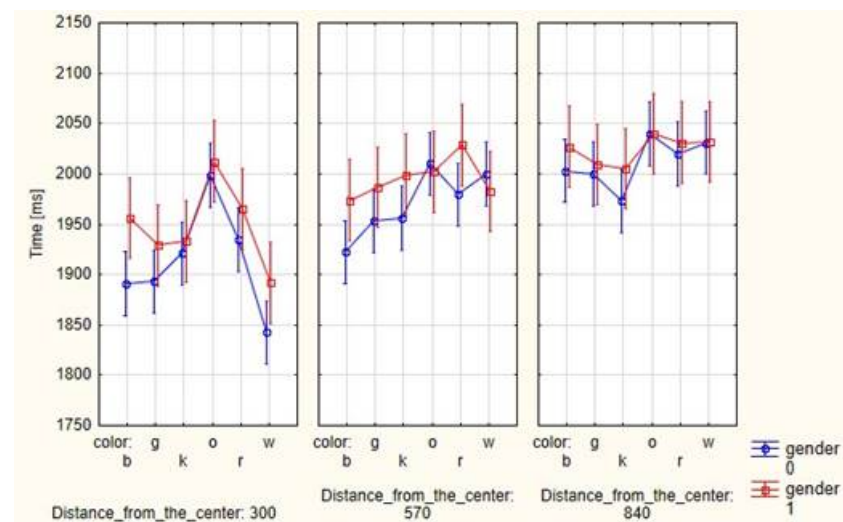


Рис. 9. График плотности распределения времени рассматривания стимула в зависимости гендерного признака (0 - женщины, 1 - мужчины), цвета и удаленности от центра

По результатам статистического анализа можно сделать несколько наблюдений:

- 1) Параметры шаблона рассматривания имеют статистически значимую зависимость от факторов цвета, размера, удаленности объекта от центра стимула.
- 2) Параметры шаблона рассматривания имеют статистически значимую зависимость от факторов типа образования (техническое - гуманитарное) и гендерного признака.
- 3) Параметры шаблона рассматривания не имеют статистически значимой зависимости от фактора наличия художественной подготовки.
- 4) Вышеперечисленные корреляции параметров шаблона рассматривания и изменяемых в эксперименте факторов зависят от решаемой испытуемым задачи.

Следует отметить, что для проведения более глубокого анализа вопроса восприятия визуальной информации человеком в области периферийного зрения необходимо увеличить количество испытуемых.

5. Выводы

В результате проделанной работы разработана методика проведения экспериментальных исследований восприятия визуальной информации человеком в области периферийного зрения с применением программно-аппаратного комплекса ай-трекинга.

Разработанная методика была апробирована на 23 испытуемых из числа студентов Санкт-Петербургского политехнического университета Петра Великого и показала свою состоятельность.

Разработанная методика может быть использована при проведении исследований по восприятию визуальной информации в графических интерфейсах систем удаленного управления динамическими объектами.

Библиография

1. Емельянова Ю. Г., Фраленко В. П. Методы когнитивно-графического представления информации для эффективного мониторинга сложных технических систем // Программные системы: теория и приложения. 2018. № 4. С. 117-158.
2. Когнитивная машинная графика в системах космического и медицинского назначения / Под общей редакцией доктора технических наук, профессора М. Н. Бурдаева.: ЛЕНАНД; Москва, 2018.
3. Сергеев С. Ф. Интеллектуальные симбионты организованных техногенных сред в управлении подвижными объектами // Мехатроника, автоматизация, управление. 2013. № 9. С. 30-36.
4. Liu Y., Zhou Z., Hu D. Gaze independent brain-computer speller with covert visual search tasks // Clinical Neurophysiology. 2011. № 122. Pp. 1127-1136.
5. McDonald S.A., Carpenter R. H., Shillcock R.C. An anatomically constrained, stochastic model of eye movement control in reading // Psychological review. 2005. № 112. Pp. 814-840.
6. Vitu F., McConkie G.W., Kerr P., O'Regan J.K. Fixation location effects on fixation durations during reading: An inverted optimal viewing position effect // Vision Res. 2001. № 41. Pp. 1513-1533.
7. Yarbus A. L. Eye Movements and Vision. Plenum Press, 1967. DOI: 10.1007/978-1-4899-5379-7
8. Юрьев Ф. И. Цветовая образность информации. Киев: Гармония сфер, 2007.
9. Барабанщиков В. А., Милад М. М. Методы окулографии в исследовании познавательных процессов и деятельности / В. А. Барабанщиков, М. М. Милад. Рос. АН. Ин-т психологии, Регион. ун-т непрерыв. образования. – М.: ИПРАН, 1994 – 88 с.
10. Гланц С. Медико-биологическая статистика / Гланц С.; Пер. с англ. д.ф.-м.н. Ю.А. Данилова под ред. Н.Е. Бузикашвили и Д.В. Самойлова. – М.: Практика, 1999 – 459 с.

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Статья «Методика проведения экспериментального исследования по восприятию визуальной информации в области периферийного зрения человека», поданная в журнал «Программные системы и вычислительные методы» является полноценным научным исследованием, выполненным на должном методологическом уровне и имеющем явное практическое применение.

Данная работа посвящена разработке методики проведения экспериментальных исследований восприятия визуальной информации человеком в области периферийного зрения с применением программно-аппаратного комплекса ай-трекинга.

Обсуждаемая статья весьма актуальна, так как описывает работу, направленную на автоматизацию рабочих мест операторов, контролирующих состояние сложных технических систем с динамически меняющимся состоянием, позволяя отойти от классического представления информации на экране монитора в виде многочисленных текстовых или числовых данных касательно наблюдаемых оператором параметров к представлению её в графическом виде и в виде анимационных диаграмм. Применительно к такому виду представления данных автор(ы) обсуждаемой статьи предлагает(ют) полагаться не только на узкий диапазон «центрального» зрения, составляющий не более 15 градусов, а задействовать периферическое зрение, эффективность восприятия которого можно повысить упражнениями.

При разработке повышения уровня восприятия визуальной информации человеком в области периферийного зрения авторы статьи руководствуются гипотезой, согласно которой графические лаконичные изображения, представляющие собой цветовые пятна определенного размера и воспринимаемые человеком как единый образ, в отличие от текста, могут восприниматься в периферийной области человеческого зрения.

Экспериментальное исследование было проведено авторами статьи на разработанной экспериментальной установке с оригинальным программным обеспечением, алгоритм работы которого в достаточной для понимания и воспроизведения эксперимента степени представлен в работе.

По результатам исследования авторы сделали следующие выводы:

- 1) Параметры шаблона рассматривания имеют статистически значимую зависимость от факторов цвета, размера, удаленности объекта от центра стимула.
- 2) Параметры шаблона рассматривания имеют статистически значимую зависимость от факторов типа образования и гендерного признака.
- 3) Параметры шаблона рассматривания не имеют статистически значимой зависимости от фактора наличия художественной подготовки.
- 4) Вышеперечисленные корреляции параметров шаблона рассматривания и изменяемых в эксперименте факторов зависят от решаемой испытуемым задачи.

Разработанная авторами статьи методика может быть использована при проведении исследований по восприятию визуальной информации в графических интерфейсах систем удаленного управления динамическими объектами, однако, как абсолютно верно пишут сами авторы работы, необходимо её дальнейшая апробация на большем числе испытуемых. При этом, рекомендую авторам статьи в будущих исследованиях существенно расширить возрастную выборку, так как восприятие информации у молодёжи и людей предпенсионного возраста может существенно отличаться.

В целом, статья «Методика проведения экспериментального исследования по восприятию визуальной информации в области периферийного зрения человека» написана хорошим и понятным языком, библиографический список достаточен, адекватен тексту статьи и не избыточен. Особой похвалы заслуживает наглядный иллюстративный материал, позволяющий читателю легко работать с материалом статьи. Данные иллюстрации, помимо всего прочего, могут быть использованы при подготовке учебных материалов для курсов психологии и биофизики сенсорных систем.

Однако вынужден заметить, что в статье имеется некоторая неточность: в разделе «Результаты эксперимента» указано, что в эксперименте участвовало 22 человека, а в разделе «Выводы», что методика была апробирована на 23 испытуемых. Авторам обязательно следует внимательно проверить текст статьи и устранить подобные ошибки.

Решение: принять статью «Методика проведения экспериментального исследования по

восприятию визуальной информации в области периферийного зрения человека» в журнал «Программные системы и вычислительные методы» после устранения ошибок в тексте. Повторное рецензирование не требуется.

Программные системы и вычислительные методы

Правильная ссылка на статью:

Дамдинова Т.Ц., Дамдинов З.Ш., Прудова Л.Ю., Бубеев И.Т. Моделирование пор нерегулярной формы по яркости пикселей цифрового изображения // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.72014 EDN: JPDKMT URL: https://nbpublish.com/library_read_article.php?id=72014

Моделирование пор нерегулярной формы по яркости пикселей цифрового изображения

Дамдинова Татьяна Цыбиковна

ORCID: 0000-0002-3597-3262

кандидат технических наук

доцент; кафедра "Инженерная и компьютерная графика"; Восточно-Сибирский государственный университет технологий и управления

670000, Россия, республика Бурятия, г. Улан-Удэ, ул. Ключевская, 40 В

✉ dtatyanac@mail.ru



Дамдинов Зоригто Ширипович

ORCID: 0009-0002-1614-3746

магистр; кафедра «Программная инженерия и искусственный интеллект»; Восточно-Сибирский государственный университет технологий и управления

670013, Россия, республика Бурятия, г. Улан-Удэ, ул. Ключевская, 40 В

✉ vdzorigto@mail.ru



Прудова Людмила Юрьевна

кандидат технических наук

доцент; кафедра "Инженерная и компьютерная графика"; Восточно-Сибирский государственный университет технологий и управления

670013, Россия, Бурятия область, г. Улан-Удэ, ул. Ключевская, 40в

✉ prudova456@mail.ru



Бубеев Иннокентий Трофимович

кандидат технических наук

доцент; кафедра "Инженерная и компьютерная графика"; Восточно-Сибирский государственный университет технологий и управления

670013, Россия, республика Бурятия, г. Улан-Удэ, ул. Ключевская, 40в

✉ it_bubeev@mail.ru



[Статья из рубрики "Компьютерная графика, обработка изображений и распознавание образов"](#)

DOI:

10.7256/2454-0714.2024.4.72014

EDN:

JPDKMT

Дата направления статьи в редакцию:

17-10-2024

Аннотация: Одним из актуальных направлений в области материаловедения является исследование и моделирование процессов, происходящих в капиллярно-пористых телах. Эти объекты играют важную роль в таких областях, как нефтедобыча, медицина, строительство, производство фильтров и аккумуляторов, создание новых материалов, где требуется точное понимание и контроль процессов, происходящих в пористых средах. Современные методы моделирования с использованием компьютерного зрения и высокие вычислительные мощности компьютеров позволяют значительно повысить точность и эффективность исследований капиллярно-пористых тел и процессов в них. Для пористых тел нерегулярной структуры точная визуальная фиксация процессов, происходящих внутри объектов, затруднительна, но потребность в этом имеется особенно для многоуровневых процессов, которые влияют на состояние пористого тела. Предлагаемые симуляторы и модели структуры таких объектов с использованием упрощений и абстрактных моделей для оценки взаимодействия веществ (процессов диффузии, перколяции и т.д.) непригодны в неоднородных и нерегулярных структурах.

Для получения геометрической модели пор произвольной формы предлагается использовать методы обработки цифровых изображений. Создан массив точек, описывающий геометрию поры. Выбран способ поверхностного моделирования на основе сплайн-интерполяции выделенных точек. Для получения поверхностной модели поры нерегулярной формы разработана методика формирования массива точек по уровням яркостей пикселей его цифрового изображения. Уровни яркостей для группировки исходных точек поры определены на основе локальных максимумов гистограммы яркостей. Разработан алгоритм построения поверхностной модели поры нерегулярной формы методами сплайн-интерполяции массивов точек. Представлен результат программной реализации разработанного алгоритма моделирования поверхности пор нерегулярной формы. По разработанной методике геометрического моделирования пор нерегулярной формы можно получить дальнейшую информацию для анализа о размере и объеме пор, пористости объекта в целом, а также проводить анализ процессов в пористых объектах. При необходимости для повышения точности формы количество уровней яркости для группировки точек можно увеличить.

Ключевые слова:

пористые материалы, пористость, обработка цифровых изображений, сплайн-интерполяция, поверхностное моделирование, нерегулярная форма, геометрическое моделирование, массив точек, гистограмма, уровень яркости пикселей

Статья написана при поддержке гранта "Молодые ученые ВСГУТУ-2024"

Введение

Одним из актуальных направлений в области материаловедения является исследование

и моделирование процессов, происходящих в капиллярно-пористых телах. Эти объекты играют важную роль в таких областях, как нефтедобыча, медицина, строительство, производство фильтров и аккумуляторов, создание новых материалов, где требуется точное понимание и контроль процессов, происходящих в пористых средах.

Современные методы моделирования и существующие модели представления пористых объектов часто не учитывают нерегулярность структур, что приводит к снижению точности моделирования и ограничивает их применение. В связи с этим возникает необходимость в разработке новых методов и инструментов для моделирования процессов в нерегулярных структурах капиллярно-пористых тел.

Постановка задачи

Капиллярно-пористые тела – это материалы, характеризующиеся наличием множества пор, заполненных жидкостью или газом. Эти материалы широко применяются в различных областях, таких как медицина (с точки зрения проникновения медикаментозных средств в ткани), производство фильтров (для определений размеров пор, влияющих на качество фильтрации), создание новых строительных материалов с заданными свойствами (например, влагопроницаемость и теплоотдача), создание новых композиционных высокопрочных, легких материалов. Понимание и моделирование процессов, происходящих в пористых структурах, является важной задачей, поскольку это позволяет улучшить характеристики и эффективность данных материалов при их использовании [\[1\]](#).

Пористость материала определяется как соотношение объема пор к общему объему материала и является важной характеристикой, влияющей на его физические и механические свойства [\[2\]](#). Пористые системы могут значительно различаться по форме, размеру и распределению пор. По геометрическим признакам пористые тела подразделяются на регулярные пористые структуры с правильным чередованием в объеме тела отдельных пор или полостей и соединяющих их каналов, а также на структуры, со случайными формами, размерами, ориентацией и взаимным расположением. В зависимости от геометрических и топологических характеристик, поры можно классифицировать на несколько типов: открытые, закрытые, тупиковые и сквозные [\[3\]](#).

Методы исследования пористых структур рассмотрены в работе [\[4\]](#). Здесь представлена классификация пористой структуры материалов по происхождению и размерно-геометрическим признакам. Авторы подчеркивают, что «Известно более 60 аналитических методов исследования пористой структуры твёрдых тел, систематизированных по физическим принципам определения её характеристик». Нерегулярные структуры капиллярно-пористых тел имеют сложные геометрические формы, что затрудняет их анализ и моделирование процессов в них [\[5\]](#). Известны инструментальные методы определения пористости, которые являются трудоемкими и материалоемкими [\[6\]](#). Эти методы требуют значительных затрат времени и ресурсов для проведения экспериментов и последующей обработки данных. В результате, процесс исследования капиллярно-пористых тел становится крайне затратным, что ограничивает возможности для проведения обширных исследований. Основной проблемой является отсутствие универсальных методов, которые могли бы точно описывать и прогнозировать состояние таких материалов. Существующие модели часто основаны на упрощенных допущениях, что приводит к снижению точности моделирования и оценки для получения адекватного представления о способе или принципах работы исследуемых объектов.

Современные методы моделирования включают использование хорошо разработанных методов компьютерного зрения [7, 8], вычислительных методов для анализа изображений пористых структур и создания соответствующих моделей [9]. Решение современных научных задач, например, таких как создание новых материалов с необходимыми характеристиками, требуют более точных данных, которые можно получить на основе анализа их цифровых изображений. Методы обработки цифровых изображений позволяют автоматизировать процесс сбора данных и их обработки, что значительно сокращает время на проведение исследований.

Отметим, что традиционно при моделировании пор используются упрощенные модели, представляющие вместо них фигуры известных форм [9-11] – круги, сферы, квадраты, кубы и т.д., а структура пористого объекта задается в виде решеток, клеток и т.п. (рис.1, рис.2)

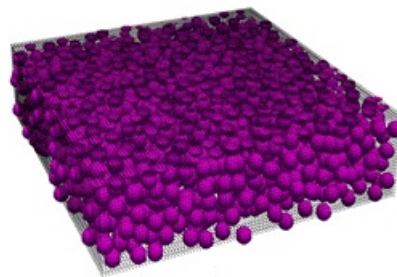


Рисунок 1 – Трехмерная модель пористого тела в сферической форме

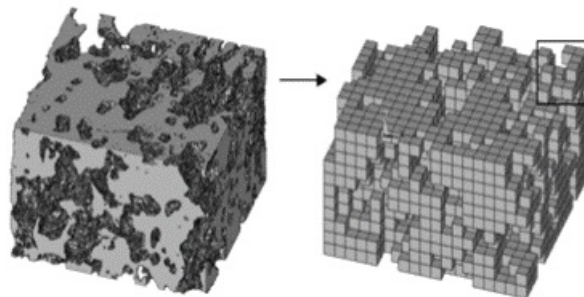


Рисунок 2 – Упрощенная модель пористого тела в виде вокселей

Основные направления современных методов моделирования пористых материалов можно представить на программах компьютерного моделирования - PoreSpy и OpenPNM. (рис.3, рис.4). Приложение PoreSpy описывает поры на плоскости, а OpenPNM – структуру пористого объекта в 3D.

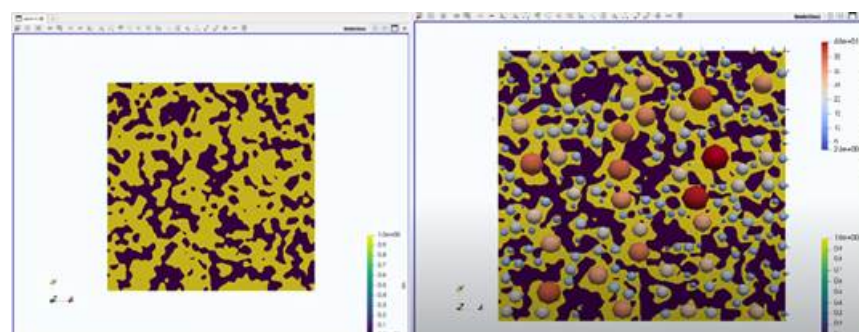


Рисунок 3 – Пример обработки пористого тела проектом PoreSpy

OpenPNM – проект с открытым исходным кодом, который также направлен на

моделирование пористых материалов. В отличие от PoreSpy, в OpenPNM пористая структура представляется в виде связанных сфер (рис.4). Такой подход является более реалистичным с точки зрения моделирования пористых материалов, так как сферы могут более точно соответствовать форме и размеру пор в пористых структурах. Однако такое представление также является абстрактным, поскольку сферы, представляющие поры связываются между собой прямыми линиями, что приводит к ограничениям в точности моделирования, особенно для пористых материалов нерегулярной формы.

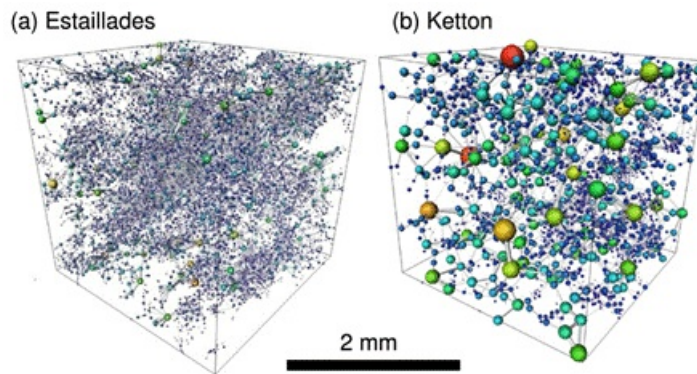


Рисунок 4 – Пример обработки пористого тела проектом OpenPNM

Алгоритм моделирования пор нерегулярной структуры

Программное решение моделирования пор нерегулярной структуры представляет собой ввод пользователем изображений срезов пористого тела, аналогично [\[12, 13\]](#). По обработанному изображению поры выделяются границы срезов на основе информации по яркости пикселей, с последующим построением 3D модели по выделенным граничным точкам срезов.

