MONETEC 2024

The international Science and Technology Conference
«Modern Network Technologies, MoNeTec-2024»

5-я международная конференция

Современные сетевые технологии

29-31 октября 2024 г.
г. Москва

Сборник трудов

Доклады сессии «Моделирование и анализ трафика гетерогенных каналов»

Стендовые доклады

ОРГАНИЗАТОРЫ

Московский государственный университет имени М.В.Ломоносова

APPLIED RESEARCH CENTER FOR COMPUTER NETWORKS

CONSORTIUM NETWORK AND CLOUD TECHNOLOGIES

ГЕНЕРАЛЬНЫЙ СПОНСОР

ELVEES

СПОНСОРЫ

JOINT INSTITUTE FOR NUCLEAR RESEARCH

Федеральный исследовательский центр Информатика и Управление Российской академии наук

МГУ ВМК

base alt

КРИПТОНИТ

ТЕХНИЧЕСКИЙ СПОНСОР

IEEE Advancing Technology for Humanity

IEEE COMPUTER SOCIETY

ПРИ ПОДДЕРЖКЕ

ИНТЕРНЕТ-ПОРТАЛ Научная Россия

INJOIT

ОТКРЫТЫЕ СИСТЕМЫ Open Systems Publications

Спектр Издательский дом

Современные информационные технологии и ИТ-образование Международный научный журнал

прикладная ИНФОРМАТИК@ научно-практический журнал

125 лет COMPUTING TELECOMMUNICATION AND CONTROL

ИНФОРМАЦИОННО-УПРАВЛЯЮЩИЕ СИСТЕМЫ НАУЧНЫЙ ЖУРНАЛ

ПАРТНЕРЫ

Российская Академия Наук

ОНИТ РАН

ОТДЕЛЕНИЕ МАТЕМАТИЧЕСКИХ НАУК

IEEE Russia Section, Communications Society Chapter

https://monetec.ru

Московский государственный университет имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики
Центр прикладных исследований компьютерных сетей» (ЦПИКС)

«Современные сетевые технологии — 2024»

Пятая международная научно-техническая конференция

Москва, Россия, 29–31 октября, 2024

Сборник трудов

Доклады сессии «Моделирование и анализ трафика гетерогенных каналов»

Стендовые доклады

Главный редактор
Р. Л. Смелянский

«Modern Network Technologies — 2024» (MoNeTeC)

The Fifth International Science and Technology Conference

Moscow, Russia, October 29–31, 2024

Proceedings

Reports of the session «Modeling and analysis of heterogeneous channel traffic»

Poster presentations

Editor-in-chief
R. L. Smelyansky

Издательство Московского университета
2025

УДК 004.7(063)
ББК 16+32.972.5я431
С 56

«Современные сетевые технологии — 2024» = «Modern Network Technologies — 2024» (MoNeTeC) : Пятая международная научно-техническая конференция, Москва, Россия, 29-31 октября, 2024 : Сборник трудов : Доклады сессии «Моделирование и анализ трафика гетерогенных каналов». Стендовые доклады / главный редактор Р. Л. Смелянский. — Москва : Издательство Московского университета, 2025. — 98, [1] с. : ил. – Электронное издание сетевого распространения.

ISBN 978-5-19-012160-5 (e-book)
DOI 10.55959/MSU012160-5-2024

Пятая международная конференция «Современные сетевые технологии» (MoNeTec-2024) была посвящена ряду актуальных тем для современных сетевых технологий и их приложений: применение методов искусственного интеллекта для управления ресурсами в сетях и облачных средах, 5G/6G сети беспроводной коммуникации, информационная безопасность в программно-конфигурируемых сетях и облачных средах, вопросы моделирования и анализа трафика гетерогенных каналов. Данный сборник трудов содержит публикации, представленные в рамках секции «Моделирование и анализ трафика гетерогенных каналов» и на стендовой сессии.

«Modern Network Technologies — 2024» (MoNeTeC) : The Fifth International Science and Technology Conference, Moscow, Russia, October 29–31, 2024 : Proceedings : Reports of the session «Modeling and analysis of heterogeneous channel traffic» : Poster presentations / Editor-in-chief R. L. Smelyansky. — Moscow : Moscow University Press, 2025. — 98, [1] p. : il. — Electronic edition of network distribution.

The Fifth International Science and Technology Conference «Modern Network Technologies (MoNeTec-2024)» was devoted to the number of hot topics of contemporary network technologies and their applications: AI application for resource management and control, 5G/6G networks for wireless communication, information security in SDN/Cloud, QoS control in data communication, problems of heterogeneous traffic modelling and analysis. This volume of Proceedings contains papers presented on the «Heterogeneous Traffic Modelling and Analysis» session and on the Poster session.

УДК 004.7(063)
ББК 16+32.972.5я431

ISBN 978-5-19-012160-5 (e-book)

Содержание

1. Исследование методов сжатия разреженных кодов в канале с аддитивным белым гауссовским шумом с использованием методов машинного обучения 5
Вотинцев А.К., Волканов Д.Ю.

2. Метод балансировки загрузки очередей маршрутизатора на основе машинного обучения 13
Гарькавый И.С., Степанов Е.П.

3. Калибровка метода оценки плотности транспортного потока с применением широкоугольной камеры 18
Кутейников И.А., Яшина М.В.

4. Эффективные методы сжатия данных при формировании луча в сетях LTE и 5G NR 23
Лысенко Д.Р., Писковский В.О.

5. Прогнозирование временных характеристик прикладных сетевых сервисов 29
Лычева Е.О., Писковский В.О., Могиленец В.М.

6. Задача распределения программы обработки сетевых пакетов на стадии конвейера сетевого процессорного устройства 38
Никифоров Н.И., Волканов Д.Ю.

7. Прогнозирование числовых значений показателей качества обслуживания гетерогенного канала при помощи авторегрессионных моделей 43
Пуртов Б.С., Волканов Д.Ю.

8. Время подготовки к работе сетевого сервиса при использовании разделяемого репозитария образов ВМ 48
Сагалевич В.Д., Писковский В.О.