Для моделирования реальных процессов в первую очередь требуется получить наиболее приближенное представление о форме пористого объекта. Одним из основных способов при моделировании пористых объектов являются методы поверхностного моделирования для создания гладких и непрерывных поверхностей, что позволяет более точно описывать форму пор и их распределение в материале. Отметим, что аппарат поверхностного моделирования достаточно хорошо разработан [\[14, 15\]](#).

Для построения поверхностных моделей имеются различные методы, каждый из которых имеет свои особенности и области применения [\[16, 17\]](#). Метод с использованием аналитических моделей предполагает создание поверхностей на основе уравнений, описывающих известные геометрические формы - цилиндрические, канонические, сферические и эллиптические поверхности. Также поверхности могут быть построены на базе кривых. Кривые, полученные из поперечных сечений объекта, могут быть использованы для создания трехмерной модели путем метода экструзии или вращения. Поверхности могут быть также построены по существующим поверхностям, используя методы интерполяции и экстраполяции. Например, для создания модели поверхности из нескольких известных поверхностей можно использовать метод билинейной интерполяции, который позволяет плавно переходить между заданными поверхностями. Такой подход применяется для моделирования сложных объектов

Другим подходом к построению поверхностей является использование данных на базе массива точек. Массив точек для построения поверхности может быть обработан методами, который включают создание полигональных сеток, биномиальных

поверхностей, бикубических поверхностей Кунса, а также четырехугольных и треугольных поверхностей Безье [15]. На практике многие кривые и поверхности состояются из относительно простых гладких частей - отрезков (кривых) или фрагментов поверхностей, каждый из которых можно с достаточной точностью описать элементарной функцией одной или двух переменных. Эти методы подходят для решения задачи восстановления объектов неправильной формы, где в качестве исходных данных используется массив точек.

Результаты

Для решения задачи моделирования трехмерной модели поры нами был выбран, как и ранее [12], самый доступный пористый объект – сыр, с порами неправильной формы (рис.5). После этапа бинаризации исходного цифрового изображения среза выделены наиболее выраженные поры, из которых была выбрана одна крупная пора для создания его объемной модели.



Рисунок 5 – Цифровое изображение среза сыра с порами модель поры нерегулярной формы а – исходное изображение, b – поры, выделенные после бинаризации, с – пора для создания 3D-модели

По яркости пикселей выбранной поры можно предположить предполагаемый размер поры в глубину [18]. Для этого по полутоновому изображению поры построена гистограмма яркостей, на которой программно определены глобальные и локальные максимумы (рис.6).

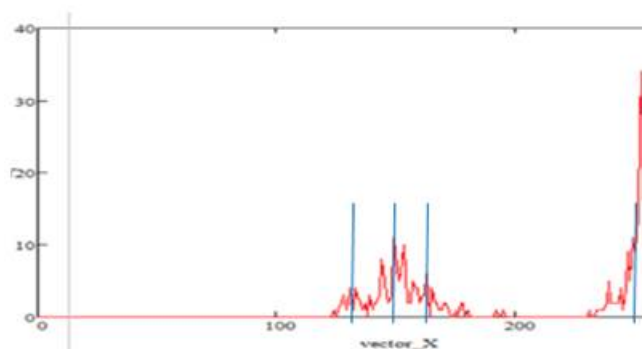


Рисунок 6 - Гистограмма яркости пикселей поры с максимумами

Из точек пор, сгруппированных на уровне этих яркостей, получены массивы граничных точек срезов поры (рис. 7).

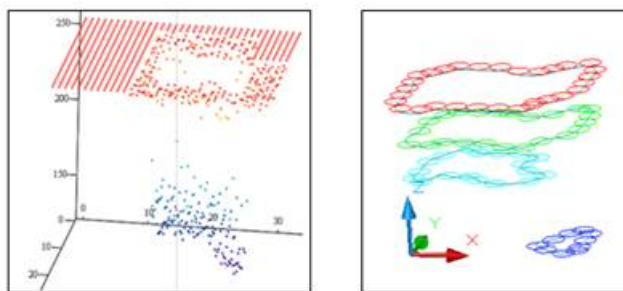
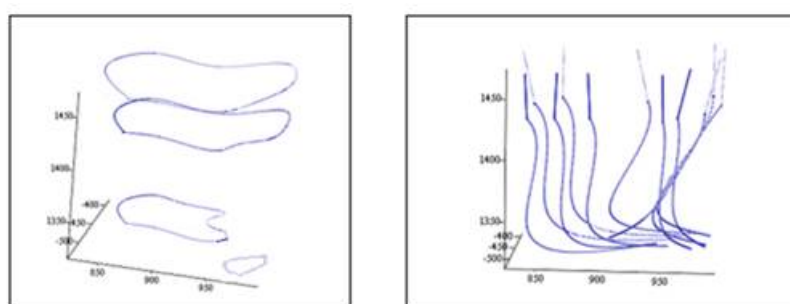


Рисунок 7 – Группировка точек на уровнях выбранных яркостей

Далее по каждому из массивов граничных точек была выполнена интерполяция кубическим сплайном, поперечные и продольные срезы которых представлены на рисунке 8. (Отметим, что для удобства визуального представления результата моделирования самая нижняя часть поры была пропущена.) Затем, на следующем шаге, на основе вычисленных сплайнов выполнено поверхностное моделирование (рис.9).



а б

Рисунок 8 – Сплайн-интерполяция граничных точек поры сыра

а – поперечные сечения, б – продольные сечения

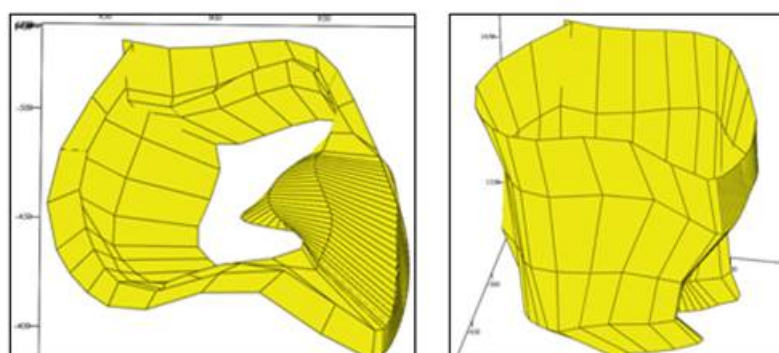


Рисунок 9 – Поверхностная модель поры нерегулярной формы

По разработанной методике геометрического моделирования пор нерегулярной формы можно получить дальнейшую информацию для анализа о размере и объеме пор, пористости объекта в целом, а также проводить анализ процессов в пористых объектах [19, 20]. При необходимости для повышения точности формы количество уровней яркости для группировки точек можно увеличить.

Заключение

Решение задачи по моделированию пор объектов с нерегулярной структурой показало, что вычислительные мощности современных компьютеров и современные методы

геометрического моделирования позволяют получить адекватную трехмерную модель поверхности поры реального объекта. Применение этого решения позволит с высокой степенью точности воспроизводить сложные геометрические структуры, характерные для пористых материалов, что позволит развивать данное направление дальше для анализа пористых тел с нерегулярной структурой и процессов в них.

Библиография

1. Лыков, А. В. Явления переноса в капиллярно-пористых телах / А. В. Лыков. – М.: Государственное издательство технико-теоретической литературы, 1954. – 296 с.
2. А.В. Медведева, Д.М. Мордасов, М.М. Мордасов. Классификация методов контроля пористости материалов // Вестник ТГТУ. 2012. Том 18. № 3.
3. Беркман, А. С. Структура и морозостойкость стеновых материалов / А. С. Беркман, И. Г. Мельникова. – М.: Госстройиздат, 1962. – 166 с.
4. Фандеев, В. П. Методы исследования пористых структур // Интернет-журнал «НАУКОВЕДЕНИЕ». – 2015. – Т. 7, № 4.
5. Хейфец, Л. И. Многофазные процессы в пористых телах / Л. И. Хейфец, А. В. Неймарк. – М.: Химия, 1982. – 320 с.
6. А. Г. Анисович, А. С. Буйницкая. Стандартные методы определения пористости материалов (обзор). // Весці нацыянальнай акадэміі навук Беларусі, 2015. № 2. URL: <https://vestift.belnauka.by/jour/article/view/103/104>
7. Гонсалес, Р. Цифровая обработка изображений в среде MATLAB / Р. Гонсалес, Р. Вудс, С. Эддинс. – М.: Техносфера, 2006. – 616 с.
8. Сулейманов К.А., Лесовик В.С., Погорелова И.А., Рябчевский И.С. Исследование макропористой структуры ячеистого бетона //Вестник БГТУ им. В.Г. Шухова 2024, № 3. DOI: 10.34031/2071-7318-2024-9-3-8-16.
9. Тарасевич, Ю. Ю. Перколяция: Теория, приложения, алгоритмы / Ю. Ю. Тарасевич. – М.: Едиториал УРСС, 2002.
10. Левандовский, А. Н. Моделирование разрушения пористого материала / А. Н. Левандовский, Б. Е. Мельников, А. А. Шамкин // Инженерно-строительный журнал. – 2017. – № 1 (69). – С. 3-22.
11. Алексеев, М. В. О расчете исходных данных для моделирования радиационно-индуцированных эффектов в материалах пористого типа / М. В. Алексеев [и др.] // Препринты ИПМ им. М. В. Келдыша. – 2018. – № 208. – 21 с. – DOI: 10.20948/prepr-2018-208.
12. Дамдинова Т.Ц., Аюшеев Т.В., Бальжинимаева С.М., Абатнин А.А. Моделирование тел со сферическими порами методом обобщенной линейной интерполяции // Программные системы и вычислительные методы. 2022. № 2. С. 42-51. DOI: 10.7256/2454-0714.2022.2.38262 EDN: ZTFTKU URL: https://e-notabene.ru/itmag/article_38262.html
13. Бальжинимаева С.М., Дамдинова Т.Ц., Аюшеев Т.В., – Моделирование тел со сферическими порами методом обобщенной линейной интерполяции. // Свидетельство об официальной регистрации программы для ЭВМ № 2022663716 от 19.07.2022. Заявка №2022662918 от 08.07.2022 г.
14. Голованов, Н. Н. Геометрическое моделирование / Н. Н. Голованов. – М.: Издательство физико-математической литературы, 2002. – 472 с.
15. Фокс, А. Вычислительная геометрия. Применение в проектировании и на производстве / А. Фокс, М. Пратт; пер. с англ. – М.: Мир, 1982. – 304 с.
16. Поверхностное моделирование: основы, применение и преимущества // Научные Статьи.Ру – портал для студентов и аспирантов. – URL: <https://nauchniestati.ru/spravka/poverhnostnoe-modelirovanie>
17. Косников, Ю. Н. Поверхностные модели в системах трехмерной компьютерной

графики: учебное пособие / Ю. Н. Косников. – Пенза: Пензенский государственный университет, 2007. – 60 с.

18. Грузман, И. С. Цифровая обработка изображений в информационных системах: учебное пособие / И. С. Грузман, В. С. Киричук, В. П. Косых, Г. И. Перетягин, А. А. Спектор. – Новосибирск: Издательство НГТУ, 2000. – 168 с.

19. Дамдинова Т.Ц., Мосоров В.И., Дамдинов З.Ш., Коробков К.С.. Исследование параметров диффузионных слоев в процессах химико-термической обработки металлов по цифровым изображениям микроструктур. //Свидетельство об официальной регистрации программы для ЭВМ №2024613959 опубл.19 февраля 2024 г. Заявка № 2024612459 от 09 февраля 2024 г.

20. Дамдинова Т.Ц., Раднаева В.Д., Шалбуев Д.В., Советкин Н.В., Дерябин М.А. Способ визуализации процесса пропитки капиллярно-пористых объектов //Пат. №2817553 опубл.16/04/2024 заявка №2023126423/28.

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

Представленная на рецензирование статья посвящена моделированию пор нерегулярной формы в материалах, характеризующихся наличием множества пор, заполненных жидкостью или газом, по яркости пикселей цифрового изображения.

Методология исследования базируется на изучении и обобщении литературных источников по теме работы, применении графики, методов обработки изображений и компьютерного информационного моделирования.

Актуальность работы авторы статьи связывают с тем, что процессы, происходящие в капиллярно-пористых телах играют важную роль в таких областях, как нефтедобыча, медицина, строительство, производство фильтров и аккумуляторов, создание новых материалов, а современные методы и модели пористых объектов часто не учитывают нерегулярность структур, что приводит к снижению точности моделирования и ограничивают их применение, поэтому ощущается необходимость разработки новых методов моделей процессов в нерегулярных структурах капиллярно-пористых тел.

Научная новизна рецензируемого исследования, по мнению рецензента заключается в выводах о том, что что вычислительные мощности современных компьютеров и современные методы геометрического моделирования позволяют получить адекватную трехмерную модель поверхности поры реального объекта и с высокой степенью точности воспроизводить сложные геометрические структуры пористых материалов.

В статье структурно выделены следующие разделы: Введение, Постановка задачи, Алгоритм моделирования пор нерегулярной структуры, Результаты, Заключение и Библиография.

Авторы, ссылаясь на опубликованные ранее работы, говорят о наличии более чем 60 аналитических методов исследования пористой структуры твёрдых тел, систематизированных по физическим принципам определения её характеристик. Сложные геометрические формы нерегулярных структур капиллярно-пористых тел затрудняют их анализ и моделирование происходящих в них процессов. Авторы считают, что для решения современных научных задач, таких как создание новых материалов с необходимыми характеристиками, необходимы точные данные, которые можно получить на основе анализа их цифровых изображений, такой подход значительно сокращает время на проведение исследований. В публикации рассмотрены традиционные для моделирования пор упрощенные модели, в которых в качестве образов пор

используются фигуры известных форм – круги, сферы, квадраты, кубы, а структура пористого объекта задается в виде решеток, клеток и т.п. Изложение сопровождается наглядными визуализациями. Современные методы моделирования пористых материалов рассмотрены в статье на примере программ компьютерного моделирования PoreSpy и OpenPNM, в которых на основе изображений срезов пористого тела по обработанному изображению выделяются границы срезов на основе информации по яркости пикселей с последующим построением 3D модели по выделенным граничным точкам срезов. Рассмотрение подходов к построению поверхностных моделей сопровождается примером решения задачи моделирования трехмерной модели поры самого доступного пористого объекта – сыра с порами неправильной формы, по яркости пикселей по полутоновому изображению поры построена гистограмма яркостей, программно определены глобальные и локальные максимумы, по каждому из массивов граничных точек выполнена интерполяция кубическим сплайном.

Библиографический список включает 20 источников – публикации отечественных ученых по теме статьи, а также патенты и программы для ЭВМ. В тексте имеются адресные ссылки на литературные источники, подтверждающие наличие апелляции к оппонентам. Рецензируемый материал соответствует направлению журнала «Программные системы и вычислительные методы», подготовлен на актуальную тему, содержит теоретические обобщения, элементы научной новизны и практической значимости. Статья рекомендуется к опубликованию.

Программные системы и вычислительные методы

Правильная ссылка на статью:

Золотухина Д.Ю. Эффективность распределённых кэширующих платформ в современных backend-архитектурах: сравнительный анализ Redis и Hazelcast // Программные системы и вычислительные методы. 2024. № 4. DOI: 10.7256/2454-0714.2024.4.72305 EDN: JNJVQQ URL: https://nbpublish.com/library_read_article.php?id=72305

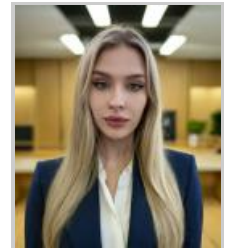
Эффективность распределённых кэширующих платформ в современных backend-архитектурах: сравнительный анализ Redis и Hazelcast

Золотухина Дарья Юрьевна

независимый исследователь

394062, Россия, Воронежская область, г. Воронеж, пер. Антокольского, 4

✉ dar.zolott@gmail.com



[Статья из рубрики "Математическое и программное обеспечение новых информационных технологий"](#)

DOI:

10.7256/2454-0714.2024.4.72305

EDN:

JNJVQQ

Дата направления статьи в редакцию:

11-11-2024

Аннотация: Объектом исследования являются две системы кэширования и распределенного хранения данных — Redis и Hazelcast, которые широко применяются для ускорения доступа к данным в высоконагруженных приложениях. В статье проводится всестороннее сравнительное исследование этих систем по ключевым аспектам, важным для эффективной работы с кэшированием: архитектурным особенностям, моделям управления памятью, подходам к кластеризации, механизмам отказоустойчивости и масштабируемости. Особое внимание уделяется исследованию возможностей работы с шаблонами кэширования и поддержке SQL-подобных запросов. Цель работы заключается в глубоком анализе преимуществ и ограничений Redis и Hazelcast в контексте кэширования данных, а также в выявлении их сильных и слабых сторон при различных нагрузках и сценариях эксплуатации. Методология исследования включает сравнительный анализ Redis и Hazelcast по ключевым аспектам, с последующим представлением результатов в виде сравнительной таблицы. Также было

проведено тестирование эффективности выполнения операций CRUD с использованием автоматизированных тестов, интегрированных в программу на платформе Spring Boot. Проведенное исследование показывает, что Redis, будучи однопоточной системой с быстрыми операциями записи и чтения, эффективен для простых и локализованных приложений, в то время как Hazelcast, поддерживающий многопоточность и динамическую кластеризацию, более эффективно справляется с большими объемами данных и распределенными задачами. Особым вкладом автора в исследование темы является комплексный сравнительный анализ этих систем с учетом их ключевых характеристик, таких как производительность, масштабируемость и отказоустойчивость, а также тестирование их работы в реальных сценариях. Новизна исследования заключается в детальном анализе применения Redis и Hazelcast для кэширования данных в высоконагруженных приложениях, что будет полезно для разработки и оптимизации инфраструктуры высокопроизводительных распределенных систем, которые требуют кэширования данных в реальном времени.

Ключевые слова:

кэширование, Redis, Hazelcast, производительность, CRUD операции, отказоустойчивость, многопоточность, кластеризация, in-memory хранение данных, распределенная система

Введение

С увеличением объемов данных и требований к быстродействию систем кэширование становится ключевым механизмом, который позволяет ускорить доступ к данным, снизив нагрузку на основные базы данных. В современном мире распределенных систем важно иметь надежное и производительное решение для кэширования, которое бы отвечало запросам как малых, так и крупных приложений. Среди множества доступных решений для кэширования два лидера рынка — Redis и Hazelcast — предлагают разнообразные возможности для обработки больших объемов данных в оперативной памяти, однако различаются по архитектуре и подходам к распределенной работе с данными.

Обзор Redis и Hazelcast

Redis — это одна из ведущих in-memory систем для кэширования, которая позволяет значительно ускорить доступ к данным за счет их хранения в оперативной памяти. Это решение стало стандартом для множества приложений, требующих высокой скорости и производительности, так как Redis обеспечивает низкие задержки и высокую пропускную способность. Благодаря своим характеристикам, Redis применяется во множестве задач, от хранения пользовательских сессий до кэширования результатов сложных вычислений и временных данных, требующих мгновенного доступа [\[1\]](#).

Популярность Redis обусловлена его универсальностью и гибкостью в настройке, что делает его подходящим для высоконагруженных систем, которым важно быстро обрабатывать огромное количество запросов в реальном времени. В отличие от традиционных систем хранения данных, Redis ориентирован на мгновенный доступ к информации, что делает его идеальным для кэширования, где требуется поддерживать высокую производительность и скорость обработки данных. В сочетании с поддержкой отказоустойчивости и возможностей масштабирования, Redis продолжает оставаться одним из самых востребованных решений в области кэширования для современных

распределенных систем.

Hazelcast — это мощная распределенная система для кэширования и управления данными, которая ориентирована на обеспечение высокой производительности и гибкости в условиях крупных распределенных приложений. Как in-memory система, Hazelcast предназначена для того, чтобы ускорить доступ к данным за счет их хранения в оперативной памяти, что особенно важно для приложений, нуждающихся в быстрой обработке данных и низкой задержке. В отличие от многих традиционных решений, Hazelcast поддерживает многопоточную архитектуру и ориентирована на работу в кластерах, что позволяет ей обеспечивать высокую масштабируемость и надежность в условиях высокой нагрузки [\[2\]](#).

Благодаря возможности автоматического управления кластерами и распределения данных между узлами, Hazelcast активно используется для кэширования и поддержки временного хранения данных в крупных системах, где требуется балансировать нагрузку и минимизировать время отклика. Hazelcast также отличается встроенной поддержкой отказоустойчивости и высокой доступности, что делает его привлекательным для критически важных приложений. Эти качества позволяют Hazelcast уверенно занимать место среди лидеров решений для кэширования и in-memory хранения данных в современных корпоративных системах, обеспечивая как гибкость в настройке, так и надежность в эксплуатации.

Сравнительный анализ

В данной статье основное сравнение между Redis и Hazelcast проводится по семи ключевым категориям:

1. Многопоточность.
2. Шаблоны кэширования.
3. Кластеризация.
4. Построение запросов.
5. Память.
6. Вычисления.
7. Отказоустойчивость.

Redis — это однопоточная система, использующая высокопроизводительное ядро с минимальными затратами памяти. Такой подход позволяет запускать несколько экземпляров Redis на одной машине, распределяя нагрузку по ядрам ЦП и максимально используя ресурсы системы. Однопоточная модель обеспечивает простоту архитектуры и снижает вероятность возникновения проблемы рассинхронизации кластера — ситуации, когда изолированные части кластера работают независимо, что может приводить к различиям в данных между узлами при операциях записи [\[3\]](#). Это особенно важно для обеспечения целостности данных.

В Hazelcast, напротив, используется многопоточная архитектура с пулом потоков для операций ввода/вывода. На каждом узле кластера операции ввода/вывода распределены между тремя типами потоков: потоки для приёма входящих запросов, потоки для чтения данных от других узлов и клиентов и потоки для записи данных. Благодаря этому Hazelcast может эффективно обрабатывать высокую нагрузку

ввода/вывода, масштабируясь в зависимости от потребностей. Однако многопоточная модель менее защищена от риска возникновения проблемы рассинхронизации кластера, что может вызывать сложности при обеспечении целостности в распределённом кластере [4].

Ключевое различие между Hazelcast и Redis при использовании их в качестве кэширующих решений заключается в гибкости подходов: Redis строго ориентирован на использование одного шаблона кэширования, тогда как Hazelcast поддерживает несколько различных схем. При применении Redis для кэширования данных, хранящихся в другом репозитории (например, в базе данных), требуется использовать шаблон *cache-aside*, что предполагает дополнительные сетевые переходы при каждом обращении к хранилищу.

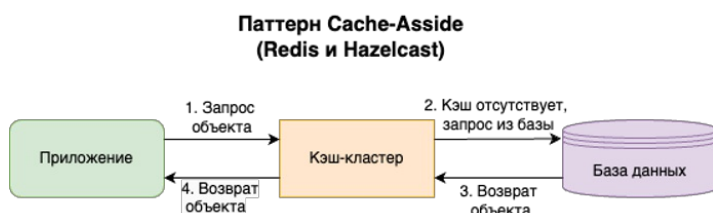


Рисунок 1. Паттерн cache-aside

Hazelcast поддерживает работу как с шаблоном *cache-aside*, так и с шаблонами *read-through* и *write-through*, при которых система автоматически взаимодействует с базой данных, выполняя чтение при отсутствии объекта в кэше или запись при изменении данных.

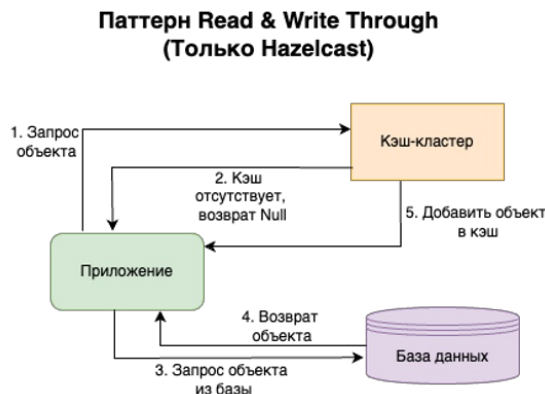


Рисунок 2. Паттерн read/write-through

Эти механизмы освобождают разработчика от необходимости вручную синхронизировать состояние между кэшем и основной базой данных, как это требуется в Redis, где также необходимо реализовывать логику для обновления и чтения данных из хранилища. В Hazelcast, напротив, вся логика взаимодействия с базой данных реализуется на уровне кэширующего слоя, что позволяет писать более чистый, структурированный и поддерживаемый код.

Redis и Hazelcast также имеют различия в реализации кластеризации — механизма объединения нескольких серверов или узлов в единую систему, что позволяет обеспечить высокую доступность, отказоустойчивость и масштабируемость приложений. В Redis кластеризация реализована через шардирование [5], что позволяет распределять данные между несколькими узлами. Каждый узел в этом случае хранит только часть

данных, а кластеры Redis автоматически определяют, какой узел отвечает за конкретный ключ. Тем не менее, процесс добавления или удаления узлов требует ручного перераспределения шардов, что увеличивает сложность управления. Кроме того, при выходе узла из строя необходимо вмешательство разработчиков для восстановления данных и функциональности, а операции с данными, расположенными на других узлах, сопряжены с дополнительными сетевыми затратами.

С другой стороны, Hazelcast изначально разработан как распределенная система и предлагает более совершенные функции кластеризации. Он обеспечивает автоматическое управление распределением данных и динамически перераспределяет их при добавлении или удалении узлов, что значительно упрощает администрирование кластера. При выходе узла из строя Hazelcast автоматически восстанавливает данные и перераспределяет нагрузку, что способствует повышению отказоустойчивости системы. Более того, узлы в Hazelcast могут взаимодействовать между собой без необходимости дополнительных сетевых обращений, что снижает задержки и улучшает производительность.

Таким образом, ключевые различия между Redis и Hazelcast в контексте кластеризации заключаются в том, что Redis требует ручного управления шардированием и восстановлением, тогда как Hazelcast автоматизирует эти процессы, обеспечивая более высокую отказоустойчивость и эффективность межузловое взаимодействия. Redis может быть более подходящим для простых сценариев, в то время как Hazelcast предпочтителен для распределенных систем, требующих высокой доступности и автоматического управления.

Запросы в Redis и Hazelcast, несмотря на общее использование модели «ключ/значение», имеют существенные различия в подходах к извлечению данных по свойствам значений, когда ключ неизвестен. Hazelcast предоставляет API предикатов, SQL-подобные выражения WHERE, а также механизмы проекций, что позволяет выполнять запросы к сложным объектам и JSON-структурам. Эти возможности делают Hazelcast более удобным для выполнения сложных выборок и анализа данных, сохраняя при этом гибкость работы с объектами различной структуры.

В Redis аналогичный подход требует иной архитектуры хранения: данные обычно представляются в виде хэша, где каждая строка сохраняется с ключом, сформированным на основе первичного ключа таблицы, и дополнительно включается в множество (Set) или отсортированное множество (Sorted Set). Такой метод хранения позволяет реализовать ограниченные возможности запросов, но требует дополнительных шагов при проектировании и может быть менее эффективным для сложных запросов по атрибутам, поскольку Redis не поддерживает полноценные предикаты и SQL-подобные запросы.

При сравнении Hazelcast и Redis в аспекте работы с памятью можно выделить несколько ключевых аспектов, которые отражают их разные архитектурные подходы и стратегии управления данными.

Redis ориентирован на хранение всех данных в оперативной памяти, что обеспечивает молниеносный доступ к ним. Его структура данных оптимизирована для минимизации использования памяти, и он поддерживает различные типы, такие как строки, хэши, списки и множества. Redis использует механизмы сжатия и эффективные аллокаторы памяти, что позволяет экономить ресурсы и увеличивать пропускную способность при работе с большими объемами данных. Однако такой подход подразумевает, что данные,

которые превышают доступный объем оперативной памяти, требуют использования дискового хранилища, что может снизить производительность.

С другой стороны, Hazelcast работает в среде Java и использует кучу памяти, управляемую сборщиком мусора. Это означает, что данные могут храниться как в оперативной памяти, так и в более стабильных хранилищах, таких как диски, что обеспечивает большую гибкость. Однако сборка мусора в Hazelcast может вызывать паузы в работе приложений, особенно при увеличении объема данных [\[6\]](#). Это может привести к временным задержкам и снижению производительности, что критично для высоконагруженных систем.

Еще один важный аспект — распределенная природа хранения данных. Redis обычно работает как единый узел, хотя и поддерживает кластеризацию, которая делит данные между несколькими экземплярами, но при этом управление памятью может стать более сложным. Hazelcast, будучи изначально распределенной системой, обеспечивает автоматическое распределение данных по узлам, что позволяет лучше использовать память и увеличивать отказоустойчивость. Это позволяет Hazelcast справляться с увеличением объема данных, распределяя нагрузку между участниками кластера.

Redis обеспечивает максимальную скорость доступа к данным в памяти, тогда как Hazelcast предлагает большую гибкость и масштабируемость за счет своей распределенной архитектуры, но при этом может столкнуться с проблемами, связанными со сборкой мусора и управлением памятью.

Redis и Hazelcast оба позволяют выполнять вычислительные функции непосредственно на узле или участнике кластера, где хранится определенный набор данных, что особенно полезно в сценариях кэширования, требующих локального доступа к кэшированным данным. Например, в Hazelcast программу можно направить на узел, где находится конкретный ключ, что позволяет минимизировать сетевые задержки и ускорить доступ к данным, уже размещенным в кэше. В Redis также реализована схожая функциональность, благодаря которой можно эффективно обрабатывать кэшированные данные, избегая дополнительных вызовов к удаленным узлам.

Однако различия в подходах к распределенным вычислениям между Redis и Hazelcast также влияют на их кэширующие возможности. Hazelcast предоставляет Java-программе, запущенной на одном из узлов, доступ к данным на других узлах кластера, что позволяет использовать распределенный кэш более гибко и эффективно. Это особенно полезно при кэшировании данных, которые часто обращаются из разных частей приложения, поскольку Hazelcast поддерживает сценарии кэширования с межузловым взаимодействием и динамическим распределением данных. В Redis же кластеры и механизм Lua-скриптов не поддерживают доступ к кэшированным данным на других узлах, ограничивая кэширование локальной областью узла и снижая возможности гибкого управления кэшем.

Таким образом, Hazelcast предоставляет более мощную инфраструктуру для распределенных вычислений и кэширования, позволяя использовать кэш на нескольких узлах кластера как единую память, что оптимально для приложений с интенсивным межузловым взаимодействием и распределенными задачами. Redis, напротив, лучше подходит для задач, где кэширование и выполнение кода сосредоточены на конкретном узле без необходимости межузловой коммуникации, что делает его эффективным для локализованного кэширования и обработки данных в пределах одного узла.

Отказоустойчивость является важным аспектом в проектировании распределенных

систем, обеспечивающим непрерывность работы приложений даже в условиях сбоя отдельных компонентов [7]. В контексте Redis отказоустойчивость достигается через механизмы репликации и использование Sentinel для мониторинга состояния узлов. В данной архитектуре основной сервер (мастер) может иметь одну или несколько реплик, которые периодически обновляются для синхронизации данных. В случае выхода основного узла из строя, реплика может быть повышена до статуса мастера. Однако этот процесс требует ручного вмешательства или дополнительной настройки Sentinel, что может вызвать временные простои.

Hazelcast изначально разработан с акцентом на автоматизацию процессов отказоустойчивости. Каждый узел в кластере хранит реплики данных [8], что позволяет системе эффективно реагировать на сбои. При выходе узла из строя Hazelcast автоматически перенаправляет запросы к другим узлам, содержащим актуальные копии данных, что минимизирует время простоя и повышает доступность системы. Более того, механизм автоматического восстановления и перераспределения данных позволяет избежать необходимости ручного управления и снижает риски, связанные с потерей данных.

Таким образом, ключевое отличие в подходах к обеспечению отказоустойчивости между Redis и Hazelcast заключается в том, что Redis требует значительных усилий со стороны разработчиков для поддержания работоспособности системы, тогда как Hazelcast предлагает более надежное и автоматизированное решение. Это делает Hazelcast более предпочтительным вариантом для приложений, требующих высокой доступности и надежности в условиях сбоя отдельных компонентов.

Результаты сравнения кратко представлены в сравнительной таблице.

Таблица 1.

Сравнение Redis и Hazelcast

Аспект	Redis	Hazelcast
Потоки	Однопоточный	Многопоточный
Шаблоны кэширования	Cache-aside	Cache-aside, read-through, write-through
Тип кластеризации	Шардирование с ручным управлением	Автоматическое управление и перераспределение
SQL-подобные запросы	Не поддерживает	Поддерживает
Память	Быстрый доступ к данным, но ограничен объемом оперативной памяти	Потенциальные задержки из-за сборки мусора, но есть возможность обработки больших объемов данных
Вычисления	Ограничен работой с данными на одном узле	Распределенные вычисления с автоматической балансировкой нагрузки по узлам
Отказоустойчивость	При сбое требует	При сбое автоматически

	ручного вмешательства и настройки	перенаправляет запросы
--	--------------------------------------	---------------------------

Тестирование производительности Redis и Hazelcast для операций CRUD

Исходные данные

Программа, содержащая тесты, написана на платформе Java Spring Boot. Тесты проводятся на наборе данных, помещенных в файл формата CSV. Он включает в себя более 250 000 уникальных записей, каждая из которых представлена различными типами данных (строки, целые числа, длинные целые числа и числа с плавающей точкой).

Используемый клиент Java для Redis — Jedis. Jedis — это клиентская библиотека Java для взаимодействия с базой данных Redis, предоставляющая удобный и высокопроизводительный API для выполнения операций с данными в Redis. Jedis поддерживает основные команды Redis, а также расширенные функции [\[9\]](#).

Используемый клиент Java для Hazelcast — Hazelcast Predicates API. Он предоставляет интерфейс для выполнения запросов по заданным условиям в распределённых структурах данных, таких как IMap. С его помощью можно фильтровать и находить данные по сложным условиям, подобным SQL-запросам, что обеспечивает гибкость и высокую производительность при работе с распределёнными данными [\[10\]](#).

Набор проведенных тестов включал в себя:

- Создание (Create): исходные данные загружаются и вставляются в структуру HSET (Redis) и IMap (Hazelcast). Тест проводится в течение 1 часа. На протяжении этого времени данные читаются из CSV файла и добавляются в базу, пока не истечёт время теста.
- Чтение (Read): выполняется 5 типов запросов с разными параметрами, которые выбираются случайным образом из предварительно определённого множества. Подготавливается набор запросов, которые выполняются по очереди в течении 1 часа. Количество выполняемых запросов в секунду фиксируется.
- Обновление (Update): аналогично тесту чтения, но с другим набором запросов. Время выполнения составляет 1 час. Результат — количество обновлённых строк в секунду.
- Удаление (Delete): тест аналогичен обновлению, выполняется в течение 1 часа, фиксируется количество удалённых строк в секунду.

Конфигурация оборудования

Тесты проводились на машине со следующими характеристиками:

- Процессор: Apple M2 @ 3.49 GHz, 8 ядер, 8 потоков.
- Оперативная память: 16 ГБ.
- Накопитель: 256GB PCIe® NVMe™ M.2 SSD.

Результаты тестирования производительности

Create: Redis и Hazelcast показали практически аналогичные результаты при вставке данных, обрабатывая одинаковое количество запросов в секунду. Redis показал в среднем вставку 1040 записей в секунду, Hazelcast – 10020 записей.

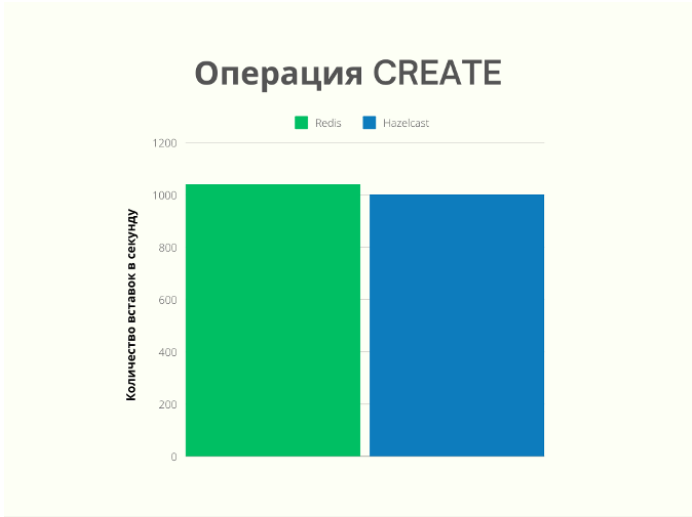


Рисунок 3. Результат операции Create

Read: Redis показал более высокую производительность при чтении данных. Это объясняется более высокой эффективностью библиотеки Redis (Jedis), что позволило быстрее обрабатывать запросы. Redis читал в среднем 14 записей в секунду, Hazelcast – 4. Однако Redis обеспечивает высокую скорость чтения за счет полного хранения данных в памяти, что делает его более требовательным к объему RAM. Таким образом, Redis понадобилось практически в два раза больше памяти, чем Hazelcast, для хранения того же объёма данных (Redis около 1300 байт на запись, Hazelcast около 650 байт).

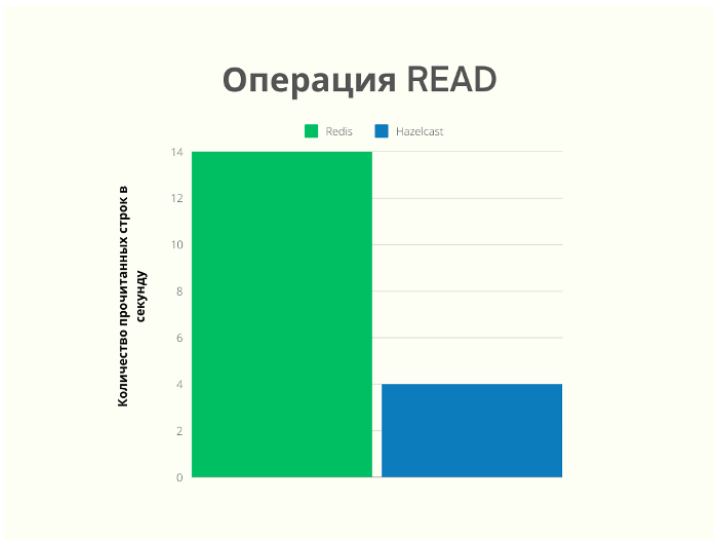


Рисунок 4. Результат операции Read

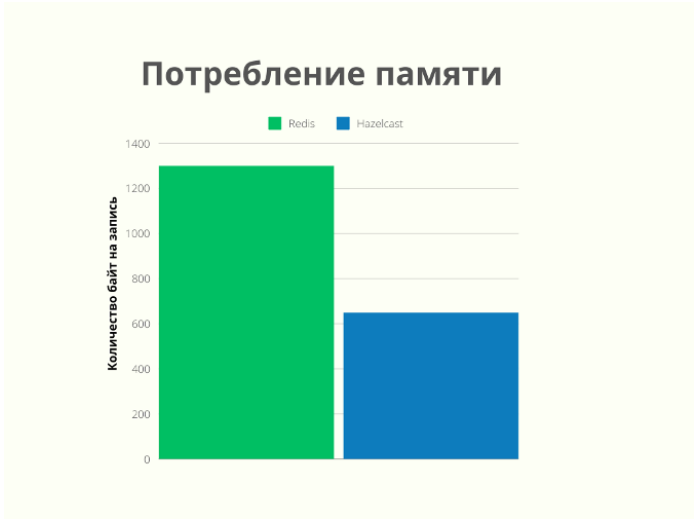


Рисунок 5. Потребление памяти

Update и Delete: Оба решения показали сопоставимую производительность при тестировании операций обновления и удаления данных, обеспечивая минимальные задержки и высокую скорость выполнения. Redis в среднем обновлял 1203 записи в секунду, Hazelcast – 1066 в секунду. При удалении Redis в среднем показал результат 1380 записей в секунду, Hazelcast – 1275 записей в секунду.