9. Задача синтеза управления транспортными потоками и ее решение методом машинного обучения 54
Софронова Е.А., Дивеев А.И., Белотелов В.Н., Бобков А.А.

10. Методы организации повторной обработки пакетов в конвейере сетевого процессорного устройства 58
Стамплевский Д.М., Волканов Д.Ю.

11. Сравнение динамики кольчуги Буслаева и цепочки контуров с симметричным или несимметричным расположением узлов 63
Яшина М.В., Таташев А.Г., Кутейников И.А.

12. Защита контейнеризованных сред. Анализ специфики уязвимостей, направленных на нарушение границ контейнера (Container Escape Attacks) и разработка методов их определения с использованием собственных реализаций Container Runtime Interface (CRI) и с Extended Berkeley Packet filter (EBPF) для ядра Linux 68
    a. *Фирсов С.В.*

13. On modern approaches to the design network processor unit 75
Nikiforov N., Volkanov D., Volchaninov A., Alexandrov A.

14. Methods for marking network traffic routes to solve the task of predicting the quality of heterogeneous channels 82
Ryazanov A., Volkanov D.

15. Models for intelligent management telecommunications into transport logistics of data economy 85
Shabanov A., Zatsarinnyy A.

16. Optimizing Parameters for Blockchain Storage for the RunOS SDN Controller 93
Shibaev P., Piskovski V.

Исследование методов сжатия кодов разреженной регрессии в канале с аддитивным белым гауссовым шумом с использованием методов машинного обучения

Вотинцев А.К.
Факультет Вычислительной математики и кибернетики
Московский государственный университет имени М.В.Ломоносова
Москва, Россия
alexeyvotintsev@yandex.ru

Волканов Д.Ю.
Факультет вычислительной математики и кибернетики
Московский государственный университет имени М.В.Ломоносова
Москва, Россия
volkanov@asvk.cs.msu.ru

Аннотация—В статье предлагается новый вид системы связи, использующий декодер приближенной передачи сообщений, AMP в канале с аддитивным белым гауссовым шумом, АБГШ с методами сжатия. Основное внимание нашего исследования уделено изучению производительности декодера AMP в сочетании с пяти различными методами сжатия: сингулярным разложением (SVD), гауссовым проецированием, случайным проецированием, сверточным и рекурентным автокодировщиком.

Экспериментальные исследования включают передачу сжатых измерений через канал АБГШ, где сигналы подвергаются воздействию шума. Посредством симуляций канала и оценок производительности была исследована эффективность декодера AMP при различных уровнях шума и различных коэффициентах сжатия. Анализируются компромиссы между точностью восстановления, скоростью передачи данных и вычислительной сложностью для каждого метода сжатия.

Сжатие сигналов показало многообещающие результаты в уменьшении количества измерений, необходимых для восстановления сигнала.

Index Terms—Аддитивный белый гауссов шум, кодек приблизительной передачи сообщений, AMP, сжатие данных, коды исправления ошибок, случайное проекции, разреженное проекции, сингулярное разложение, метод главных компонент, автокодировщики.

I. Введение

Несомненно, задача точного декодирования информации является одной из важнейших проблем в области связи. В физическом канале, а также в WiFi и других беспроводных каналах, возникают помехи при передаче пакетов. Они проявляются в различных формах, но каждая из них требует эффективного решения.

Быстрое развитие методов кодирования с исправлением ошибок требует от специалистов в области цифровых технологий интенсивного изучения все более эффективных алгоритмов декодирования, обеспечивающих легкость достижения высокой надежности передачи и хранения данных при высоких уровнях шума.

В данном исследовании основное внимание уделяется каналу с АБГШ, который часто используется в спутниковой связи и дорогих каналах, где эффективное использование важно для максимизации пропускной способности канала. Рассматриваются коды разреженной регрессии, появившиеся в начале 21 века и подошедшие лучше других кодов для данного канала к границе Шеннону, задаче, поставленной более 70 лет назад [1]. Цель состоит в достижении "высоконадежного оптимального декодирования с минимальной сложностью в непосредственной близости пропускной способности канала связи."

Глава 2 содержит постановку задачи. В статье канал АБГШ описан в главе 3. В главе 4 представлена информация о декодере AMP, Approximate Message Passing. В глава 5 обзор похожих задач. В главе 6 представлен обзор методов сжатия данных. В главе 7 описана программная реализация. В главе 8 представлены методики проведения экспериментального исследования и результаты исследования методов сжатия с использованием декодера, и на основе этого тестирования сделаны выводы. Глава 9 содержит заключение работы.

II. Постановка задачи

- Дано:
  1) Набор сообщений, исходные данные, представленные в виде вектора $\mathbf{x}$
  2) Канал АБГШ. Это можно записать как:

$$\mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{n}$$

где: - $\mathbf{A}$ - матрица канала или матрица передачи, которая отображает исходные данные в полученные зашумленные данные. - $\mathbf{n}$ -

случайный шум, представляющий АБГШ с нулевым средним и известной дисперсией σ^2.

3) Выбранный кодер, относящийся к кодам разреженной регрессии.
   Алфавит A = $\{a_1,\ a_2...,\ a_n\ \}$ - исходные кодируемые данные. Алфавит B = $\{b_1,\ b_2...,\ b_n\ \}$ - закодированные данные. Алфавитное кодирование - преобразование

$$A \to B, \forall a_i \subset A \to f(A) = f(a_1 \dots a_n) = f(a_1) \dots$$

   Слова в алфавите B называют кодовыми, а их совокупность - кодом
   где $\hat{\mathbf{x}}$ - оценка исходных данных.

- Требуется:
  Предложить алгоритм сжатия данных, передаваемых в канале используемым кодеком.
- Ограничения:
  1) Уменьшить объем передаваемых данных не менее, чем в 2 раза.
  2) Общая погрешность кодека разреженной регрессии и выбранного метода сжатия не должна превышать 10% относительно погрешности кодека разреженной регрессии на тех же самых данных.
  3) Сложность алгоритма не должна увеличиться более чем в 1.5 раза.