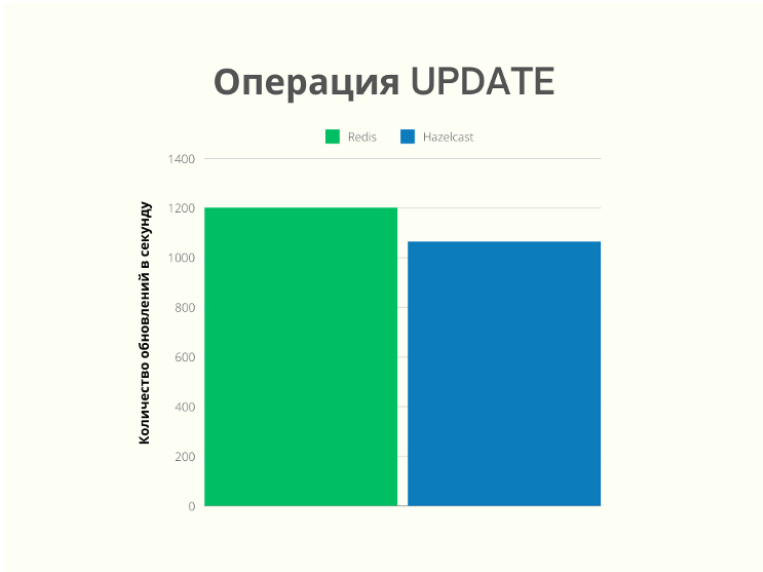


Рисунок 6. Результат операции Update



Рисунок 7. Результат операции Delete

Выводы

В ходе проведенного исследования был осуществлен сравнительный анализ двух популярных систем кэширования и распределенного хранения данных — Redis и Hazelcast. Оценка их архитектуры, производительности, механизмов отказоустойчивости и масштабируемости позволила выявить ключевые особенности, влияющие на выбор оптимального решения в контексте высоконагруженных распределенных приложений.

Redis, характеризующийся однопоточной архитектурой и высокой производительностью в операциях с данными, демонстрирует низкие задержки и быстрый доступ к данным, что делает его подходящим для приложений, ориентированных на быструю работу с данными в памяти. Однако его ограниченная гибкость в вопросах кластеризации и отказоустойчивости может потребовать дополнительных усилий при управлении несколькими узлами и обеспечении доступности данных в более сложных сценариях. Redis оптимален для горизонтально масштабируемых систем с простыми требованиями к распределению данных.

Система Hazelcast с многопоточной архитектурой и встроенной поддержкой кластеризации предоставляет более гибкие и масштабируемые возможности для построения распределенных систем. Механизмы автоматического распределения данных и масштабирования значительно упрощают администрирование и повышают отказоустойчивость. Hazelcast более эффективно справляется с интенсивными межузловыми взаимодействиями и высокими требованиями к отказоустойчивости, что делает его предпочтительным решением для крупных приложений с высокой нагрузкой.

Тестирование производительности показало, что Redis превосходит Hazelcast по скорости выполнения операций чтения. Обе системы показали схожие результаты при операциях обновления и удаления данных, но Redis требует большего объема оперативной памяти, что следует учитывать при выборе решения для ресурсоемких приложений.

Таким образом, выбор между Redis и Hazelcast должен основываться на специфике задач. Redis более подходит для приложений, где критична высокая производительность при работе с данными в памяти, а Hazelcast является более оптимальным вариантом для масштабируемых распределенных систем, требующих высокой отказоустойчивости и гибкости.

Библиография

1. Carlson J. Redis in Action. New York: Manning, 2013.
2. Johns M. Getting Started with Hazelcast. Birmingham: Packt Publishing, 2015.
3. Биллиг, В. А. Параллельные вычисления и многопоточное программирование. Москва: ИНТУИТ, 2016.
4. Кадомский А. А., Захаров В. А. Эффективность многопоточных приложений // Научный журнал. 2016. №7 (8). URL: <https://scientificmagazine.ru/images/PDF/2016/8/Nauchnyj-zhurnal-7-8.pdf> (дата обращения: 08.11.2024).
5. Бойченко А. В., Рогожин Д. К., Корнеев Д. Г. Алгоритм динамического масштабирования реляционных баз данных в облачных средах // Статистика и экономика. 2014. №6-2. URL: <https://statecon.rea.ru/jour/article/view/584/566> (дата обращения: 08.11.2024).
6. Филатов А. Ю., Михеев В. В. Анализ эффективности потоково-локальной сборки мусора в распределённых системах хранения и обработки данных // Вестник СибГУТИ. 2022. №1 (57). URL: <https://vestnik.sibsutis.ru/jour/article/view/122/126> (дата обращения: 08.11.2024).
7. Голева А. И., Стороженко Н. Р., Потапов В. И., Шафеева О. П. Математическое моделирование отказоустойчивости информационных систем // Вестник НГУ. Серия: Информационные технологии. 2019. №4. URL: <https://intechngu.elpub.ru/jour/article/view/110/98> (дата обращения: 08.11.2024).
8. Борсук Н. А., Козеева О. О. Анализ методов репликации баз данных при разработке онлайн-сервиса // Символ науки. 2016. №11-3. URL: <https://os-russia.com/SBORNIKI/SN-2016-11-3.pdf> (дата обращения: 08.11.2024).
9. Intro to Jedis – the Java Redis – URL: <https://www.baeldung.com/jedis-java-redis-client-library> (date of access: 08.11.2024).
10. Predicates API – URL: <https://docs.hazelcast.com/hazelcast/5.5/query/predicate-overview> (date of access: 08.11.2024).

Результаты процедуры рецензирования статьи

В связи с политикой двойного слепого рецензирования личность рецензента не раскрывается.

Со списком рецензентов издательства можно ознакомиться [здесь](#).

В статье рассматривается сравнительный анализ двух ведущих in-memory решений для кэширования в распределённых системах — Redis и Hazelcast. Оба инструмента используются для повышения производительности современных распределённых систем за счёт ускорения доступа к данным и снижения нагрузки на основные базы данных. Основное внимание уделено сравнительным характеристикам архитектуры, отказоустойчивости, масштабируемости и производительности, что позволяет оценить их применимость в контексте конкретных задач.

Методология исследования базируется на сравнительном анализе архитектурных особенностей Redis и Hazelcast, а также на тестировании производительности этих платформ при выполнении операций CRUD. Тестирование проводилось с использованием Java Spring Boot и клиентских библиотек, таких как Jedis и Hazelcast Predicates API. Результаты тестирования проанализированы для различных операций: создания, чтения, обновления и удаления данных, что позволило сделать выводы о производительности и потреблении ресурсов.

Актуальность исследования обусловлена ростом объемов данных и требованиями к быстрдействию современных распределённых систем. В условиях необходимости

масштабирования приложений и увеличения скорости обработки запросов выбор оптимального кэширующего решения становится важным этапом проектирования системы. Поэтому анализ преимуществ и ограничений Redis и Hazelcast является своевременным и востребованным.

Научная новизна работы заключается в комплексном сравнительном анализе Redis и Hazelcast, проведённом в контексте их применения в современных backend-архитектурах. В статье подробно рассматриваются ключевые аспекты, влияющие на производительность и отказоустойчивость, такие как многопоточность, поддержка различных шаблонов кэширования, особенности кластеризации и работа с памятью. Также важным вкладом является предоставление результатов тестирования на реальных данных, что подтверждает выводы о практической применимости рассматриваемых решений.

Статья отличается хорошо структурированным изложением материала. Введение предоставляет необходимый контекст и объясняет значимость исследования. Основная часть разделена на тематические блоки, каждый из которых рассматривает отдельный аспект сравнения, что позволяет легко следовать ходу анализа. Результаты тестирования представлены с использованием наглядных иллюстраций и графиков, что способствует более глубокому пониманию полученных данных.

В статье сделаны важные выводы относительно преимуществ и недостатков Redis и Hazelcast в зависимости от задач, стоящих перед разработчиком. Redis демонстрирует высокую производительность в операциях чтения данных, однако требует большего объёма оперативной памяти и ручного управления при кластеризации. Hazelcast предлагает более гибкие возможности для распределённых систем, включая автоматическое масштабирование и управление отказоустойчивостью. Такие результаты будут интересны специалистам в области разработки высоконагруженных приложений и архитектуры распределённых систем, а также всем, кто рассматривает вопросы оптимизации производительности с использованием кэширующих решений.

Статья представляет собой ценный вклад в исследование эффективности распределённых кэширующих платформ. Проведённый анализ выявляет сильные и слабые стороны Redis и Hazelcast, что помогает сделать обоснованный выбор при проектировании backend-архитектуры современных приложений. Работа написана в академическом стиле, структура изложения логична и позволяет легко воспринимать материал. Рекомендуется к публикации, так как содержит ценные данные и выводы, которые могут быть полезны широкой аудитории специалистов и исследователей в области разработки программного обеспечения.

Англоязычные метаданные

Analysis of the possibilities of determining location in a Wi-Fi network using neural network algorithms**Lizneva Yuliya Sergeevna**

PhD in Technical Science

Associate Professor; Telecommunications Institute; Siberian State University of Telecommunications and Informatics

86 Kirova str., Novosibirsk, Novosibirsk region, 630102, Russia

✉ ktm5r@rambler.ru

**Kostyukovich Anatoliy Egorovich**

PhD in Technical Science

Associate Professor; Department of Automatic Telecommunications; Siberian State University of Telecommunications and Informatics

Head; Scientific and Educational Center for high-tech industrial enterprises of Novosibirsk

630102, Russia, Novosibirsk region, Novosibirsk, Kirova str., 86, office 404

✉ aek1954@gmail.com

**Kokoreva Elena Viktorovna**

PhD in Technical Science

Associate Professor, Siberian State University of Telecommunications and Informatics

630102, Russia, Novosibirsk region, Novosibirsk, ul. Kirova, 86, office 605

✉ elen.vik@gmail.com



Abstract. Indoor positioning on a Wi-Fi network belongs to a class of tasks in which the dependence of output characteristics on input variables is influenced by many parameters and external factors. When solving such problems, it is necessary to take into account that in determining the location, it is of significant interest not only to determine the static coordinates of an object, but also to predict the vector of its movements. In the case where the location of an object is determined only by the level of signal power received from several access points on a Wi-Fi network, the use of signal attenuation models that take into account the conditions of propagation of radio waves indoors is difficult due to the need for reliable information about the material of ceilings, floors and ceilings, the presence of fixed and mobile shading objects, etc. Since the electromagnetic environment inside the room varies depending on many factors, the above-mentioned models have to be adjusted to these changes. Since finding patterns in a large amount of data requires non-standard algorithms, artificial neural networks can be used to solve the positioning problem. It is important to choose a neural network architecture that can take into account changes in the signal strength received by a mobile device from Wi-Fi access points. Before training a neural network, statistical data is preprocessed. For example, abnormal cases are excluded from the machine learning dataset when the device detects a signal from less than three access points at one measuring point.

As a result of the analysis of statistical data, it was found that the same distance between the measuring points leads to the fact that the neural network incorrectly determines the location of the object. The paper shows that in order to increase the accuracy of positioning the location in conditions of complex radio placement, when compiling radio maps, it is necessary to determine the optimal varying distances between measuring points. The

conducted experimental studies, taking into account the proposed approach to optimizing the distances between measuring points, prove that the accuracy of location determination in the vast majority of measuring points reaches 100%.

Keywords: machine learning, hidden layer, signal strength, neural network, RSSI, measuring point, positioning, Wi-Fi, training sample, training set

References (transliterated)

1. Andreev R.A., Ostroumov S.I., Fedorov A.S. Metody pozitsionirovaniya v setyakh Wi-Fi // *Ekonomika i kachestvo sistem svyazi*. 2021. № 3 (21). S. 50-63.
2. Kokoreva E.V., Kostyukovich A.E., Doshchinskii I.V. Otsenka pogreshnosti izmerenii mestonakhozhdeniya abonenta v seti Wi-Fi // *Programmnye sistemy i vychislitel'nye metody*. 2019. № 4. S. 30-38. DOI: 10.7256/2454-0714.2019.4.31316 URL: https://e-notabene.ru/itmag/article_31316.html
3. Kokoreva, E.V., Shurygina, K.I. Bragin, A.S. Impact of Wi-Fi network coverage planning on the logistics objects location accuracy // *XV International Scientific Conference on Precision Agriculture and Agricultural Machinery Industry "State and Prospects for the Development of Agribusiness-INTERAGROMASH 2022"*. 2022. Vol. 363.
4. Kokoreva, E.V. & Shurygina, K.I. An Assessment of the Local Positioning System Effectiveness // *Lecture Notes in Networks and Systems*. 2022. Vol. 246. Pp. 436–443.
5. Kokoreva E., Kostyukovich A., Shurygina K., Doshchinsky I. Experimental Study of the Positioning System in the Centralized Wi-Fi Network // *Lecture Notes on Data Engineering and Communications Technologies*. 2022. Vol. 107. Pp. 346–357.
6. Kireev A.V., Fokin G.A. Otsenka tochnosti lokal'nogo pozitsionirovaniya mobil'nykh ustroystv s pomoshch'yu radiokart i inertsial'noi navigatsionnoi sistemy // *Trudy uchebnykh zavedenii svyazi*. 2017. № 4. S. 54-62.
7. Kuchin I.Yu., Iksanov Sh.Sh., Rozhdestvenskii S.K., Koryakov A.N. Razrabotka sistemy pozitsionirovaniya i kontrolya ob'ektov s pomoshch'yu besprovodnoi tekhnologii Wi-Fi // *Nauchnyi vestnik Novosibirskogo gosudarstvennogo tekhnicheskogo universiteta*. 2015. № 3 (60). S. 130-146.
8. Sadovnikova N.A., Shmoilova R.A. Analiz vremennykh ryadov i prognozirovanie / M.: Evraziiskii otkrytyi institut, 2024.
9. Akhmetkhanov R.S., Dubinin E.F., Kuksova V.I. Analiz vremennykh ryadov v diagnostike tekhnicheskikh sistem // *Mashinostroyeniye i inzhenernoye obrazovanie*. 2013. № 2. S. 11–20.
10. Chen W., Hussain W., Cauteruccio F., Zhang X. Deep Learning for Financial Time Series Prediction: A State-of-the-Art Review of Standalone and Hybrid Models // *Computer Modeling in Engineering & Sciences*. 2024. Vol. 139. № 1. Pp. 187-224.
11. Ezhov A.A., Shumskii S.A. Neirokomp'yuting i ego primeneniya v ekonomike i biznese. M.: INTUIT, 2016.
12. Karakida, R., Takase, T. Optimal layer selection for latent data augmentation // *Neural Networks*. 2024. Vol. 181.
13. Sandnes, A.T., Grimstad, B., & Kolbjørnsen, O. Multi-task neural networks by learned contextual inputs // *Neural Networks*. 2024. Vol. 179.
14. Khaikin S. Neironnye seti. Polnyi kurs. M.: Vil'yams, 2019.
15. Zhou, X., You, Zh., Sun, W., Zhao, D., Yan, Sh. Fractional-order stochastic gradient descent method with momentum and energy for deep neural networks // *Neural*

Networks. 2024. Vol. 181.

16. Lizneva Yu.S. Issledovanie trafika OKS N 7 i razrabotka metodiki ego prognozirovaniya / Novosibirsk: SibGUTI, 2008.
17. Marshoodulla S.Z, Saha G. A survey of data mining methodologies in the environment of IoT and its variants // Journal of Network and Computer Applications. 2024. Vol. 228.
18. Zhou X., Du H., Xue Sh., Ma Zh. Recent advances in data mining and machine learning for enhanced building energy management // Energy. 2024. Vol. 307.
19. Ivan'ko A.F., Ivan'ko M.A., Sizova Yu.A. neironnye seti: obshchie tekhnologicheskie kharakteristiki // Nauchnoe obozrenie. Tekhnicheskie nauki. 2019. № 2. S. 17-23.
20. Sevilla-Salcedo C., Gallardo-Antolín A., Gómez-Verdejo V., Parrado-Hernández E. Bayesian learning of feature spaces for multitask regression // Neural Networks. 2024. Vol. 179.

Comparison of automatic summarization of texts in Russian

Dagaev Alexander Evgenevich 

Postgraduate student, Department of Informatics and Information Technologies, Moscow Polytechnic University

38 Bolshaya Semyonovskaya str., Moscow, 107023, Russia

✉ alejaandro@bk.ru

Popov Dmitry Ivanovich 

Doctor of Technical Science

Sochi State University, Professor of the Department of Information Technologies and Mathematics

354000, Krasnodar region, Sochi, st. Plastunskaya, 94

✉ damitry.popov@gmail.com

Abstract. The subject of the research in this article is the generalization of texts in Russian using artificial intelligence models. In particular, the authors compare the popular models GigaChat, YaGPT2, ChatGPT-3.5, ChatGPT-4, Bard, Bing AI and YouChat and conduct a comparative study of their work on Russian texts. The article uses datasets for the Russian language, such as Gazeta, XL-Sum and WikiLingua, as source materials for subsequent generalization, as well as additional datasets in English, CNN Dailymail and XSum, were taken to compare the effectiveness of generalization. The article uses the following indicators: ROUGE, BLEU score, BERTScore, METEOR and BLEURT to assess the quality of text synthesis. In this article, a comparative analysis of data obtained during automatic generalization using artificial intelligence models is used as a research method. The scientific novelty of the research is to conduct a comparative analysis of the quality of automatic generalization of texts in Russian and English using various neural network models of natural language processing. The authors of the study drew attention to the new models GigaChat, YaGPT2, ChatGPT-3.5, ChatGPT-4, Bard, Bing AI and YouChat, considering and analyzing their effectiveness in the task of text generalization. The results of the generalization in Russian show that YouChat demonstrates the highest results in terms of the set of ratings, emphasizing the effectiveness of the model in processing and generating text with a more accurate reproduction of key elements of content. Unlike YouChat, the Bard model showed the worst results, representing the model with the least ability to generate coherent and relevant text. The data obtained during the comparison will contribute to a deeper understanding of the models under consideration, helping to make a choice when using artificial intelligence for

text summarization tasks as a basis for future developments.

Keywords: text compression, YouChat, Bing AI, Bard, ChatGPT-4, ChatGPT-3, YaGPT2, GigaChat, text summarization, natural language processing

References (transliterated)

1. Goyal T., Li J. J., Durrett G. News summarization and evaluation in the era of gpt-3 //arXiv preprint arXiv:2209.12356. – 2022.
2. Zhang T. et al. Benchmarking large language models for news summarization //arXiv preprint arXiv:2301.13848. – 2023.
3. Gusev I. Dataset for automatic summarization of Russian news //Artificial Intelligence and Natural Language: 9th Conference, AINL 2020, Helsinki, Finland, October 7–9, 2020, Proceedings 9. – Springer International Publishing, 2020. – S. 122-134.
4. Lin C. Y. Rouge: A package for automatic evaluation of summaries //Text summarization branches out. – 2004. – S. 74-81.
5. Post M. A call for clarity in reporting BLEU scores //arXiv preprint arXiv:1804.08771. – 2018.
6. Bhaskar A., Fabbri A., Durrett G. Prompted opinion summarization with GPT-3.5 //Findings of the Association for Computational Linguistics: ACL 2023. – 2023. – S. 9282-9300.
7. Tang L. et al. Evaluating large language models on medical evidence summarization //npj Digital Medicine. – 2023. – T. 6. – №. 1. – S. 158.
8. Hendy A. et al. How good are gpt models at machine translation? a comprehensive evaluation //arXiv preprint arXiv:2302.09210. – 2023.
9. Jiao W. et al. Is ChatGPT a good translator? Yes with GPT-4 as the engine //arXiv preprint arXiv:2301.08745. – 2023.
10. Narayan S., Cohen S. B., Lapata M. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization //arXiv preprint arXiv:1808.08745. – 2018.
11. Nallapati R. et al. Abstractive text summarization using sequence-to-sequence rnns and beyond //arXiv preprint arXiv:1602.06023. – 2016.
12. Hasan T. et al. XL-sum: Large-scale multilingual abstractive summarization for 44 languages //arXiv preprint arXiv:2106.13822. – 2021.
13. Zhang T. et al. Bertscore: Evaluating text generation with bert //arXiv preprint arXiv:1904.09675. – 2019.
14. Banerjee S., Lavie A. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments //Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization. – 2005. – S. 65-72.
15. Sellam T., Das D., Parikh A. P. BLEURT: Learning robust metrics for text generation //arXiv preprint arXiv:2004.04696. – 2020.
16. Ladhak F. et al. WikiLingua: A new benchmark dataset for cross-lingual abstractive summarization //arXiv preprint arXiv:2010.03093. – 2020.

Proof of Performance Consensus Model and Algorithm

Postgraduate student; Department of Computer Engineering and Automated Systems Software; Belgorod State Technological University named after V.G. Shukhov

46 Kostyukova str., Belgorod, Belgorod region, 308012, Russia

✉ bulgakowlad@yandex.ru

Gvozdevsky Igor Nikolaevich 

PhD in Technical Science

Associate Professor; Department of Computer Engineering and Automated Systems Software; Belgorod State Technological University named after V.G. Shukhov

308012, Russia, Belgorod region, Belgorod, Kostyukova str., 36, office 421

✉ Gvozdevskiy.in@bstu.ru

Abstract. The article examines the working principle of the Proof of Performance (PoP) model, based on a consensus algorithm that supports horizontal sharding functions. The PoP model introduces changes to the traditional block structure used in Proof of Stake algorithms and Tendermint-based networks. Horizontal sharding allows transactions to be distributed among multiple nodes (shards), significantly increasing the network's throughput. The main goal of the study is to explore ways to enhance the efficiency and scalability of blockchain networks through dynamic transaction distribution and adaptive node management. An important aspect is the definition of parameters and adjustable characteristics of nodes, such as performance and reliability, to ensure even and fair load distribution within the network. This provides the system with the ability to adapt to changing load conditions.

The study employs analytical and formal methods to describe the block structure, transaction distribution mechanism, and the system of penalties and rewards for shards.

The research represents an innovative approach to managing blockchain networks, focusing on node performance. The PoP model with horizontal sharding provides higher throughput and scalability compared to traditional consensus algorithms. A system of dynamic load distribution and adaptive weight adjustment of nodes based on their performance is proposed, which contributes to the improvement of the network's efficiency and reliability. The results of the study demonstrate that the Proof of Performance model significantly increases transaction processing speed and overall blockchain network performance. Application examples confirm the model's effectiveness in various types of networks, such as DeFi platforms, supply chain management systems, and IoT networks. The PoP model encourages nodes to maintain high performance, ensuring fair load distribution and enhancing the overall network resilience.

Keywords: Blockchain, Load distribution, Performance, Horizontal sharding, Proof of Stake, Proof of Performance, Block, Shard, Consensus algorithm, Consensus model

References (transliterated)

1. Bauer V.P., Pobyvaev S.A., Kuznetsov N.V. Potentsial ispol'zovaniya tekhnologii raspredelennogo reestra (blokchein) v sistemakh gosudarstvennogo upravleniya // Fundamental'nye issledovaniya. 2019. № 12 (chast' 2). S. 247–252.
2. Boriskevich I.A. Algoritmy konsensusa v blokchein setyakh // Belorusskii gosudarstvennyi universitet informatiki i radioelektroniki. 6-ya Nauchnaya Konferentsiya Aspirantov, Magistrantov i Studentov BGUIR. Minsk, 2020. S. 116–117.
3. Luu, L., Narayanan, V., Zheng, C., Baweja, K., Gilbert, S., & Saxena, P. (2016). A

- Secure Sharding Protocol for Open Blockchains. In Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (pp. 17–30).
4. Kokoris-Kogias, E., Jovanovic, P., Gasser, L., Gailly, N., Syta, E., & Ford, B. (2017). OmniLedger: A Secure, Scale-Out, Decentralized Ledger via Sharding. IACR Cryptology ePrint Archive, 2017(406).
 5. Zamani, M., Movahedi, M., & Raykova, M. (2018). RapidChain: Scaling Blockchain via Full Sharding. In Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (pp. 931–948).
 6. King, S., & Nadal, S. (2012). PPCoin: Peer-to-Peer Crypto-Currency with Proof-of-Stake. Retrieved from <https://bitcoin.pryaudio.org/vendor/peercoin-paper.pdf>
 7. Swan, M. (2015). Blockchain 2.0: Contracts. In Blockchain: Blueprint for a New Economy (pp. 9–10). O'Reilly Media, Inc.
 8. Shalamov G. A., Petukhov A. S. Sliyanie tekhnologii IoT i blokchein: ot teorii do real'nogo vremeni // Progressivnaya ekonomika. 2023. № 9. S. 32. DOI: 10.54861/27131211_2021_9_32.
 9. Goswami, S. (2017). Scalability analysis of blockchains through blockchain simulation. Thesis, Master of Science in Computer Science, University of Nevada, Las Vegas.
 10. Zamyatin, A., Harz, D., Lind, J., Gudgeon, L., Werner, S., & Knottenbelt, W. J. (2019). XCLAIM: Trustless, Interoperable, Cryptocurrency-Backed Assets. In IEEE Symposium on Security and Privacy (pp. 193–210).
 11. Makrakis, D., & Senhaji, A. (2023). Sharding-Based Proof-of-Stake Blockchain Protocols: Key Components & Probabilistic Security Analysis. Sensors, 23(5), 2819. DOI: 10.3390/s23052819.
 12. Eyal, I., Gencer, A. E., Sirer, E. G., & van Renesse, R. (2016). Bitcoin-NG: A Scalable Blockchain Protocol. In Proceedings of the 13th USENIX Symposium on Networked Systems Design and Implementation (NSDI '16) (pp. 45–59).
 13. Abdulzhalilov A.Z. Metody i strategii masshtabiruemosti blokchein-tekhnologii: analiz, sravnenie i perspektivy // Mezhdunarodnyi nauchnyi zhurnal «Vestnik nauki». 2023. № 11 (68). T. 4. S. 625–634.
 14. Grepan V.N. Prakticheskie problemy ispol'zovaniya blokchein-tekhnologii // Nauchnyi setevoy zhurnal «Stolypinskii vestnik.» 2024. № 9.

Analysis of the appropriate behavior of various types of automata in the conditions of the placement game

Dimitrichenko Dmitry Petrovich 

PhD in Technical Science

Researcher, Institute of Applied Mathematics and Automation

89a Shortanova str., Nalchik, 360000, Russia, Republic of Kbr

✉ dimdp@rambler.ru

Abstract. The object of research in this work is homogeneous collectives of automata with the property of purposeful behavior. The subject of this study is a comparison of different designs of such machines in the implementation of the conditions of the game of placement. The aim of the study is to establish the best (or similar) structures in terms of properties in order to optimize the time and computational costs of more complex machine learning models based on the principle of reinforcement learning. In the collectives under consideration, automata perform actions in a given habitat (functioning) with varying degrees

of effectiveness. The automata, in accordance with their design, react to the input signal with another action. The evaluation of the effectiveness of the machine is defined as the sum of positive signals (rewards) or negative signals (penalties) received by the machine during the considered period of time. This characteristic depends on both the declared design of the machine and the depth of its memory. It is necessary to determine the simplest designs of automata that allow achieving optimal efficiency in a given environment in the shortest possible way. The formalization of both the properties of the environment and the actions of automata, as well as the processing of the results obtained, is carried out using the apparatus of game theory. In this case, the values of the effectiveness of the functioning of the machines are represented as the cumulative amounts of winnings and losses of the slot machine players. As result of the research the designs of automata that provide a given efficiency of functioning with a minimum depth of memory (the least complex design) are presented. The result obtained makes it possible to trace the influence of the inertial qualities of automata, implemented in the form of appropriate structures, on the efficiency of functioning in a given environment, formalized in the form of a game of placement. An automaton with linear tactics and a Krylov automaton form two marginal implementations of an automaton strategy for approaching the optimum. The first is due to the high speed of changing actions, the second is due to a long stay in states close to optimal. The field of application of the results obtained is further investigation of more complex dynamic environments using the simplest designs of automata, since synchronous collectives of automata in the process of computational implementation are difficult to parallelize, which leads to a significant increase in time and computational costs with the complication of the structure of dynamic environments or with an increase in these optimization tasks.

Keywords: optimal strategy, game theory, memory depth, machine, reinforcement learning, fines, incentives, appropriate behavior, homogeneous group of machines, placement game

References (transliterated)

1. Stefanyuk V.L. Lokal'naya organizatsiya intellektual'nykh sistem. M.: Fizmatlit. 2004. S. 328.
2. Gaaze-Rapoport M.G., Pospelov D.A. Ot ameby do robota. Modeli povedeniya. M.: Lenand. 2019. S. 304.
3. Zhuravlev A.L., Savchenko T.N., Golovina G.M. Matematicheskaya psikhologiya: shkola V. Yu. Krylova-Ser. Nauchnye shkoly Instituta psikhologii RAN. M.: Izd-vo IP RAN. 2010. S. 512.
4. Zhdanov A.A. Avtonomnyi iskusstvennyi intellekt. M.: Laboratoriya znaniy. 2024. S. 362.
5. Dimitrichenko D.P. Optimizatsiya rekurrentnoi neironnoi seti pri pomoshchi avtomatov s peremennoi strukturoi // Programmnye sistemy i vychislitel'nye metody. 2023. № 4. S. 30-43. DOI: 10.7256/2454-0714.2023.4.69011 EDN: FEIPTC URL: https://e-notabene.ru/itmag/article_69011.html
6. Karpov V.E., Karpova I.P., Kulinich A.A. Sotsial'nye soobshchestva robotov. M.: URSS. 2019. C. 352.
7. Karpov V.E., Koroleva M.N. K voprosu o formalizatsii etiki povedeniya kollaborativnogo robota // Informatsionnye i matematicheskie tekhnologii v nauke i upravlenii. 2022. № 4 (28). S. 223-233.
8. Pospelov D.A. Igry i avtomaty. M.: Energiya. 1966. S. 136.
9. Tsetlin M.L. Issledovaniya po teorii avtomatov i modelirovaniyu biologicheskikh sistem.

M.: Nauka. 1969. S. 316.

10. Pospelov D.A. Veroyatnostnye avtomaty. M.: Energiya. 1970. S. 88.

11. Varshavskii V.I. Kollektivnoe povedenie avtomatov. M.: Nauka. 1973. S. 408.

12. Varshavskii V.I., Pospelov D.A. Orkestr igraet bez dirizhera: razmyshleniya ob evolyutsii nekotorykh tekhnicheskikh sistem i upravlenie imi. M.: Nauka. 1984. S. 208.

A method for detecting objects in images based on neural networks on graphs and a small number of training examples

Zakharov Aleksei Aleksandrovich 

PhD in Technical Science

Associate Professor; Department of Software Engineering; Murom Institute (branch) of the Federal State Budgetary Educational Institution of Higher Education 'Madimir State University named after Alexander Grigoryevich and Nikolai Grigoryevich Stoletov

Leading researcher; Murom Institute (branch) of the Federal State Budgetary Educational Institution of Higher Education 'Madimir State University named after Alexander Grigoryevich and Nikolai Grigoryevich Stoletov

602264, Russia, Vladimir region, Murom, Orlovskaya str., 23, room 402

✉ aa-zaharov@ya.ru

Abstract. In the presented work, the object of research is computer vision systems. The subject of the study is a method for detecting objects in images based on neural networks on graphs and a small number of training examples. Such aspects of the topic as the use of a structural representation of the scene to improve the accuracy of object detection are discussed in detail. It is proposed to share information about the structure of the scene based on neural networks on graphs and training from "multiple shots" to increase the accuracy of object detection. Relationships between classes are established using external semantic links. To do this, a knowledge graph is pre-created. The method contains two stages. At the first stage, object detection is performed based on training with "multiple shots". At the second stage, the detection accuracy is improved using a neural network on graphs. The basis of the developed method is the use of convolution based on spectral graph theory. Each vertex represents a category in the knowledge graph, and the edge weight of the graph is calculated based on conditional probability. Based on the convolution, information from neighboring vertices and edges is combined to update the vertex values. The scientific novelty of the developed method lies in the joint use of convolutional networks on graphs and training from "multiple shots" to increase the accuracy of object detection. A special contribution of the author to the research of the topic is the use of a convolutional network based on a knowledge graph to improve the results of the object detection method using a small number of training examples. The method was studied on test sets of images from the field of computer vision. Using the PASCAL VOC and MS COCO datasets, it is demonstrated that the proposed method increases the accuracy of object detection by analyzing structural relationships. The average accuracy of object detection using the developed method increases by 1-5% compared to the "multiple shots" training method without using a structural representation.

Keywords: artificial intelligence, pattern recognition, graph, limited annotation, deep learning, small data set, convolutional networks, object detection, computer vision, structural representation of scenes

References (transliterated)

1. Zou Z., Chen K., Shi Z., Guo Y., Ye J. Object Detection in 20 Years: A Survey //

- Proceedings of the IEEE. 2023. Vol. 111 (3). Pp. 257-276.
2. Redmon J., Divvala S., Girshick R., Farhadi A. You only look once: Unified, real-time object detection // IEEE Conference on Computer Vision and Pattern Recognition. 2016. Pp. 779-788.
 3. Liu W., Anguelov D., Erhan D., Szegedy C., Reed S., Fu C.Y., Berg A. C. Ssd: Single shot multibox detector // European Conference on Computer Vision. 2016. Pp. 21-37.
 4. Lin T.Y., Goyal P., Girshick R., He K., Dollar P. Focal loss for dense object detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2018. Vol. 42(2). Pp. 318-327.
 5. Girshick P. Fast R-CNN // 2015 IEEE International Conference on Computer Vision (ICCV). 2015. Pp. 1440-1448.
 6. Ren S., He K., Girshick R., Sun J. Faster R-CNN: Towards real-time object detection with region proposal networks // Advances in Neural Information Processing System. 2015. Pp. 91-99.
 7. He K., Gkioxari G., Dollar P., Girshick R. Mask R-CNN // Proceedings of the IEEE International Conference on Computer Vision. 2017. Pp. 2961-2969.
 8. Köhler M., Eisenbach M., Gross H. M. Few-Shot Object Detection: A Survey // IEEE Transactions on Neural Networks and Learning Systems. 2024. Vol. 35 (9). Pp. 11958-11978.
 9. Huang G., Laradji I., Vazquez D., Lacoste-Julien S., Rodriguez P. A Survey of Self-Supervised and Few-Shot Object Detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2023. Vol. 45(4). Pp. 4071-4089.
 10. Wu J., Liu S., Huang D., Wang Y. Multi-scale positive sample refinement for few-shot object detection // European Conference on Computer Vision. 2020. Pp. 456-472.
 11. Wang X., Huang T. E., Gonzalez J., Darrell T., Yu F. Frustratingly simple few-shot object detection // Proceedings of the 37th International Conference on Machine Learning (ICML). 2020. Pp. 9919-9928.
 12. Kang B., Liu Z., Wang X., Yu F., Feng J., Darrell T. Few-shot object detection via feature reweighting // 2019 IEEE/CVF International Conference on Computer Vision. 2019.
 13. Zakharov A.A., Tuzhilkin A.Yu. Segmentatsiya sputnikovykh izobrazhenii na osnove superpikselei i razrezov na grafakh // Programmnye sistemy i vychislitel'nye metody. 2018. № 1. S. 7-17. DOI: 10.7256/2454-0714.2018.1.25629 URL: https://e-notabene.ru/itmag/article_25629.html
 14. Zakharov. A.A., Titov D.V., Zhiznyakov A.L., Titov V.S. Metod vizual'nogo vnimaniya na osnove ranzhirovaniya vershin grafa po raznorodnym priznakam izobrazhenii // Komp'yuternaya optika. 2020. T. 44, № 3. S. 427-435.
 15. Barinov A.E., Zakharov A.A. Clustering using a random walk on graph for head pose estimation // International Conference on Mechanical Engineering, Automation and Control Systems, MEACS. 2015.
 16. Cao P., Zhu Z., Wang Z., Zhu Y., Niu Q. Applications of graph convolutional networks in computer vision // Neural Computing and Applications. 2022. № 34. Pp. 13387-13405.
 17. Kipf T.N. Deep Learning with Graph-Structured Representations, Universiteit van Amsterdam, 2020.
 18. Li W., Liu X., Yuan Y. SIGMA++: Improved Semantic-Complete Graph Matching for Domain Adaptive Object Detection // IEEE Transactions on Pattern Analysis and

Machine Intelligence. 2023. Vol. 45 (7). Pp. 9022-9040.

19. Chen C., Li J., Zhou H.Y., Han X., Huang Y., Ding X., Yu Y. Relation matters: Foreground-aware graph-based relational reasoning for domain adaptive object detection // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2023. Vol. 45 (3). Pp. 3677-3694.
20. Chen T., Lin L., Chen R., Hui X., Wu X. Knowledge-Guided Multi-Label Few-Shot Learning for General Image Recognition // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2022. Vol. 44 (3). Pp.1371-1384.
21. Liu Z., Jiang Z., Feng W., Feng H. OD-GCN: Object Detection Boosted by Knowledge GCN // 2020 IEEE International Conference on Multimedia & Expo Workshops (ICMEW). 2020.

An approach to choosing protection mechanisms for personal Internet of Things devices based on a mathematical model with two criteria

Kryazev Maxim Andreevich 

Postgraduate student; Department of Information Security; MREA – Russian Technological University, Institute of Artificial Intelligence

78 Vernadsky Ave., Moscow, 119454, Russia

✉ maxiknyaz@mail.ru

Shabrova Anna Sergeevna 

Student; Department of Information Security; Bauman Moscow State Technical University

5, 2nd Baumanskaya str., building 4, Moscow, 105005, Russia

✉ shabrova.anna.2410@list.ru

Kryuchkov Andrey Andreevich 

Senior Lecturer; Department of Information Security; MREA – Russian Technological University

78 Vernadsky Ave., Moscow, 119454, Russia

✉ kryuchkov_a@mirea.ru

Abstract. Existing methods to protect Personal Internet of Things (PIoT) devices require continuous updates to counter new threats and vulnerabilities. A major task is developing a universal, efficient approach that accounts for the resource constraints of consumer electronics manufacturers. This study introduces a mathematical model applying criteria for implementation complexity and versatility of protection mechanisms to rank them. The goal is to enhance portable smart device security while controlling costs and complying with information security requirements.

The object of the study is the process of securing PIoT devices under current regulatory and technical constraints, as well as limited resources. Its subject is a set of security mechanisms chosen and ranked using a two-criteria model. Within this research, a detailed analysis of recommendations in international and domestic standards was conducted, and feasibility of their adoption was examined through effective resource allocation guided by the model. This research's novelty lies in an original method for selecting PIoT security mechanisms based on complexity and versatility. By accounting for emerging threats, regulatory mandates, and development expenses, the model provides an integrated strategy that minimizes operational overhead. It delivers comprehensive coverage of security needs despite manufacturers' resource constraints. Findings confirm that adopting this model is a promising,

cost-effective way to address PIoT security challenges. By prioritizing complex yet versatile measures, developers can align with legislative requirements and evolving risks. In conclusion, the proposed approach addresses both emergent vulnerabilities and stringent legal obligations, ensuring efficient use of limited resources. By incorporating these criteria, developers can systematically balance complexity, versatility, and cost, achieving stronger protection for consumer IoT products. By emphasizing these factors, it ensures product compliance with evolving standards. The study shows that a two-criteria mathematical model can steer manufacturers toward robust, practical security solutions for Personal Internet of Things devices.