III. Канал с аддитивным белым гауссовским шумом

В статье рассматривается канал с аддитивным белым гауссовым шумом, далее АБГШ [2] [3] , который представляет собой тип помех в канале связи. Он характеризуется равномерной, то есть одинаковой на всех частотах, спектральной плотностью мощности, нормально распределенными значениями во времени и аддитивным воздействием на сигнал. Белый шум — это шум, содержащий равномерно распределенные компоненты по всему диапазону частот. Гауссов шум — это шум, имеющий нормальное распределение вероятностей.

Такой шум возникает, например, при передаче данных по радиосвязи или кабельному телевидению. В канале АБГШ каждый отдельный бит или символ может быть искажен шумом, что может привести к ошибкам декодирования.

Канал АБГШ часто используется в исследованиях и при разработке алгоритмов цифровой обработки сигналов и телекоммуникаций.

Канал генерирует выходной сигнал $y \in \mathbb{R}$ из входного сигнала $x \in \mathbb{R}$ согласно

$$y = x + w$$

где w извлекается из гауссовского распределения с нулевым средним и дисперсией σ^2, обозначаемого как $w \sim \mathcal{N}(0, \sigma^2)$. Канал АБГШ не имеет памяти, то есть выходной сигнал y канала зависит только от текущего входного сигнала x, а не от предыдущих входов. Входной сигнал в канал имеет ограничение на среднюю мощность P: если $x_1, x_2, \dots, x_n$ передаются через канал за n использований, то требуется, чтобы

$$\frac{1}{n}\sum_{i=1}^{n} x_i^2 \le P \tag{1}$$

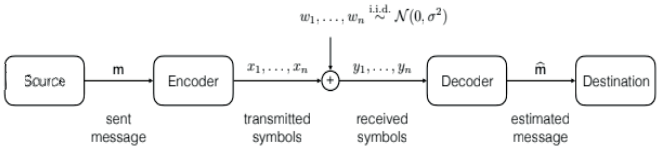

Рисунок 1 - Общая схема передачи сообщения

Для оценки качества передачи данных по каналу АБГШ используется отношение сигнал/шум (SNR), которое определяет соотношение мощности сигнала к мощности шума. Чем выше значение SNR, тем лучше качество передачи данных. Таким образом, отношение сигнал/шум для канала АБГШ равно P/σ^2.

IV. Алгоритм Приближенной Передачи Сообщений, AMP

Алгоритм AMP [4] [5] обладает рядом преимуществ перед другими методами декодирования [6], такими как высокая вычислительная скорость и эффективность при работе с разреженными сигналами. Его также можно использовать для декодирования сигналов с неизвестной структурой, что делает его полезным для широкого спектра задач передачи данных в каналах АБГШ.

A. Разреженные Коды

Коды разреженной регрессии — это набор алгоритмов и методов, которые позволяют решать задачи регрессии, используя разреженные модели [7] [8] . Этот тип кодов относится к категории мягкого декодирования. Разреженные модели характеризуются использованием только небольшого числа признаков для предсказания целевой переменной, в то время как большинство признаков не вносят вклад в предсказание.

1) Преимущества и Недостатки: Разреженные регрессионные коды являются мощным инструментом для передачи данных в каналах связи, особенно в каналах АБГШ, где они могут обеспечить высокую скорость передачи и низкую вероятность ошибок.

Преимущества:

1) Высокая скорость передачи данных: коды могут достигать высоких скоростей передачи данных при оптимальном выборе параметров и матриц.
2) Низкая вероятность ошибки: коды способны обнаруживать и исправлять ошибки при передаче данных через каналы АБГШ.
3) Эффективное использование канала: коды эффективно используют канал связи по сравнению с другими методами кодирования.

Недостатки:

1) Чувствительность к шуму: коды могут быть чувствительны к шуму в канале связи, что может привести к некорректному декодированию данных.
2) Большой объем передаваемых данных и низкая скорость кода (скорость $R = \frac{k}{n}$, характеризующая избыточность, введенную при кодировании).
3) Высокие требования к вычислительной мощности: Эффективное использование кодов требует мощных вычислительных ресурсов для выполнения сложных вычислений на каждом этапе.

B. Кодировщик

Сообщение β представляет собой последовательность передаваемых символов, общее количество которых равно L. Тогда β делится на L секций длиной M, где M — это размер используемого алфавита. Для передачи одного символа конструируется вектор β_i, содержащий $M-1$ нулевых элементов и только 1 оставшийся ненулевой элемент. Этот ненулевой элемент идентифицирует соответствующий символ. Например, рассматривая ASCII-алфавит и букву "C"в латинском алфавите, которая имеет код 67, т.е. занимает 67-ю позицию в алфавите. Тогда для декодирования мы передаем вектор из 257 элементов, где 67-й элемент имеет ненулевое значение, а остальные нулевые:

$$\text{если } x = "C"$$
$$\beta_i = (a_0 \dots a_{65} \dots a_{256}) = (0 \dots a_{66} \dots 0)$$

Это значение a_i определяется как $\sqrt{nP_i}$:

$$a_1 = \sqrt{nP_1}, a_2 = \sqrt{nP_2}, \dots, a_M = \sqrt{nP_M}$$

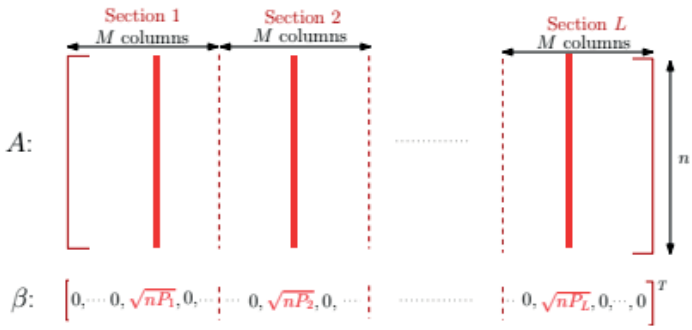

Рисунок 2

Тогда

$$y = \sqrt{nP_1 A_{i_1}} + \sqrt{nP_2 A_{i_2}} + \dots + \sqrt{nP_L A_{i_L}} + \omega = \sum_{j=1}^{n} \sqrt{nP_j A_{i_j}} + \omega (2)$$

Задача декодирования заключается в восстановлении ненулевых позиций $i_1, \dots, i_L$. Идея распределения мощности состоит в том, чтобы облегчить итеративный декодер, который сначала декодирует (точно или приближенно) секции с наибольшей мощностью, затем секции со следующей наибольшей мощностью и так далее.