Keywords: IoT security, mathematical model, portable smart devices, Internet, Device protection, Smart device protection, security methodology, PIoT, PIoT devices, personal Internet of Things

References (transliterated)

1. L'vovich I.Ya., Preobrazhenskii A.P., Preobrazhenskii Yu.P., Choporov O.N., Problemy ispol'zovaniya tekhnologii internet veshchei. *Vestnik Voronezhskogo instituta vysokikh tekhnologii*. 2019;13(1):73-75.
2. Biswa Mohan Sahoo, Mohanty SP, Deepak Puthal, Pillai P. Personal Internet of Things (PIoT): What Is It Exactly? *Cyber Security for Next-Generation Computing Technologies*. 2021 Nov 1;10(6):58-60. DOI:10.1201/9781003404361-14
3. Fariha Eusufzai, Aldrin Nippon Bobby, Farzana Shabnam, Saifur Rahman Sabuj. Personal internet of things networks: An overview of 3GPP architecture, applications, key technologies, and future trends. *International journal of intelligent networks*. 2024 Feb 1; 5(6):77-91; DOI:10.1016/j.ijin.2024.02.001
4. Informatsionnaya bezopasnost' v sisteme "Internet veshchei" / A.G. Korobeinikov, A.Yu. Grishentsev, D.I. Dikii [i dr.]. *Vestnik Chuvashskogo universiteta*. 2018. № 1. S. 117-128.
5. Dean A, Agyeman M.O. A Study of the Advances in IoT Security. *Proceedings of the 2nd International Symposium on Computer Science and Intelligent Control - ISCSIC '18*. 2018;1-5; DOI:10.1145/3284557.3284560
6. Kazhenova Zh.S. Bezopasnost' v protokolakh i tekhnologiyakh IoT: obzor / Zh.S. Kazhenova, Zh.E. Kenzhebaeva. *International Journal of Open Information Technologies*. 2022. № 3. S. 10-15. – ISSN 2307-8162
7. Kerimov Vagif Asad Ogly. Algoritm prinyatiya resheniya dlya odnoi mnogokriterial'noi zadachi s matrichnoi model'yu / Vagif Asad Ogly Kerimov, Faik Gasan Ogly Gadzhiev. *Universum: tekhnicheskie nauki*. 2023. № 2. S. 62-65.
8. Yurlov F.F. Metodika kompleksnogo primeneniya nabora printsipov optimal'nosti pri vybore effektivnykh reshenii pri nalichii neopredelennosti vneshnei sredy i mnogokriterial'nosti / F.F. Yurlov, S.N. Yashin, A.F. Plekhanova. *Vestnik Nizhegorodskogo universiteta im. N. I. Lobachevskogo. Seriya: Sotsial'nye nauki*. 2022. № 1. S. 49-55.
9. Bass A.V. Osobennosti raboty s mikrokontrollerom stm32. *Izvestiya Tul'skogo gosudarstvennogo universiteta. Tekhnicheskie nauki*. 2019. № 1. S. 35-40.
10. Saenko M.A. Analiz uyazvimostei besprovodnykh kanalov peredachi informatsii / M.A. Saenko, D.A. Mel'nikov, M.A. Danilov. *Obrazovatel'nye resursy i tekhnologii*. 2023. № 1. S. 82-90.
11. Chimtchik N.V. Vulnerabilities detection via static taint analysis / N.V. Chimtchik, V.N.

- Ignatiev. *Trudy Instituta sistemnogo programmirovaniya RAN*. 2019. T. 31, № 3. S. 177-189.
12. Abdusalomova N.M. Matematicheskoe modelirovanie nauchnykh znaniy kak otdel'naya pozitsiya mezhdru teorii i eksperimentom. *Mirovaya nauka*. 2024. № 6. S. 44-47.
 13. Istratova E.E. Primenenie neironnykh setei dlya obnaruzheniya anomal'nogo trafika v setyakh Interneta veshchei. *International Journal of Open Information Technologies*. 2024. № 1. S. 65-70.
 14. K. E. Jeon, J. She, P. Soonsawad and P. C. Ng. BLE Beacons for Internet of Things Applications: Survey, Challenges, and Opportunities. *IEEE Internet of Things Journal*. V. 5, № 2, R. 811-828, April 2018, DOI: 10.1109/JIOT.2017.2788449.
 15. Prathibha Muraleedhara, Christo S, Jaya J, D. Yuvasini. Any Bluetooth Device Can be Hacked. Know How? *Cyber Security and Applications*. 2024 Feb 1;100041-1. DOI:10.1016/j.csa.2024.100041

The role of operating systems and shells in cloud computing: analysis of OS and shells used in cloud platforms and their impact on cloud infrastructure

Khlestkin Andrei Yurevich 

PhD in Technical Science

Associate Professor; Department of Computer Science and Computer Engineering; Volga Region State University of Telecommunications and Informatics

23 Lva Tolstogo str., Samara, Samara region, 443010, Russia

✉ a.hlestkin@psuti.ru

Raikov Aleksandr Vyacheslavovich 

Student, Department of Computer Science and Computer Engineering, Volga Region State University of Telecommunications and Informatics

23 Lva Tolstogo str., Samara, Samara region, 443010, Russia

✉ sraikov7@mail.ru

Kazantsev Andrei Alekseevich 

Student, Faculty of Information Systems and Technologies, Volga Region State University of Telecommunications and Informatics

23 Lva Tolstogo str., Samara, Samara region, 443010, Russia

✉ NuclearAndGoner@gmail.com

Emelin Daniil Pavlovich 

Student, Department of Computer Science and Computer Engineering, Volga Region State University of Telecommunications and Informatics

23 Lva Tolstogo str., Samara, Samara region, 443010, Russia

✉ demelin163@gmail.com

Larin Denis Vyacheslavovich 

Student, Department of Computer Science and Computer Engineering, Volga Region State University of Telecommunications and Informatics

23 Lva Tolstogo str., Samara, Samara region, 443010, Russia

✉ denlar1989@gmail.com

Abstract. The main focus of the article is on operating systems, cloud computing and command shells, which have been actively developing for several decades and are already part of the life of both an ordinary user and a computer technology professional. These objects are considered as separate components of information technology, as well as their relationship and the results of this relationship. Operating systems in cloud servers perform a managerial role. To be more precise, they manage the resources of physical servers. Operating systems or OS in this case define several parameters. These parameters include how operating systems can use and manage memory and storage for various virtual machines. Command shells, in turn, are represented by an application that provides the user with a command-line interface in which he enters commands both individually and runs scripts consisting of a list of commands. Research methods include theoretical (classification, comparative analysis, literature analysis) and practical (experiment, modeling) approaches. This allows for a comprehensive analysis of the functioning of operating systems and command shells in cloud computing. The scientific novelty of our research is the creation of scripts to perform a particular task in the field of cloud computing on a specific operating system using the above-described command shells.

Thus, the authors provided theoretical data on operating systems and command shells. The authors provided examples of scripts for Bash and Bourne Shell (sh) command shells for the Linux operating system and scripts for Command Prompt command shells (cmd.exe) and Windows PowerShell for the Microsoft Windows operating system. As a result of the analysis of the scripts, a table was compiled with the impact of the operating system and the command shell on cloud computing. The analysis of the table allowed the authors to characterize the objects of research of this scientific work and draw appropriate conclusions.

Keywords: Hyperfine, Bourne Shell, Bash, Microsoft Windows, Linux, cloud infrastructure, cloud platforms, cloud computing, command shells, operating systems

References (transliterated)

1. Savel'ev D.N., Gavrilov S.V. Rol' operatsionnykh sistem v oblachnykh vychisleniyakh: vyzovy i perspektivy / Savel'ev D.N., Gavrilov S.V. [Elektronnyi resurs] // Elektronnyi periodicheskii nauchnyi zhurnal sci-article.ru : [sait]. – URL: <https://sci-article.ru/stat.php/stat.php?i=1697382177> (data obrashcheniya: 15.04.2024).
2. Trubacheva S.I. Pochemu Linux i sistemy real'nogo vremeni? [Tekst] / Trubacheva S.I. // Vestnik Volzhskogo universiteta im. V.N. Tatishcheva. – 2015. – № 2(24). – S. 99-105.
3. Kolosov, L. S., Umaraliev, I. V. Obzor evolyutsii funktsionala operatsionnoi sistemy GNU/Linux pri evolyutsii yadra [Tekst] / L. S. Kolosov, I. V. Umaraliev // Nauchnyi aspekt. – 2023. – № 7. – S. 1390-1393.
4. Passos L., Czarnecki K., Wasowski A. (2012). Towards a catalog of variability evolution patterns: the Linux kernel case. In FOSD '12 Proceedings of the 4th International Workshop on Feature-Oriented Software Development (pp. 62-69). Association for Computing Machinery.
5. Upasana Why do you need the different Linux Shells? / Upasana [Elektronnyi resurs] // edureka : [sait]. – URL: <https://www.edureka.co/blog/types-of-shells-in-linux/#differenttypesofshells> (data obrashcheniya: 18.04.2024).
6. siberianMan O raznykh komandnykh obolochkakh Linux i Unix / siberianMan [Elektronnyi resurs] // Khabr : [sait]. – URL: <https://habr.com/ru/articles/157283/> (data obrashcheniya: 18.04.2024).

7. Leont'ev, V. O., Velikosel'skii, S. A. Semeistvo Microsoft Windows [Tekst] / V. O. Leont'ev, S. A. Velikosel'skii // Alleya Nauki. – 2018. – № 3(19). – S. 727-729.
8. Vlad Massino, Aram Papoyan Tri desyatiletiya Windows / Vlad Massino, Aram Papoyan [Elektronnyi resurs] // gazeta.ru : [sait]. – URL: https://www.gazeta.ru/tech/2015/11/19/7902437/windows_30th_anniversary.shtml (data obrashcheniya: 18.04.2024).
9. Gaurav Bidasaria Command Prompt vs PowerShell vs Windows Terminal: How They Differ / Gaurav Bidasaria [Elektronnyi resurs] // techwiser : [sait]. – URL: <https://techwiser.com/command-prompt-vs-powershell-vs-windows-terminal-comparison/> (data obrashcheniya: 18.04.2024).
10. ru_vds Chto takoe Windows PowerShell i s chem ego edyat? Chast' 1: osnovnye vozmozhnosti / ru_vds [Elektronnyi resurs] // Khabr : [sait]. – URL: <https://habr.com/ru/companies/ruvds/articles/487876/> (data obrashcheniya: 18.04.2024)
11. sharkdp Hyperfine releases v.1.19.0 / sharkdp [Elektronnyi resurs] // github : [sait]. – URL: <https://github.com/sharkdp/hyperfine> (data obrashcheniya: 01.12.2024)

A comparative analysis of the performance of JavaScript code execution environments: Node.js, Deno and Bun

Smirnov Andrei Aleksandrovich 

Student; Faculty of Software Engineering and Computer Technology; ITMO National Research University

197101, Russia, Saint Petersburg, Kronverkskiy str., 49

✉ smirnov.andrew.1999@yandex.ru

Podolskiy Egor Aleksandrovich 

Student; Faculty of Software Engineering and Computer Technology; ITMO National Research University

195112, Russia, Saint Petersburg, Zanevsky str., 28-30-32, sq. 614

✉ egorpodolskiy51@yandex.ru

Cherenkov Artem Vyacheslavovich 

Student; Faculty of Software Engineering and Computer Technology; ITMO National Research University

197101, Russia, Saint Petersburg, Kronverkskiy str., 49 lit. A

✉ art.cherenkov@gmail.com

Gosudarev Ilia Borisovich 

PhD in Pedagogy

Associate Professor; Faculty of Software Engineering and Computer Technology; ITMO National Research University

197046, Russia, Saint Petersburg, Kronverkskiy ave., 49

✉ goss@itmo.ru

Abstract. The subject of the study was the performance of JavaScript program execution in modern environments Node.js, Deno and Bun. These platforms are used to develop server applications and have significant differences in architecture, functionality and performance. Node.js is the most mature and widespread solution and is actively used in most modern web applications. Deno is a newer environment developed by the creator of Node.js, offering improved security, TypeScript support, and other innovations. Bun, on the other hand, is a

modern and high-performance alternative focused on the speed of server-side applications. The purpose of the study is to identify the performance differences between the major modern runtime environments (Node.js, Deno and Bun) for further utilization of these environments in web application development. A computer experiment method using Docker containers and process automation using Ansible was used for the study. The execution time of different scenarios in each of the execution environments was measured. The scientific novelty of this study lies in the fact that for the first time a holistic and valid methodology for measuring and comparing JavaScript code performance in modern runtime environments has been proposed, which will allow researchers to build on the proposed approach in further experiments and extend it to new runtime environments. The results show that Bun shows the best performance in synchronous computations (sorting, JSON processing), but is behind Node.js and Deno in checking for prime numbers. Deno showed high performance in asynchronous operations, thanks to the use of Rust and the Tokio library. Node.js, despite lower results in synchronous tasks, showed stable performance in tests and remains a solid choice for large projects. In the course of the study, recommendations were developed for selecting the appropriate server-side JavaScript code execution environment for various tasks.

Keywords: Docker, Server, Web, Backend, Computer experiment, Performance, Bun, Deno, Node.js, JavaScript

References (transliterated)

1. Martinsen, J., Grahn, H. A methodology for evaluating JavaScript execution behavior in interactive web applications. // V materialakh 9-i Mezhdunarodnoi konferentsii po komp'yuternym sistemam i prilozheniyam IEEE/ACS (AICCSA), 2011. S. 241-248. DOI: 10.1109/AICCSA.2011.6126611.
2. Rodygina I. V., Nalivaiko A. V. Sravnitel'nyi analiz tekhnologii dlya razrabotki servernoi chasti sistemy upravleniya prodazhami: stat'ya v zhurnale // Izvestiya YuFU. Tekhnicheskie nauki. № 4 (221). 2021. S. 256-266. DOI: 10.18522/2311-3103-2021-4-256-266.
3. Tikhonov D.S., Cherenkov A.V., Dolgov A.V. Sravnitel'nyi analiz tekhnologii servernoi razrabotki na platformakh deno i bun // Nauchno-tekhnicheskie innovatsii i veb-tekhnologii. 2023. № 2. S. 55-59. URL: <https://elibrary.ru/item.asp?id=56409551>.
4. Kniazev I., Fitiskin A. Choosing the right javascript runtime: an in-depth comparison of node.js and bun: stat'ya v zhurnale // Norwegian journal of development of the international science. 2023. S. 72-84. DOI: 10.5281/zenodo.7945166. URL: <https://elibrary.ru/item.asp?id=53844306>.
5. Koper D., Woda M. Performance Analysis and Comparison of Acceleration Methods in javascript Environments Based on Simplified Standard Hough Transform Algorithm //International Conference on Dependability and Complex Systems. Cham: Springer International Publishing, 2022. S. 131-142. DOI: 10.1007/978-3-031-06746-4_13.
6. Akin'shin A. Professional'nyi benchmark: iskusstvo izmereniya proizvoditel'nosti. spb.: Piter, 2022. 576 s.: il. (Seriya «Biblioteka programmista»). ISBN 978-5-4461-1551-8.
7. Sal'nikova, K. V. Analiza massiva dannykh s pomoshch'yu instrumenta vizualizatsii "Yashchik s usami" / K. V. Sal'nikova // Universum: ekonomika i yurisprudentsiya. 2021. № 6(81). S. 11-17. DOI 10.32743/unilaw.2021.81.6.11778.
8. Gregg B. BPF Performance Tools (Addison-Wesley Professional Computing Series) // Update. T. 517. S. 18. 2019.
9. Lu J., Gokhale S. S. performance analysis of a Web Server //International Journal of

- Information Technology and Web Engineering (IJITWE). – 2008. – Т. 3. – № 3. – С. 50-65.
10. Lundar J. A., Grønli T. M., Ghinea G. Performance evaluation of a modern web architecture // International Journal of Information Technology and Web Engineering (IJITWE). – 2013. – Т. 8. – № 1. – С. 36-50.
 11. Smirnov A. A., Cherenkov A. V., Podol'skii E. A. Sravnitel'nyi analiz sred ispolneniya Javascript koda Node.js, Deno i Bun dlya razrabotki servernoi chasti veb-prilozheniya // Mezhdunarodnyi zhurnal informatsionnykh tekhnologii i energoeffektivnosti. 2024. Т. 9, № 4(42). С. 100-106.
 12. Lei, K., Ma, Y., & Tan, Z. (2014). Performance Comparison and Evaluation of Web Development Technologies in PHP, Python, and Node.js. // V materialakh 17-i Mezhdunarodnoi konferentsii IEEE po vychislitel'noi nauke i inzhenerii (CSE), 2014. S. 661-668. DOI: 10.1109/CSE.2014.142.
 13. Suvorov D. A. Izmerenie tekushchego bystrodeistviya protsessorov i kachestva programm. Sposoby otsenki i povysheniya real'noi proizvoditel'nosti //Otkrytoe obrazovanie. 2009. № 6. S. 59-65.
 14. Pentkovskii V. M. i dr. Metodologiya otsenki proizvoditel'nosti vychislitel'nykh sistem //Aktual'nye problemy sovremennoi nauki. 2012. № 6. S. 358-363.
 15. Zhuikov R., Sharygin E. Metody predvaritel'noi optimizatsii programm na yazyke JavaScript //Trudy Instituta sistemnogo programmirovaniya RAN. 2015. Т. 27. № 6. S. 67-86.
 16. B. Basumatary, N. Agnihotri BENEFITS AND CHALLENGES OF USING NODE.JS // International Journal of Innovative Research in Computer Science & Technology. 2022. № 10-3. S. 67-70. URL: <https://acspublisher.com/journals/index.php/ijrcst/article/view/10433/9623>

The use of artificial intelligence systems for data processing in the educational process

Kopysheva Tatyana Nikolaevna 

PhD in Physics and Mathematics

Associate Professor; Department of Mathematical and Hardware Support of Information Systems; I.N. Ulyanov Chuvash State University

428015, Russia, Republic of Chuvashia, Cheboksary, Moskovsky ave., 15, office B-305

✉ tn_pavlova@mail.ru

Mitrofanova Tatiana Valerievna 

PhD in Physics and Mathematics

Associate Professor; Department of Mathematical and Hardware Support of Information Systems; I.N. Ulyanov Chuvash State University

428015, Russia, Republic of Chuvashia, Cheboksary, Moskovsky str., 15, office B-304

✉ mitrofanova_tv@mail.ru

Smirnova Tatiana Nikolaevna 

PhD in Physics and Mathematics

Associate Professor; Department of Mathematical and Hardware Support of Information Systems; I.N. Ulyanov Chuvash State University

428015, Russia, Republic of Chuvashia, Cheboksary, Moskovsky ave., 15, office B-304

✉ smirnova-tanechka@yandex.ru

Khristoforova Anastasiia Vladimirovna

PhD in Physics and Mathematics

Associate Professor; Department of Mathematical and Hardware Support of Information Systems; I.N. Ulyanov
Chuvash State University

428015, Russia, Republic of Chuvashia, Cheboksary, Mskovskyst., 15, office B-304

✉ dlya.nastenki@mail.ru



Abstract. In the Russian Federation, much attention is paid to the development of end-to-end digital technologies, including artificial intelligence (AI) technologies. Decree of the President of the Russian Federation No. 490 dated October 10, 2019 "On the development of artificial intelligence in the Russian Federation" approved the National Strategy for the Development of Artificial Intelligence for the period up to 2030. In accordance with subparagraph (c) of paragraph 51.5 of the Strategy, one of the directions for improving the level of AI competencies and the level of awareness of citizens about AI is the development of skills in using AI technologies among graduates of educational institutions of higher education through the inclusion of AI modules in each educational program. The object of the study of this article is the use of AI systems in the course of laboratory and practical classes, as well as independent work of students in the disciplines "Artificial intelligence systems", "Fundamentals of artificial intelligence" and similar disciplines. The subject of the research is the methods of solving problems of approximation of functions and classification of data by means of specialized platforms Loginom Community and Neural Network Wizard. The methods of training a neural network, methods for evaluating the quality of training samples of a neural network are considered in detail. The research methodology is based on a combination of theoretical and practical approaches using methods of analysis, comparison, generalization, synthesis, classification, and modeling. Materials have been developed, during the study of which students should gain theoretical knowledge in the field of function approximation and data classification, familiarize themselves with the basic concepts and methods of artificial intelligence, as well as their application in various fields of data processing, consolidate practical skills in working with neural networks, as well as specialized platforms and tools Loginom Community and Neural Network Wizard. The results of the control measures showed that the skills of using artificial intelligence technologies were formed at a sufficient level among students of training areas not related to the field of artificial intelligence.

Keywords: regression, training sample, Neural Network Wizard, Loginom Community, model training, data processing, data classification, function approximation, neural networks, artificial intelligence

References (transliterated)

1. Komp'yuternoe zrenie v teorii i na praktike / V. A. Artem'ev, T. N. Kopysheva, T. V. Mitrofanova [i dr.]. // Innovatsii v informatsionnykh tekhnologiyakh, mashinostroenii i avtotransporte : Sbornik materialov V Mezhdunarodnoi nauchno-prakticheskoi konferentsii, Kemerovo, 19–20 oktyabrya 2021 goda / Redkollegiya: D.M. Dubinkin (otv. red.) [i dr.]. Kemerovo: Kuzbasskii gosudarstvennyi tekhnicheskii universitet imeni T.F. Gorbacheva, 2021. S. 47–50. EDN UQSPNB.
2. Lavrent'ev, L. F. Finansovoe prognozirovanie na osnove apparata neironnykh setei / L. F. Lavrent'ev, V. P. Filippov // Vestnik Rossiiskogo universiteta kooperatsii. 2014. № 2(16). S. 122–127. EDN QNPSGO.
3. Osobennosti prognozirovaniya sportivnykh sobytii na osnove ispol'zovaniya apparata

- neironnykh setei / M. S. Portnov, A. V. Rechnov, V. P. Filippov [i dr.]. // Vestnik Rossiiskogo universiteta kooperatsii. 2019. № 2(36). S. 76–79. EDN KIZOXH.
4. Wheat Yield Estimation Based on Unmanned Aerial Vehicle Multispectral Images and Texture Feature Indices / Y. Kang, Ya. Wang, Ya. Fan [et al.] // Agriculture. 2024. Vol. 14, No. 2. P. 167. DOI: 10.3390/agriculture14020167. EDN PRFCIE.
 5. Vayssade, J. A. Wizard: Unsupervised goats tracking algorithm / J. A. Vayssade, X. Godard, M. Bonneau // Computers and Electronics in Agriculture. 2023. Vol. 209. P. 107831. DOI: 10.1016/j.compag.2023.107831. EDN TIHIOK.
 6. Kafiev, I. R. K voprosu avtomatizatsii protsessa sortirovki belykh gribov s ispol'zovaniem neironnykh setei / I. R. Kafiev, P. S. Romanov, I. P. Romanova // Vestnik NGIEI. 2024. № 4(155). S. 34-49. DOI: 10.24412/2227-9407-2024-4-34-49. EDN DENZQG.
 7. Audio based depression detection using Convolutional Autoencoder / S. Sardari, B. Nakisa, M. N. Rastgoo, P. Eklund // Expert Systems with Applications. 2022. Vol. 189. P. 116076. DOI: 10.1016/j.eswa.2021.116076. EDN JXRGNN.
 8. Faye, G. Localized states in an unbounded neural field equation with smooth firing rate function: a multi-parameter analysis / G. Faye, Ja. Rankin, P. Chossat // Journal of Mathematical Biology. 2013. Vol. 66, No. 6. P. 1303-1338. DOI: 10.1007/s00285-012-0532-y. EDN GPURTX.
 9. Improving knowledge-based dialogue generation through two-stage knowledge selection and knowledge selection-guided pointer network / M. Liu, P. Zhao, J. Liu [et al.] // Journal of Intelligent Information Systems. 2022. Vol. 59, No. 3. P. 591-611. DOI: 10.1007/s10844-022-00709-5. EDN ULVRYK.
 10. Minsheng, L. Application of interactive information system in college personnel management by using BP neural network algorithm / L. Minsheng // Soft Computing-A Fusion of Foundations, Methodologies and Applications. 2023. DOI: 10.1007/s00500-023-08617-8. EDN CTVCYU.
 11. Primenenie sistem iskusstvennogo intellekta dlya obrabotki dannykh : praktikum / M. S. Portnov, A. V. Rechnov, T. N. Smirnova [i dr.]. Cheboksary: Izd-vo Chu-vash. un-ta, 2023. 52 s.
 12. Gorban', A. N. Obobshchennaya approksimatsionnaya teorema i vychislitel'nye vozmozhnosti neironnykh setei / A. N. Gorban' // Sibirskii zhurnal vychislitel'noi matematiki. 1998. T. 1, № 1. S. 11-24. EDN PRZOML.
 13. Prognozirovanie srednesutochnoi temperatury vozdukha v Bishkeke s pomoshch'yu neironnykh setei v srede Neural Network Wizard / A. A. Nasirov, E. M. Imamaliyev, T. T. Talantbekov, A. Zh. Andakulov // Nauka i innovatsionnye tekhnologii. 2022. № 2(23). S. 138-143. DOI: 10.33942/sititpr202265. EDN OVGMTA.
 14. Stupnikov, A. V. Prognozirovanie tseny legkovykh avtomobilei s pomoshch'yu neironnykh setei v srede Neural Network Wizard / A. V. Stupnikov, R. I. Bazhenov // Sovremennaya tekhnika i tekhnologii. 2015. № 7(47). S. 3-10. EDN UHWDBB.
 15. Kozich, P. A. Primenenie regressionnogo analiza i neironnykh setei dlya postroeniya modelei rynka smartfonov Honor / P. A. Kozich, R. I. Bazhenov // Postulat. 2019. № 1-1(39). S. 23. EDN ZAHJUT.
 16. Klinskii, S. D. Prognozirovanie kotirovok valyut na finansovoi birzhe / S. D. Klinskii // Postulat. 2019. № 12(50). S. 9. EDN TAFBLL.
 17. Markevich, D. V. Integratsiya sistem biznes-analitiki s sistemami upravleniya bazami dannykh na transporte / D. V. Markevich, V. V. Kharlanova, A. D. Khomonenko // Naukoemkie tekhnologii v kosmicheskikh issledovaniyakh Zemli. 2023. T. 15, № 2. S.

- 41-48. DOI: 10.36724/2409-5419-2023-15-2-41-48. EDN XAPPBD.
18. Evdokimova, S. A. Analiz tovarnogo assortimenta zapasnykh chastei dilerskogo predpriyatiya avtomobil'nogo servisa s pomoshch'yu algoritma FP-Growth / S. A. Evdokimova, K. V. Frolov, A. I. Novikov // Modelirovanie sistem i protsessov. 2022. T. 15, № 4. S. 24-33. – DOI: 10.12737/2219-0767-2022-15-4-24-33. EDN JCNHGB.
19. Popova, S. A. Razrabotka modeli sotsial'nogo reitinga i ee neirosetevaya realizatsiya v Loginom Community / S. A. Popova, D. B. Vladimirova // Ustoichivoe razvitie obshchestva: novye nauchnye podkhody i issledovaniya : Sbornik nauchnykh trudov po materialam V Mezhdunarodnoi nauchno-prakticheskoi konferentsii, Moskva, 10 aprelya 2024 goda. – Moskva: Tsentr razvitiya obrazovaniya i nauki, 2024. S. 151-159. EDN TSVPXH.
20. Evdokimova, S. A. Primenenie algoritmov klasterizatsii dlya analiza clientskoi bazy magazina / S. A. Evdokimova, A. V. Zhuravlev, T. P. Novikova // Modelirovanie sistem i protsessov. 2021. T. 14, № 2. S. 4-12. DOI: 10.12737/2219-0767-2021-14-2-4-12. EDN DXGWQN

The Role of LLM in Next-Generation Integrated Development Environments

Ishankhonov Azizkhon YUnushon 

Graduate student; Department of Non-Ferrous Metals Metallurgy; National Research Technological University 'MIS'

119049, Russia, Moscow, Leninsky Prospekt, 4, building 1

✉ m180119@edu.misis.ru

Pshychenko Dmitrii Viktorovich 

independent researcher

Shabolovka str., 26-28, Moscow, 119049, Russia

✉ dmitry.pshychenko@rambler.ru

Mozharovskii Evgenii Aleksandrovich 

independent researcher

119991, Russia, Moscow, Leninskie gory str., 1

✉ mozharovsky_ea@rambler.ru

Aluev Andrei Sergeevich 

Master's degree; Ural Federal University

620062, Russia, Yekaterinburg, Mira str., 19

✉ aluev_andrei@rambler.ru

Abstract. The role of Large Language Models (LLM) in new generation integrated development environments (IDEs). Tools such as GitHub Copilot, IntelliCode and Alice Code Assistant are explored in the context of their use in programming. The authors examine how LLMs enable the automation of key development tasks, including code autocompletion, error detection, refactoring, and code generation, which result in increased development efficiency and improved code quality. Special emphasis is placed on how LLMs affect developers' cognitive processes, such as problem-solving abilities, creativity, and professional skills. A review of existing integrated development environments that utilize large language models. LLM

functionality for code autocompletion, fragment generation, error detection and correction was evaluated. Comparative methods were applied to evaluate the effectiveness of LLM compared to traditional development tools. Special attention was paid to analyzing the cognitive load caused by the use of LLMs and assessing their impact on the creative process. The novelty of the research consists in the complex analysis of LLM application in modern IDEs, as well as in revealing their potential for increasing developers' productivity and improving the quality of program code. It is concluded that LLM integration into IDEs allows not only speeding up the process of code creation, but also considerably increasing its quality due to intellectual support and automation of the routine tasks. However, while the benefits of integrating LLMs into IDEs are clear, limitations related to cognitive load, ethical issues, data security, and the need to maintain a balance between automation and development of programmers' skills are also identified.

Keywords: development process optimization, machine learning, software systems, code completion, artificial intelligence, code improvement, programming automation, Integrated Development Environments, Large Language Models, data analysis

References (transliterated)

1. Ivanov K. N., Zakharova O. I. Obrabotka estestvennogo yazyka. Primenenie yazykovykh modelei // Aktual'nye problemy informatiki, radiotekhniki i svyazi. – 2023. – S. 155-156.
2. Korostin O. Comparative analysis of NLP algorithms for optimizing communications in the maritime industry // Journal of science. Lyon. – 2024. – № 56. – C. 19-22.
3. Qin Z., Yang S., Zhong Y. Hierarchically gated recurrent neural network for sequence modeling // Advances in Neural Information Processing Systems. – 2024. – V. 36.
4. Uzkikh G. Yu. Primenenie transformerov v obrabotke estestvennogo yazyka // Vestnik nauki. – 2024. – T. 4. – № 8 (77). – S. 186-189.
5. Gweon H., Schonlau M. Automated classification for open-ended questions with BERT // Journal of Survey Statistics and Methodology. – 2024. – V. 12. – № 2. – P. 493-504.
6. Liukko V., Knappe A., Anttila T., Hakala J. ChatGPT as a Full-Stack Web Developer // Generative AI for Effective Software Development. – Cham: Springer Nature Switzerland. – 2024. – P. 197-215.
7. Ponomarev E. Optimizing android application performance: modern methods and practices // Sciences of Europe. – 2024. – № 149. – C. 62-64.
8. Makar'yan O. S. Razrabotka programmnoho obespecheniya s ispol'zovaniem iskusstvennogo intellekta // Vestnik magistratury. – 2024. – S. 23.
9. Bobunov A. Yu. Cravnenie praktik avtomatizatsii testirovaniya v traditsionnykh bankakh i fintekh-kompaniyakh // Dnevnik nauki. 2024. № 8 [Elektronnyi resurs]. URL: <http://www.dnevniknauki.ru/images/publications/2024/8/technics/Bobunov.pdf>
10. Koyanagi K., Wang D., Noguchi K., i t. d. Exploring the effect of multiple natural languages on code suggestion using github copilot // 2024 IEEE/ACM 21st International Conference on Mining Software Repositories (MSR). – 2024. – P. 481-486.
11. Oh S., Lee K., Park S., Kim D. Poisoned chatgpt finds work for idle hands: exploring developers' coding practices with insecure suggestions from poisoned ai models // 2024 IEEE Symposium on Security and Privacy (SP). – 2024. – P. 1141-1159.
12. Zhikulina K. P., Perfil'eva N. V., Man' L. Tsifrovoy strat paradigmy yazyka // Vestnik Rossiiskogo universiteta druzhby narodov. Seriya: Teoriya yazyka. Semiotika. Semantika. – 2024. – T. 15. – № 2. – S. 364-375.

13. Pekareva, V. V. Semanticheskii analiz definitsii «informatsiya» v tselyakh sistemizatsii podkhodov i faktorov obespecheniya informatsionnoi bezopasnosti v usloviyakh tsifrovizatsii / V. V. Pekareva, Yu. I. Frolovskaya // Agrarnoe i zemel'noe pravo. – 2024. – № 3(231). – S. 89-92. – DOI: 10.47643/1815-1329_2024_3_89. – EDN PQGQDB
14. Verner D. Integration of artificial intelligence in backend development // Annali d'Italia. – 2024. – № 59. – P. 88-91.

Improving the Efficiency of Software Development Processes: Container Technologies

Sheinman Vered 

Independent researcher

199 Abba Hoshi Street, Haifa, Israel, 3498838

✉ vered.sheinman@gmail.com

Starikov Dmitrii Dmitrievich 

Student; Institute No. 8 'Computer Science and Applied Mathematics'; Moscow Aviation Institute (National Research University)

125993, Russia, Moscow, Volokolamsk highway, 4

✉ dd.starikov@gmail.com

Tiumentsev Denis Viktorovich 

independent researcher

40B Klyuchevskaya str., Ulan-Ude, Republic of Buryatia, 670013, Russia

✉ tyumencev_dv@rambler.ru

Vavilov Georgiy Davidovich 

Student; Department of Radiotherapy; National Medical Research Center named after Academician E.N. Meshalkin

15 Rechkunovskaya str., Novosibirsk, 630055, Russia

✉ Frost20@narod.ru

Abstract. The article discusses the impact of containerized technologies on software development processes. It focuses on the role of containerization in optimizing the deployment and management of applications, as well as in increasing the flexibility and scalability of software systems. The study analyzes key aspects of containerization, including application isolation, increasing software portability between different environments, and reducing operating costs by optimizing the use of computing resources. Modern tools such as Docker and Kubernetes, which allow standardizing and automating the processes of infrastructure deployment and management, are considered. To analyze the effectiveness of container technologies, benchmarking techniques have been used to evaluate their impact on infrastructure flexibility and software system performance. The sources of data were scientific publications. The novelty of the research lies in considering the application of container technologies in the context of modern software development practices, which allows to significantly accelerate the processes of development, testing and deployment of software products. The results show that containerization improves system performance, simplifies application management, and reduces operational costs. Examples of practical use of Docker and Kubernetes in large companies demonstrate that containerization significantly increases infrastructure flexibility and scalability of solutions, allowing developers to easily adapt to

changing conditions and market requirements. In conclusion, it is emphasized that container technologies play a key role in modern software development processes, and their further development will contribute to even more significant improvements in automation and infrastructure management of software systems.

Keywords: platforms, automation, software operation, resource optimization, scalability, Kubernetes, software development, Docker, containerization, process isolation

References (transliterated)

1. Beloded N.I., Demidenko K.G. Razvitie i primeneniye tekhnologii konteinerizatsii v razrabotke programmnoy obespecheniya // Aktual'nye problemy nauchnykh issledovaniy: teoreticheskie. – 2023. – S. 57.
2. Bondarenko A.S., Zaytsev K.S. Using container management systems to build distributed cloud information systems with microservice architecture // International Journal of Open Information Technologies. – 2023. – V. 11. – № 8. – P. 17-23.
3. Mozharovskii E.A. Razrabotka mobil'nykh prilozhenii: ot idei do rynka // Sovremennyye nauchnye issledovaniya i innovatsii. – 2024. – № 1.
4. Aluev A. Scalable web applications: a cost-effectiveness study using microservice architecture // Cold Science. – 2024. – № 8. – C. 32-38.
5. Muzumdar P., Bhosale A., Basyal G., Kurian G. Navigating the Docker ecosystem: a comprehensive taxonomy and survey // arXiv preprint arXiv:2403.17940. – 2024.
6. Christudas B.A. Introducing Docker // Java Microservices and containers in the Cloud: with Spring Boot, Kafka, PostgreSQL, Kubernetes, Helm, Terraform and AWS EKS. – Berkeley, CA: Apress, 2024. – P. 281-343.
7. Higgins T., Jha D.N., Ranjan R. Swarm Storm: an automated chaos tool for docker swarm applications // Proceedings of the 33rd International Symposium on High-Performance Parallel and Distributed Computing. – 2024. – P. 367-369.
8. Mironov T.O. Postroyeniye informatsionnoy arkhitektury sistemy avtomatizatsii tsikla vypuska programmnoy obespecheniya // Inzhiniring predpriyatii i upravleniye znaniyami. – S. 280.
9. Dudak A.A. Sravnitel'nyi analiz instrumentov razrabotki dlya sistem upravleniya proektami: preimushchestva stek-tekhnologii TypeScript i React // Novaya nauka: ot idei k rezul'tatu. – 2024. – № 9. – S. 32-40.
10. Glumov K.S. Shablony, lezhashchie v osnove java, kubernetes i sovremennykh raspredelennykh sistem // Khlebopechenie Rossii. – 2024. – T. 68. – № 1. – S. 6-12.
11. Poggiani L., Puliafito C., Virdis A., Mingozzi E. Live Migration of Multi-Container Kubernetes Pods in Multi-Cluster Serverless Edge Systems // Proceedings of the 1st Workshop on Serverless at the Edge. – 2024. – P. 9-16.
12. Stephey L., Canon S., Gaur A., Fulton D., Younge A. Scaling Podman on Perlmutter: Embracing a community-supported container ecosystem // 2022 IEEE/ACM 4th International Workshop on Containers and New Orchestration Paradigms for Isolated Environments in HPC (CANOPIE-HPC). – IEEE, 2022. – P. 25-35.
13. Sidorov D. Leveraging web components for scalable and maintainable development // Sciences of Europe. – 2024. – № 150. – P. 87-89.
14. Cherednikov K. A., Lavrova E. D., Marukhlenko A. L. Sovremennyy vzglyad na konteinerizatsiyu // Sovremennyye informatsionnyye tekhnologii i informatsionnaya bezopasnost'. – 2023. – P. 115-119.

15. Erdenebat B., Bud B., Kozsik T. Challenges in service discovery for microservices deployed in a Kubernetes cluster – a case study // Infocommunications Journal. – 2023. – V. 15. – № SI. – P. 69-75.
16. Makarova N.V., Savichev D.E. Primenenie metodov iskusstvennogo intellekta pri ekspluatatsii programmogo obespecheniya // Aktual'nye problemy ekonomiki i upravleniya. – 2023. – № 1. – S. 17.
17. Kosarev V.E., Dobridnik S.L. Prakticheskie aspekty razrabotki i vnedreniya tsifrovogo rublya v bankovskie informatsionnye sistemy // Innovatsii i investitsii. – 2023. – № 2. – S. 143-149.