Тогда

$$P = P_1 + P_2 + \dots + P_L = \sum_{j=1}^{L} P_j \quad (3)$$

C. Декодировщик

Декодирование с использованием алгоритма AMP основано на оценке вероятности того, что каждый бит данных был передан без ошибок. Алгоритм AMP начинается с оценки вероятностей всех битов данных на основе известных входных данных и оценки шума канала. Затем алгоритм AMP обновляет эти оценки вероятностей с использованием рекурсивных формул, которые учитывают ошибки передачи данных и шум канала.

Другими словами, алгоритм декодирования AMP состоит из нескольких итераций, каждая из которых включает следующие шаги:

1) Сжатие принятого сигнала: принятый сигнал сжимается с использованием случайной матрицы.
2) Оценка сигнала: на основе сжатого сигнала выполняется оценка истинного сигнала с использованием матричных операций и формул. Эта оценка не является точной, но содержит информацию о приблизительном значении истинного сигнала.
3) Вычисление ошибки: с использованием полученной оценки сигнала вычисляется ошибка декодирования. Ошибка представляет собой разницу между принятым сигналом и его оценкой. Эта ошибка является важной информацией, используемой в последующих итерациях.
4) Обновление оценки сигнала: сиспользованием ошибки декодирования и матричных операций корректируется и обновляется предыдущая оценка сигнала. Это позволяет уточнять оценку сигнала на каждой итерации.
5) Завершение алгоритма: алгоритм продолжает работать до достижения определенного числа итераций или до тех пор, пока ошибка декодирования не станет достаточно маленькой.

D. Вероятность ошибки

Эффективность декодера измеряется через долю символов, декодированных с ошибками (Symbol Error Rate, SER), которая определяется как доля секций, декодированных с ошибкамии и долей битов, декодированных с ошибками (Bit Error Rate, BER). SER определяется следующим образом:

$$\text{SER} = \frac{1}{L} \sum_{\ell=1}^{L} \mathbb{1}\{\beta_{\text{sec}}^{(\ell)} \neq \hat{\beta}_{\text{sec}}^{(\ell)}\} \quad (4)$$

где $\mathbb{1}\{\cdot\}$ является индикаторной функцией, $\beta_{\text{sec}}^{(\ell)} \in \mathbb{R}^M$ представляет ℓ-ю секцию вектора сообщения, а $\hat{\beta}_{\text{sec}}^{(\ell)}$

является оценкой этой секции, сделанной декодером AMP.

E. Проблемы с декодером AMP

Несмотря на свою эффективность в каналах с АБГШ, декодер AMP [8] [9] имеет свои ограничения и проблемы:

1) Ограниченная производительность при высоких уровнях шума. В ситуациях, когда шум канала очень высок, декодер AMP может испытывать трудности с восстановлением исходного сообщения, что приводит к ошибкам декодирования.
2) Создание значительной избыточности данных. Декодер AMP может вводить значительную избыточность в передаваемые данные.
3) Необходимость правильного выбора параметров. Декодер AMP требует оптимального выбора параметров для каждой конкретной задачи декодирования. Неправильный выбор параметров может привести к низкой производительности и снижению точности декодирования.
4) Высокая вычислительная сложность. В некоторых случаях декодер AMP может требовать значительных вычислительных ресурсов, что может быть проблематично для реализации на мобильных устройствах или других устройствах с ограниченными вычислительными возможностями.

Проблема, которую мы будем рассматривать в данном исследовании, заключается в создании значительной избыточности в передаваемых данных.

V. Обзор похожих задач

A. Измерения сжимающего кодирования с 1 битом

Однобитное сжимающее кодирование (1-bit CS) представляет собой вариант задачи обнаружения сжатого сигнала, которая заключается в восстановлении разреженного сигнала из его бинарных (однобитных) измерений. В традиционных задачах сжатого обнаружения сигналы измеряются в непрерывных значениях (аналоговых), но в случае 1-bit CS каждое измерение является бинарным значением — либо 0, либо 1 [10].

Одним из ключевых аспектов 1-bit CS является выбор оптимального порогового значения для решения задачи восстановления. Это связано с тем, что бинарные измерения могут только указывать на то, что значение сигнала выше или ниже определенного порога.

1) Модель измерений: В классическом однобитовом сжатом обнаружении каждое измерение y_i^* представляет собой знак скалярного произведения разреженного вектора x на вектор измерений a_i:

$$y_i^* = \mathrm{sgn}((a_i, x_i)). \tag{5}$$

В экспериментальной настройке приемник получает измерение y_i, которое соответствует однобитовому измерению y_i^*, искаженному добавлением нулевого белого гауссовского шума w_i с дисперсией ^2_w, т.е., $w_i \sim \mathcal{N}(0,^2_w)$. Это можно записать как

$$y_i = y_i^* + w_i. \tag{6}$$

Вектор измерений y размерности $M \times 1$ компактно выражается как

$$y = \mathrm{sgn}(Ax) + w, \tag{7}$$

где i-я строка $A_{M\times N}$ - это a_i[5].

B. Универсальное удаление шума

Универсальные сети удаления шума (Universal Denoising Networks, UDN) [11] являются типом глубоких нейронных сетей, разработанных для решения задач удаления шума из сигналов, изображений или других данных. Основной особенностью универсальных сетей удаления шума является возможность работы с различными типами шума или сигналов без необходимости предварительной настройки или знания характеристик конкретного шума.

Традиционные алгоритмы удаления шума обычно разрабатываются и оптимизируются для конкретных типов шума, что ограничивает их использование в реальных условиях, где шум может быть разнообразным.

UDN исследуют возможности глубокого обучения для создания универсальной модели удаления шума, способной обрабатывать различные типы шума и сигналов без необходимости специализации для каждого типа. Они обучаются на большом наборе разнообразных шумных данных, чтобы научиться эффективно удалять шум из новых, неизвестных данных.

VI. Сжатие

Сжатие данных — это процесс уменьшения размера данных путем удаления избыточной информации или кодирования ее таким образом, чтобы она занимала меньше места на жестком диске или в памяти и ускоряла передачу данных по сети [12] [13] .

A. Сжатие разреженной матрицы с сохранением разреженного формата

Сжатие разреженной матрицы с сохранением разреженного формата предполагает уменьшение размерности матрицы при сохранении ее разреженной структуры. Такой подход позволяет сократить затраты на хранение и обработку разреженных матриц.

Существует несколько возможных разложений матрицы на составные матрицы. Но только SVD-разложение позволяет уменьшить размерность хранимых данных [14] [15] . SVD — это метод разложения матрицы на три более маленькие матрицы и является одним из ключевых инструментов линейной алгебры и науки о данных.

Для матрицы $M \times N$ A SVD представляет ее в виде произведения трех матриц:

$$A = U \cdot S \cdot V^T \tag{8}$$

Где U — ортогональная матрица размером $M \times M$, S — диагональная матрица размером $M \times N$ с неотрицательными элементами, а V^T — транспонированная ортогональная матрица размером $N \times N$. Элементы на диагонали матрицы S называются сингулярными значениями матрицы A, а столбцы матриц U и V называются соответственно левыми и правыми сингулярными векторами.

Сингулярное разложение позволяет представить матрицу A в виде линейной комбинации базисных векторов, которые наиболее существенно влияют на ее структуру. В частности, SVD выявляет наиболее значимые направления в данных, которые можно использовать для уменьшения размерности или для поиска наиболее важных компонентов в данных.

B. Сжатие измерений

Сжатое восприятие (Compressive Sensing, CS) — это метод обработки сигналов и сжатия данных, основанный на идее получения большого количества информации о сигнале с помощью относительно небольшого числа измерений [16] [17] . Этот метод используется, когда количество измерений ограничено, а сигнал разрежен или имеет разреженное представление в некотором базисе.

Идея, лежащая в основе сжатого восприятия, заключается в получении информации о сигнале с помощью только небольшого числа выбранных измерений, а затем восстановлении исходного сигнала из этих измерений с использованием алгоритмов восстановления сигнала, таких как минимизация L1 или метод наименьших квадратов с ограничениями.

a) Случайные проекции: Метод случайных проекций [18] является одним из наиболее распространенных методов сжатого восприятия на основе идеи случайного проецирования матрицы измерений.

Основная идея случайных проекций заключается в следующем. Имеются многомерные входные данные, и необходимо уменьшить их размерность, чтобы упростить вычисления или извлечь важные признаки. Вместо использования сложных алгоритмов снижения размерности, метод случайных проекций предлагает проецировать входные данные на случайно сгенерированные более низкоразмерные подпространства.

- Принцип работы: проецирует исходные данные на случайно сгенерированные низкоразмерные подпространства.
- Преимущества: простая реализация, низкая вычислительная сложность, сохранение структурных свойств данных.
- Ограничения: может привести к потере информации или искажению структуры данных, если случайная матрица не выбрана правильно.

b) Разреженные гауссовские проекции: Метод [7] генерирует проекции, имеющие разреженную структуру или смещение к нулю. В этом случае большинство элементов проекции равны нулю, и только небольшое количество элементов имеют ненулевые значения. Этот подход позволяет представить данные с использованием значительно меньшего объема информации, снижая размерность и объем данных.

Распределение вероятностей для генерации разреженных проекций основано на гауссовском распределении, но с добавленными ограничениями разреженности. Эти ограничения гарантируют, что большинство элементов проекции равны нулю, в то время как только небольшое количество элементов случайно выбираются из гауссовского распределения.

C. Автокодировщики

Автокодировщики, autoencoders - это класс моделей глубокого обучения, которые используются для обучения эффективного представления данных. Они работают путем сжатия, кодирования входных данных в более низкоразмерное пространство и затем восстановления, декодирования исходных данных из этого сжатого представления [19] [20].

Основная идея автокодировщиков состоит в том, чтобы обучить модель реконструировать входные данные на выходе с минимальной потерей информации. Для этого модель сначала преобразует входные данные в сжатое представление - код, а затем восстанавливает исходные данные из этого сжатого представления.

Автокодировщики могут сжимать данные путем извлечения наиболее информативных признаков из исходных данных и представления их в более компактной форме. Это происходит за счет двух основных механизмов: кодирования и декодирования.

a) Convolutional Autoencoders: Сверточные автокодировщики - это класс моделей глубокого обучения, которые используют комбинацию сверточных слоев и слоев субдискретизации для извлечения пространственных признаков из входных данных и их последующего сжатия [21].

Основные блоки:

1) Сверточный слой, Convolutional Layer. Сверточные слои являются основным строительным блоком CNN. Каждый нейрон в сверточном слое соответствует небольшой области входных данных, и вычисляет свертку этой области с набором локальных фильтров, ядер. Результатом является карта признаков, которая содержит активации признаков на различных уровнях абстракции.
   Входные данные X размером $W_{\text{in}} \times H_{\text{in}} \times D_{\text{in}}$, где W_{in} - ширина, H_{in} - высота, D_{in} - количество каналов. Также ядро свертки K размером $F \times F \times D_{\text{in}}$, где F - размер ядра свертки. Сверточная операция применяется к входным данным следующим образом:

$$(X*K)(i,j,d) = \sum_{l=0}^{F-1} \sum_{m=0}^{F-1} \sum_{n=0}^{D_{\text{in}}-1} X(i+l, j+m, n) \cdot K(l,m,n)$$

2) Слой подвыборки, Pooling Layer. Слой подвыборки используется для уменьшения размерности признаковых карт и создания инвариантности к небольшим пространственным трансформациям входных данных. Обычно используется операция подвыборки, которая объединяет значения признаков в небольших областях и уменьшает размерность карт признаков.
Входные данные X размером $W_{\text{in}} \times H_{\text{in}} \times D_{\text{in}}$. Операция максимальной подвыборки размера $S \times S$ на каждом канале d применяется следующим образом:

$$(X_{\text{pool}})(i,j,d) = \max_{l=0}^{S-1} \max_{m=0}^{S-1} X(i \cdot S + l, j \cdot S + m, d)$$

b) Recurrent Autoencoders: Рекуррентные автокодировщики - это модели глубокого обучения, которые комбинируют в себе принципы автокодировщиков и рекуррентных нейронных сетей, RNN. Они применяются для работы с последовательными данных, такими как аудио, видео, временные ряды и тексты [20] [zou2021channel] [22].