Object-transactional models of programs in algorithmic languages

Pekunov Vladimir Viktorovich

Doctor of Technical Science

Software Engineer, JSC "Informatika"

✉ pekunov@mail.ru



Abstract. This paper is devoted to the issue of representability of programs written in algorithmic languages using formalisms based on the idea of using marginal partially transactional memory, including a single transactional cell and many ordinary cells. It is assumed that such formalisms are based on the concept of a network of objects representing both the main and auxiliary elements of the solving problem. Objects function in memory of the specified type, executing methods containing exclusively branching code devoid of cycles. Cycles in this approach are replaced by multiple special object network renegotiation, similar to that implemented in classical transactional memory. Based on the most general ideas about the process of solving a problem in a certain subject area, the concept of an object-transactional model is introduced for the first time and their basic properties are formulated. The methods of discrete mathematics and the theory of algorithms are used in the formulation of the structure and basic principles of the functioning of object-transactional models. The concept of marginal partially transactional memory containing a single transactional cell with a special agreement is introduced. The features of matching such memory in the context of the proposed models are described. A hypothesis is put forward about the feasibility of arbitrary algorithms using object-transactional models. The basic principles of the functioning of such models are described, and their basic properties are formulated. The concepts of marginal non-parallel and parallel models are introduced. It is proved that the limiting nonparallel model is capable of executing an arbitrary Turing-solvable algorithm. It is proved that the limiting parallel model of $K+2$ nodes is equivalent to a system of K parallel running Turing machines and, accordingly, is capable of executing an arbitrary Turing-solvable algorithm implying the presence of K parallel branches. Thus, the hypothesis put forward in the paper on the feasibility of arbitrary algorithms has been proved.

Keywords: parallel algorithm, theory, Turing machine, object-transaction model, limit theorems, special reconciliation, transactional memory, program's model, sequential algorithm, feasibility of algorithms

References (transliterated)

1. Chernyak, L. Tranzaktsionnaya pamyat'-pervye shagi / L. Chernyak // Otkrytye sistemy. SUBD. – 2007. – № 4. – S. 12-15.
2. Marathe, V.J., Scherer, W.N., Scott, M.L. (2005). Adaptive Software Transactional Memory. In: Fraigniaud, P. (eds) Distributed Computing. DISC 2005. Lecture Notes in Computer Science, vol 3724. Springer, Berlin, Heidelberg.
https://doi.org/10.1007/11561927_26
3. M. Herlihy and J. E. B. Moss. Transactional memory: Architectural support for lock-free data structures. In A. J. Smith, editor, Proceedings of the 20th Annual International Symposium on Computer Architecture. San Diego, CA, May 1993, pages 289–300. ACM, 1993.
4. V. Luchangco and V. J. Marathe. Transaction communicators: Enabling cooperation among concurrent transactions. In Proceedings of the 16th ACM Symposium on Principles and Practice of Parallel Programming, PPOPP '11, pages 169-178, New York, NY, USA, 2011. ACM.
5. Miculan, M., Peressotti, M. Software Transactional Memory with Interactions. Proceedings of the 21st Italian Conference on Theoretical Computer Science, Ischia, Italy, September 14-16, 2020, pp. 67-80.
6. Pekunov V.V. Sverkhoptimistichnye vychisleniya: kontseptsiya i aprobatsiya v zadache o modelirovanii elektrostaticheskoi linzy // Programmnye sistemy i vychislitel'nye metody. – 2020. № 2. – S. 37-44. DOI: 10.7256/2454-0714.2020.2.32232. URL: https://e-notabene.ru/ppsvm/article_32232.htm

Methodology for conducting an experimental study on the perception of visual information in the field of human peripheral vision

Kheyfits Antonina Evgenievna 

Assistant, Higher School of Design and Architecture, Peter the Great St. Petersburg Polytechnic University

195251, Russia, Leningradskaya oblast', g. Saint Petersburg, ul. Politekhnikeskaya, 29

✉ antoni.t-h@mail.ru

Yanchus Viktor Edmundasovich 

PhD in Technical Science

Associate Professor, Higher School of Design and Architecture, Peter the Great St. Petersburg Polytechnic University

195251, Russia, Leningradskaya oblast', g. Saint Petersburg, ul. Politekhnikeskaya, 29

✉ victorimop@mail.ru

Borevich Ekaterina Vladislavovna 

Assistant, Higher School of Design and Architecture, Peter the Great St. Petersburg Polytechnic University

195251, Russia, Leningradskaya oblast', g. Saint Petersburg, ul. Politekhnikeskaya, 29

✉ plasma5210@mail.ru

Abstract. This article describes the method of conducting an experiment in the framework of the study of visual information perception in the field of peripheral vision of a person. The authors describe the methodology of preparation and design of stimulus material, experimental setup, experimental methodology and processing of the data obtained. During the design of the experiment, the question of expanding the reading area of visual

information when working with a computer system is raised. Since the analysis of incoming information was carried out with a gradual complication of the task, the development of a possible solution in the course of work became a production necessity. Any monitor, regardless of size, limits the analysis of visual perception. Incorrect increase or decrease of objects in the discipline format leads to data degradation. A correct image reduces the level of fatigue when reading information, improves the quality of its perception. The peripheral vision area remains unaffected in work processes, being an important part of human vision. The study of the perception of graphic images in the peripheral area of human vision will potentially expand the efficiency of the interface.

The results obtained on the basis of experimental data can be rationally used in the development of human-computer interaction interfaces. The methodology includes the development and consideration of factors of color, size and distance in the stimulus material. The stimulus material is designed using a software module that is subsequently responsible for the random and independent location of the stimulus material. To fix the parameters of the viewing pattern, an AI-tracking software and hardware complex is used.

Keywords: Visual system, Human-Computer Interaction, Eye Tracker, Intelligent Interface, Statistical analysis, Stimulus material, Eye Tracking, Graphical interface, Peripheral vision, Visual perception

References (transliterated)

1. Emel'yanova Yu. G., Fralenko V. P. Metody kognitivno-graficheskogo predstavleniya informatsii dlya effektivnogo monitoringa slozhnykh tekhnicheskikh sistem // Programmnye sistemy: teoriya i prilozheniya. 2018. № 4. S. 117-158.
2. Kognitivnaya mashinnaya grafika v sistemakh kosmicheskogo i meditsinskogo naznacheniya / Pod obshchei redaktsiei doktora tekhnicheskikh nauk, professora M. N. Burdaeva.: LENAND; Moskva, 2018.
3. Sergeev S. F. Intellektnye simbioty organizovannykh tekhnogennykh sred v upravlenii podvizhnymi ob'ektami // Mekhatronika, avtomatizatsiya, upravlenie. 2013. № 9. S. 30-36.
4. Liu Y., Zhou Z., Hu D. Gaze independent brain-computer speller with covert visual search tasks // Clinical Neurophysiology. 2011. № 122. Pp. 1127-1136.
5. McDonald S.A., Carpenter R. H., Shillcock R.C. An anatomically constrained, stochastic model of eye movement control in reading // Psychological review. 2005. № 112. Pp. 814-840.
6. Vitu F., McConkie G.W., Kerr P., O'Regan J.K. Fixation location effects on fixation durations during reading: An inverted optimal viewing position effect // Vision Res. 2001. № 41. Pp. 1513-1533.
7. Yarbus A. L. Eye Movements and Vision. Plenum Press, 1967. DOI: 10.1007/978-1-4899-5379-7
8. Yur'ev F. I. Tsvetovaya obraznost' informatsii. Kiev: Garmoniya sfer, 2007.
9. Barabanshchikov V. A., Milad M. M. Metody okulografii v issledovanii poznavatel'-nykh protsessov i deyatelnosti / V. A. Barabanshchikov, M. M. Milad. Ros. AN. In-t psikhologii, Region. un-t nepreryv. obrazovaniya. – M.: IPRAN, 1994 – 88 c.
10. Glants S. Mediko-biologicheskaya statistika / Glants S.; Per. s angl. d.f.-m.n. Yu.A. Danilova pod red. N.E. Buzikashvili i D.V. Samoilova. – M.: Praktika, 1999 – 459 c.

Modeling irregular-shaped pores based on the brightness of

pixels in a digital image

Damdina Tatiana Tsybikova 

PhD in Technical Science

Associate Professor, East-Siberian State University of Technology and Management

670000, Russia, Republic of Buryatia, Ulan-Ude, Klyuchevskaya str., 40 V

✉ dtatyanac@mail.ru

Damdinov Zorigto Shiripovich 

Master's degree; Department of Software Engineering and Artificial Intelligence; East Siberian State University of Technology and Management

40 Klyuchevskaya str., Ulan-Ude, Republic of Buryatia, 670013, Russia

✉ vdzorigto@mail.ru

Prudova Lyudmila Yur'evna 

PhD in Technical Science

Associate Professor, Department of Engineering and Computer Graphics, East Siberian State University of Technology and Management

40b Klyuchevskaya str., Ulan-Ude, Buryatia region, 670013, Russia

✉ prudova456@mail.ru

Bubeev Innokentii Trofimovich 

PhD in Technical Science

Associate Professor; Department of Engineering and Computer Graphics; East Siberian State University of Technology and Management

40b Klyuchevskaya str., Ulan-Ude, Republic of Buryatia, 670013, Russia

✉ it_bubeev@mail.ru

Abstract. One of the current directions in the field of materials science is the study and modeling of processes occurring in capillary-porous bodies. These facilities play an important role in areas such as oil production, medicine, construction, production of filters and batteries, and the creation of new materials, where accurate understanding and control of processes occurring in porous media is required. Modern modeling methods using computer vision and high computing power of computers can significantly improve the accuracy and efficiency of studies of capillary-porous bodies and processes in them. For porous bodies of irregular structure, accurate visual fixation of processes occurring inside objects is difficult, but there is a need for this especially for multilevel processes that affect the state of a porous body. The proposed simulators and models of the structure of such objects using simplifications and abstract models to assess the interaction of substances (diffusion, percolation, etc.) are unsuitable in heterogeneous and irregular structures. To obtain a geometric model of arbitrary-shaped pores, it is proposed to use digital image processing methods. An array of points describing the geometry of the pore has been created. A method of surface modeling based on spline interpolation of selected points is was chosen. To obtain a surface model of an irregular-shaped pore, a technique has been developed for forming an array of points based on the brightness levels of pixels of its digital image. The brightness levels for grouping the initial pore points are determined based on the local maxima of the brightness histogram. An algorithm for constructing a surface model of an irregular-shaped pore using spline interpolation of point arrays has been developed. The result of the software implementation of the developed algorithm for modeling the surface of irregular pores is presented. According to

the developed method of geometric modeling of irregular pores, it is possible to obtain further information for analysis on the size and volume of pores, porosity of the object as a whole, as well as to analyze processes in porous objects. If necessary, to increase the accuracy of the shape, the number of brightness levels for grouping points can be increased.

Keywords: histogram, array of points, geometric modeling, irregular shape, surface modeling, spine interpolation, digital image processing, porosity, porous materials, pixel brightness level

References (transliterated)

1. Lykov, A. V. Yavleniya perenosa v kapillyarno-poristyykh telakh / A. V. Lykov. – M.: Gosudarstvennoe izdatel'stvo tekhniko-teoreticheskoi literatury, 1954. – 296 s.
2. A.V. Medvedeva, D.M. Mordasov, M.M. Mordasov. Klassifikatsiya metodov kontrolya poristosti materialov // Vestnik TGTU. 2012. Tom 18. № 3.
3. Berkman, A. S. Struktura i morozostoikost' stenovykh materialov / A. S. Berkman, I. G. Mel'nikova. – M.: Gosstroizdat, 1962. – 166 s.
4. Fandeev, V. P. Metody issledovaniya poristyykh struktur // Internet-zhurnal «NAUKOVEDENIE». – 2015. – T. 7, № 4.
5. Kheifets, L. I. Mnogofaznye protsessy v poristyykh telakh / L. I. Kheifets, A. V. Neimark. – M.: Khimiya, 1982. – 320 s.
6. A. G. Anisovich, A. S. Buinitskaya. Standartnye metody opredeleniya poristosti materialov (obzor). // Vestsi natsyyanal'nai akademii navuk Belarusi, 2015. № 2. URL: <https://vestift.belnauka.by/jour/article/view/103/104>
7. Gonsales, R. Tsifrovaya obrabotka izobrazhenii v srede MATLAB / R. Gonsales, R. Vuds, S. Eddins. – M.: Tekhnosfera, 2006. – 616 s.
8. Suleimanov K.A., Lesovik V.S., Pogorelova I.A., Ryabchevskii I.S. Issledovanie makroporistoi struktury yacheistogo betona //Vestnik BGTU im. V.G. Shukhova 2024, № 3. DOI: 10.34031/2071-7318-2024-9-3-8-16.
9. Tarasevich, Yu. Yu. Perkolyatsiya: Teoriya, prilozheniya, algoritmy / Yu. Yu. Tarasevich. – M.: Editorial URSS, 2002.
10. Levandovskii, A. N. Modelirovanie razrusheniya poristogo materiala / A. N. Levandovskii, B. E. Mel'nikov, A. A. Shamkin // Inzhenerno-stroitel'nyi zhurnal. – 2017. – № 1 (69). – S. 3-22.
11. Alekseev, M. V. O raschete iskhodnykh dannykh dlya modelirovaniya radiatsionno-indutsirovannykh effektov v materialakh poristogo tipa / M. V. Alekseev [i dr.] // Preprinty IPM im. M. V. Keldysha. – 2018. – № 208. – 21 s. – DOI: 10.20948/prepr-2018-208.
12. Damdinova T.Ts., Ayusheev T.V., Bal'zhinimaeva S.M., Abatnin A.A. Modelirovanie tel so sfericheskimi porami metodom obobshchennoi lineinoi interpol'yatsii // Programmnye sistemy i vychislitel'nye metody. 2022. № 2. S. 42-51. DOI: 10.7256/2454-0714.2022.2.38262 EDN: ZTFTKU URL: https://e-notabene.ru/itmag/article_38262.html
13. Bal'zhinimaeva S.M., Damdinova T.Ts., Ayusheev T.V., – Modelirovanie tel so sfericheskimi porami metodom obobshchennoi lineinoi interpol'yatsii. // Svidetel'stvo ob ofitsial'noi registratsii programmy dlya EVM № 2022663716 ot 19.07.2022. Zayavka №2022662918 ot 08.07.2022 g.
14. Golovanov, N. N. Geometricheskoe modelirovanie / N. N. Golovanov. – M.: Izdatel'stvo fiziko-matematicheskoi literatury, 2002. – 472 s.
15. Foks, A. Vychislitel'naya geometriya. Primenenie v proektirovanii i na proizvodstve / A.

- Foks, M. Pratt; per. s angl. – M.: Mir, 1982. – 304 s.
16. Poverkhnostnoe modelirovanie: osnovy, primeneniye i preimushchestva // Nauchnye Stat'i.Ru – portal dlya studentov i aspirantov. – URL: <https://nauchniestati.ru/spravka/poverkhnostnoe-modelirovanie>
 17. Kosnikov, Yu. N. Poverkhnostnye modeli v sistemakh trekhmernoi komp'yuternoi grafiki: uchebnoe posobie / Yu. N. Kosnikov. – Penza: Penzenskii gosudarstvennyi universitet, 2007. – 60 s.
 18. Gruzman, I. S. Tsifrovaya obrabotka izobrazhenii v informatsionnykh sistemakh: uchebnoe posobie / I. S. Gruzman, V. S. Kirichuk, V. P. Kosykh, G. I. Peretyagin, A. A. Spektor. – Novosibirsk: Izdatel'stvo NGTU, 2000. – 168 s.
 19. Damdinova T.Ts., Mosorov V.I., Damdinov Z.Sh., Korobkov K.S.. Issledovanie parametrov diffuzionnykh sloev v protsessakh khimiko-termicheskoi obrabotki metallov po tsifrovym izobrazheniyam mikrostruktur. //Svidetel'stvo ob ofitsial'noi registratsii programmy dlya EVM №2024613959 opubl.19 fevralya 2024 g. Zayavka № 2024612459 ot 09 fevralya 2024 g.
 20. Damdinova T.Ts., Radnaeva V.D., Shalbuev D.V., Sovetkin N.V., Deryabin M.A. Sposob vizualizatsii protsessa propitki kapillyarno-poristykh ob'ektov //Pat. №2817553 opubl.16/04/2024 zayavka №2023126423/28.

The Efficiency of Distributed Caching Platforms in Modern Backend Architectures: A Comparative Analysis of Redis and Hazelcast

Zolotukhina Daria 

independent researcher

394062, Russia, Voronezh region, Voronezh, lane Antokolsky, 4

✉ dar.zolott@gmail.com

Abstract. The object of this study is two caching and distributed data storage systems — Redis and Hazelcast — which are widely used to accelerate data access in high-load applications. This article presents a comprehensive comparative analysis of these systems based on key aspects important for efficient caching: architectural features, memory management models, clustering approaches, fault tolerance mechanisms, and scalability. Special attention is given to investigating caching patterns and support for SQL-like queries. The aim of the work is to provide an in-depth analysis of the advantages and limitations of Redis and Hazelcast in the context of data caching, as well as to identify their strengths and weaknesses under different loads and usage scenarios. The methodology of the research includes a comparative analysis of Redis and Hazelcast based on key aspects, with results presented in the form of a comparative table. Performance testing of CRUD operations was also conducted using automated tests integrated into a Spring Boot application. The study shows that Redis, being a single-threaded system with fast read and write operations, is more efficient for simple, localized applications, while Hazelcast, which supports multi-threading and dynamic clustering, handles large data volumes and distributed tasks more effectively. The author's contribution to the research is a comprehensive comparative analysis of these systems, considering key characteristics such as performance, scalability, and fault tolerance, along with testing their performance in real-world scenarios. The novelty of the research lies in the detailed examination of Redis and Hazelcast for data caching in high-load applications, which will be valuable for the development and optimization of the infrastructure of high-performance distributed systems that require real-time data caching.

Keywords: in-memory data storage, clustering, multithreading, fault tolerance, CRUD operations, performance, Hazelcast, Redis, caching, distributed system

References (transliterated)

1. Carlson J. Redis in Action. New York: Manning, 2013.
2. Johns M. Getting Started with Hazelcast. Birmingham: Packt Publishing, 2015.
3. Billig, V. A. Parallel'nye vychisleniya i mnogopotochnoe programmirovaniye. Moskva: INTUIT, 2016.
4. Kadomskii A. A., Zakharov V. A. Effektivnost' mnogopotochnykh prilozhenii // Nauchnyi zhurnal. 2016. №7 (8). URL: <https://scientificmagazine.ru/images/PDF/2016/8/Nauchnyj-zhurnal-7-8.pdf> (data obrashcheniya: 08.11.2024).
5. Boichenko A. V., Rogozhin D. K., Korneev D. G. Algoritm dinamicheskogo masshtabirovaniya relyatsionnykh baz dannykh v oblachnykh sredakh // Statistika i ekonomika. 2014. №6-2. URL: <https://statecon.rea.ru/jour/article/view/584/566> (data obrashcheniya: 08.11.2024).
6. Filatov A. Yu., Mikheev V. V. Analiz effektivnosti potokovo-lokal'noi sborki musora v raspredelennykh sistemakh khraneniya i obrabotki dannykh // Vestnik SibGUTI. 2022. №1 (57). URL: <https://vestnik.sibsutis.ru/jour/article/view/122/126> (data obrashcheniya: 08.11.2024).
7. Goleva A. I., Storozhenko N. R., Potapov V. I., Shafeeva O. P. Matematicheskoe modelirovanie otkazoustoichivosti informatsionnykh sistem // Vestnik NGU. Seriya: Informatsionnye tekhnologii. 2019. №4. URL: <https://intechngu.elpub.ru/jour/article/view/110/98> (data obrashcheniya: 08.11.2024).
8. Borsuk N. A., Kozeeva O. O. Analiz metodov replikatsii baz dannykh pri razrabotke onlain-servisa // Simvol nauki. 2016. №11-3. URL: <https://os-russia.com/SBORNIKI/SN-2016-11-3.pdf> (data obrashcheniya: 08.11.2024).
9. Intro to Jedis – the Java Redis – URL: <https://www.baeldung.com/jedis-java-redis-client-library> (date of access: 08.11.2024).
10. Predicates API – URL: <https://docs.hazelcast.com/hazelcast/5.5/query/predicate-overview> (date of access: 08.11.2024).