Ключевым компонентом является рекуррентный блок, позволяющий модели учитывать предыдущие состояния при обработке текущего входа. На каждом временном шаге t входной элемент x_t подается на вход рекуррентному слою. Этот вход используется вместе с предыдущим скрытым состоянием h_{t-1} (которое является выходом рекуррентного слоя на предыдущем временном шаге) для вычисления нового скрытого состояния h_t. Математически это может быть представлено как:

$$h_t = f(x_t, h_{t-1})$$

где f - функция активации рекуррентного слоя.

VII. Программная реализация

a) Функция потерь: Функция потерь - это функция, которая измеряет разницу между предсказанными значениями модели и их истинными значениями.

В работе рассматривалась MSE, она измеряет среднее квадратичное отклонение между восстановленными данными и их исходными версиями.

Формула MSE выглядит следующим образом:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

где: n - количество примеров в выборке, y_i - истинное значение, $\hat{y}_i$ - предсказанное значение.

b) Оптимизатор: В работе рассматривался Adam (Adaptive Moment Estimation). Это комбинация метода стохастического градиентного спуска и метода адаптивной оценки моментов. Формулы обновления весов:

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$$

$$v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

$$\theta_{t+1} = \theta_t - \frac{\alpha}{\sqrt{\hat{v}_t} + \epsilon} \cdot \hat{m}_t$$

Где m_t и v_t - оценки первого и второго моментов градиента на шаге t, g_t - градиент функции потерь на шаге t, β_1 и β_2 - параметры сглаживания, ϵ - небольшое число для численной стабильности.

VIII. Экспериментальные исследования

A. Методика исследования

Используемое устройство - MacBook Pro 2023г на процессоре M2 с операционной системой Ventura 13. Вся работа производилась в Google Colaboratory.

Symbol Error Rate - определяется как отношение числа неправильно распознанных символов к общему числу переданных символов. Bit Error Rate - определяется как отношение числа неправильно распознанных битов к общему числу переданных битов.

Время замерялось с помощью time.perf_counter(), а time импортировался в программу с помощью import time.

B. Результаты экспериментов

Выбранные методы сжатия данных сравнивались с использованием эмулятора канала АБГШ с декодером приближенного передачи сообщений (AMP) [4].

Настройки декодера:

1) M: количество символов в алфавите = 1024.
2) L: количество передаваемых символов в 1 сообщении (варьируется от 1024 до 1000 в зависимости от задачи).
3) P: ограничение на среднюю мощность символа кодового слова = 15.
4) R: скорость, тестировалась на базовом значении 1.3.
5) Итерации: 30.
6) Symbol Error Rate, SER.
7) Bit Error Rate, BER.

Тестовые данные были случайным образом создавались кодером каждый раз для последующей работы. Результаты базового декодера без каких-либо модификаций представлены ниже:

AWGN channel noise variance	SER	BER	Time
1	0.112	0.02	1.2
1.2	0.135	0.128	0.6
1.5	0.173	0.1095	0.8
2	0.313	0.2123	2.4
4	0.8	0.4084	0.782

Таблица 1 - Показатели SER и BER в канале с различным распределением шума без сжатия

C. Результаты метода случайных проекций

Time	Array width size	Compress
52	10	51.9
42	20	19.9
32	50	6.1
35	100	3.4
40	200	1.12

Таблица 2 - Зависимость времени сжатия и коэффициента сжатия метода разреженной и случайной проекции

Noise variance	SER	BER
1	0.1415 – 0.1576	0.0234 – 0.0292
1.2	0.3045 - 0.425	0.2342 - 0.2814
1.5	0.501 - 0.864	0.1846 - 0.3234
2.0	0.5170 - 0.6302	0.2073 - 0.2919

Таблица 3 - Зависимость BER и SER при использовании случайной проекции на дисперсию шума канала АБГШ при различных коэффициентах сжатия

Как видно из предоставленных данных, алгоритм может сжать передаваемые значения почти в 50 раз, преобразовав вектор β в матрицу размером 10 * ($\beta.size$/10). Однако в этом случае скорость обработки сообщения значительно замедлится.

Недостаток этого метода заключается в длительной обработке сигнала и декомпрессии.

D. Результаты метода разреженной проекции

Noise variance	SER	BER
1	0.1205 – 0.1880	0.0209 – 0.0363
1.2	0.2914 – 0.4544	0.1305 – 0.2435
1.5	0.5213 – 0.734	0.2846 – 0.3734
2	0.5790 – 0.6510	0.2852 – 0.3294

Таблица 4 - Зависимость BER и SER при использовании метода разреженной проекции на дисперсию шума канала АБГШ при различных коэффициентах сжатия

Как и метод случайных проекций, этот метод показывает многообещающие результаты в сжатии матриц, но как сжатие, так и последующие процессы декомпрессии являются времязатратными операциями.

E. Результаты сжатия с использованием декомпозиции SVD

Time	Number of singular numbers	Compress
2.8 - 4.2	10	51.1
3.3 - 3.92	20	25.5
3.5 - 4.2	50	10.2
3.87 - 4.1	100	5.1
3.9 - 4.1	200	2.05

Таблица 5 - Зависимость времени сжатия и коэффициента сжатия SVD

Noise variance	SER	BER
1.0	0.13 - 0.182	0.02 - 0.42
1.2	0.158 - 0.258	0.158 - 0.84
1.5	0.393 - 0.714	0.1938 - 0.2991
2.0	0.6947 - 0.8582	0.3835 - 0.5284

Таблица 6 - Зависимость BER и SER при использовании SVD на дисперсию шума канала АБГШ при различных коэффициентах сжатия

SVD-декомпозиция продемонстрировала хорошие результаты, показав относительно низкие уровни ошибок по сравнению с базовым алгоритмом, при этом демонстрируя хорошие характеристики сжатия и скорость обработки, которая не существенно превышает скорость базового алгоритма.

F. Результаты рекурентного автокодировщика

Noise variance	SER	Compress	Time
1.3	0.2542	2	2.023
1.3	0.3136	5	2.016
1.3	0.3724	10	2.038
1.3	0.4245	20	2.043

Таблица 7 - Зависимость времени сжати, восстановления информации от коэффициента сжатия Рекурентного автокодировщика

Рекурентный автокодировщик показал хорошие результаты при обработке текста, является самым быстрым из рассматриваемых методов восстановления информации. Но качество восстановления информации, при чередовании данных различного формата, в среднем хуже, чем у SVD разложения. Также имеет зависимости от дисперсии шума. Помимо этого, невозможно изменение параметра коэффициента сжатия, для этого нужно заново обучать модель автокодировщика.

G. Результаты сверточного автокодировщика

Noise variance	SER	Compress	Time
1.3	0.1358	2	5.041
1.3	0.1479	5	5.263
1.3	0.1767	10	5.248
1.3	0.1962	20	5.284

Таблица 8 - Зависимость времени сжати, восстановления информации от коэффициента сжатия Сверточного автокодировщика

Сверточный автокодирвощик показал хорошие результаты при обработке фото и видео изображний. Но качество восстановления информации, при чередовании данных различного формата, в среднем хуже, чем у SVD разложения. Имеет зависимоть от дисперссии шума и коэффициента сжатия.

H. Вывод

Как видно из проведенных выше экспериментов, 3 из 4 методов сжатия обеспечивают сильное сжатие с минимальной потерей передаваемых данных, 2 из которых требуют значительных ограничений на задачу и. Оставшихся 2 метода плохо показали себя, как по

качеству, так и по скорость декодировния. Декомпозиция SVD проявила себя исключительно хорошо при решении поставленной задачи.

Однако стоит отметить, что у декомпозиции SVD есть свои недостатки, один из которых заключается в работе с большими матрицами. В таких случаях предпочтительнее представить передаваемый информационный вектор не как единичную матрицу и применить декомпозицию, а как набор матриц, для каждой из которых применяется декомпозиция SVD.

Если же известен тип передаваемых данных и гарантировано, что дисперсия шума будет изменяться не сильно со временем, а также можно подсчитать заранее приемлемый коэффициент сжатия, то лучше использовать подходящий автокодировщик.

IX. Заключение

В данной работе рассматривалась проблема снижения размерности передаваемых данных посредством разреженных кодов в канале АБГШ.

SVD-разложение очень хорошо показало себя при решении поставленной задачи при неизвестном типе передаваемых данных. Но, SVD-разложение имеет свои минусы, одним из которых является затратное по вычислительной мощности разложение больших матриц.

Также оба автокодировщика показали хороший результат, но они в основном эффективны при работе с подходящим себе по архитектуре типу данных, для сверточного автокодировщика это изображения, для рекурентного - текст и временные ряды.

Список литературы

1. Dobrushin R. Mathematical problems in the Shannon theory of optimal coding of information // Proc. 4th Berkeley Symp. Math., Statist., Probabil. Т. 1. — 1961. — С. 211—252.
2. Bergmans P. A simple converse for broadcast channels with additive white gaussian noise (corresp.) // IEEE Transactions on Information Theory. — 1974. — Т. 20, № 2. — С. 279—280.
3. Золотарев В. В., Овечкин Г. В. Помехоустойчивое кодирование. Методы и алгоритмы. — Научно-техническое издательство Горячая линия-Телеком, 2004.
4. Hsieh K., Venkataramanan R. Modulated sparse regression codes // 2020 IEEE International Symposium on Information Theory (ISIT). — IEEE. 2020. — С. 1432—1437.
5. Hsieh K. A Python implementation of sparse regression codes. — 2020.
6. Крутов А. Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение. // Беспроводные технологии. — 2007. — № 1.
7. Sparse regression codes / R. Venkataramanan, S. Tatikonda, A. Barron [и др.] // Foundations and Trends® in Communications and Information Theory. — 2019. — Т. 15, № 1/2. — С. 1—195.
8. Hsieh K. Spatially coupled sparse regression codes for single-and multi-user communications : дис. ... канд. / Hsieh Kuan. — 2021.
9. Hsieh K., Venkataramanan R. Modulated sparse superposition codes for the complex AWGN channel // IEEE Transactions on Information Theory. — 2021. — Т. 67, № 7. — С. 4385—4404.
10. Musa O., Hannak G., Goertz N. Generalized approximate message passing for one-bit compressed sensing with AWGN // 2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP). — IEEE. 2016. — С. 1428—1432.
11. Ma Y., Zhu J., Baron D. Compressed sensing via universal denoising and approximate message passing // arXiv preprint arXiv:1407.1944. — 2014.
12. Compressed sensing MRI / M. Lustig [и др.] // IEEE signal processing magazine. — 2008. — Т. 25, № 2. — С. 72—82.
13. Marcelloni F., Vecchio M. A simple algorithm for data compression in wireless sensor networks // IEEE communications letters. — 2008. — Т. 12, № 6. — С. 411—413.
14. Liu W., Lin W. Additive white Gaussian noise level estimation in SVD domain for images // IEEE Transactions on Image processing. — 2012. — Т. 22, № 3. — С. 872—883.
15. Souza J. C. S. de, Assis T. M. L., Pal B. C. Data compression in smart distribution systems via singular value decomposition // IEEE transactions on smart grid. — 2015. — Т. 8, № 1. — С. 275—284.
16. Fornasier M., Rauhut H. Compressive Sensing. // Handbook of mathematical methods in imaging. — 2015. — Т. 1. — С. 187—229.
17. Ji S., Xue Y., Carin L. Bayesian compressive sensing // IEEE Transactions on signal processing. — 2008. — Т. 56, № 6. — С. 2346—2356.
18. Fradkin D., Madigan D. Experiments with random projections for machine learning // Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining. — 2003. — С. 517—522.
19. Sparse autoencoder / A. Ng [и др.] // CS294A Lecture notes. — 2011. — Т. 72, № 2011. — С. 1—19.
20. Turbo autoencoder: Deep learning based channel codes for point-to-point communication channels / Y. Jiang [и др.] // Advances in neural information processing systems. — 2019. — Т. 32.
21. Kalphana I., Kesavamurthy T. Convolutional Neural Network Auto Encoder Channel Estimation Algorithm in MIMO-OFDM System. // Comput. Syst. Sci. Eng. — 2022. — Т. 41, № 1. — С. 171—185.
22. Shang W., Sohn K., Tian Y. Channel-recurrent autoencoding for image modeling // 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). — IEEE. 2018. — С. 1195—1204.

Метод балансировки загрузки очередей маршрутизатора на основе машинного обучения

Иван Гарькавый
кафедра автоматизации систем вычислительных комплексов, факультет вычислительной математики и кибернетики
Московский государственный университет им.М.В.Ломоносова
Москва, Россия
garx@asvk.cs.msu.ru

Евгений Степанов
кафедра автоматизации систем вычислительных комплексов, факультет вычислительной математики и кибернетики
Московский государственный университет им.М.В.Ломоносова
Москва, Россия
estepanov@lvk.cs.msu.ru

***Аннотация*: В сетях нового поколения проблема минимизации задержки передачи данных является одной из ключевых из-за повышения требований приложений. Одним из подходов к минимизации задержки является обеспечение сбалансированной загрузки очередей на портах сетевых маршрутизаторов. Этого можно достичь при помощи балансировки входного трафика по выходным портам маршрутизатора. Задача такой балансировки пакетов по очередям востребована в магистральных сетях, сетях SD-WAN. Чтобы не возникало переупорядочивания пакетов на получателе, вместо попакетной балансировки используется балансировка специальных сегментов потоков, называемых флоулетами. Предложен метод балансировки флоулетов по выходным портам маршрутизатора на основе обучения с подкреплением, который минимизирует вариацию загрузки очередей. Результаты сравнения с методами балансировки нагрузки ECMP, Let It Flow показывают, что предложенный метод обеспечивает загрузку очередей ближе к равномерной, чем сравниваемые с ним методы.**

Ключевые слова: задачи управления, компьютерные сети, балансировка нагрузки, управление трафиком, качество сервиса, обучение с подкреплением.

I. Введение

В сетях нового поколения, таких как Network Powered by Computing, проблема минимизации задержки передачи данных является одной из ключевых из-за повышения требований приложений [1]. Задержка в очередях сетевых устройств значительно велика и сложно предсказуема, но на неё можно влиять путём влияния на порядок заполнения очередей. Одним из подходов к минимизации задержки является обеспечение сбалансированной загрузки очередей на портах сетевых маршрутизаторов. Этого можно достичь при помощи балансировки пакетов одного и того же потока по разным выходным портам маршрутизатора. Задача такой балансировки пакетов по очередям также востребована в магистральных сетях, сетях SD-WAN.

В статье представлена математическая модель функционирования маршрутизатора. Для каждого поступающего в маршрутизатор пакета задано множество допустимых выходных портов для его передачи. Функция управления отображает количество поступивших пакетов на матрицу распределения пакетов по выходным портам. Длина очереди в каждый момент времени определяется при помощи нелинейных разностных уравнений.

В рамках представленной модели поставлена задача оптимального управления загрузкой очередей. Требуется найти функцию управления, минимизирующую вариацию длин очередей на разных выходных портах на рассматриваемом временном интервале. То есть, поступающие в маршрутизатор пакеты должны распределяться по допустимым выходным портам маршрутизатора, равномерно заполняя очереди выходных портов.

Чтобы предотвратить переупорядочивание пакетов потока на стороне получателя из-за передачи их по маршрутам с разной задержкой, в задаче рассматривается балансировка флоулетов [4, 5] – сегментов потоков, передача которых по разным маршрутам не приведет к переупорядочиванию пакетов на получателе. Флоулеты определяются как сегменты потока, расстояние между которыми больше заданного порога, который, в свою очередь, больше разницы максимальной и минимальной сквозных задержек на маршрутах, по которым происходит балансировка.

Предложен метод решения поставленной задачи при помощи обучения с подкреплением. Метод учитывает историю входной нагрузки и заполненность очередей, рассчитываемую в каждый момент времени при помощи представленной модели работы маршрутизатора. Проведено сравнение предложенного метода с методами балансировки нагрузки ECMP [2], Let It Flow [4] по критерию вариации загрузки очередей. Результаты сравнения показывают, что предложенный метод обеспечивает загрузку очередей ближе к равномерной, чем сравниваемые с ним методы.

II. Модель функционирования маршрутизатора и постановка задачи балансировки загрузки очередей

A. Модель функционирования маршрутизатора

Дан маршрутизатор с n входными и n выходными портами.

Маршрутизатор проверяет новые поступившие пакеты на входные порты с периодом t_p секунд. Будем измерять время в модели в тактах длины t_p.

$\{t\} = \{0, \dots, T\}$ – интервал дискретного времени, на котором рассматривается работа маршрутизатора. В каждый из этих моментов t маршрутизатор извлекает все пакеты, поступившие в маршрутизатор в течение интервала $(t-1, t]$, и распределяет их по выходным портам. Временем передачи пакета с входного на выходной порт пренебрегается.

На каждом выходном порту есть единственная FIFO-очередь пакетов на выдачу в канал. Длина очереди не ограничена. В начальный момент времени очереди пусты.

Каждый поступающий в маршрутизатор пакет принадлежит некоторому потоку из множества потоков $\{c\} = \{1, \dots, m\}$.

В данной модели работы маршрутизатора будем использовать частный случай попакетной балансировки нагрузки по выходным портам – балансировка флоулетов. Флоулеты – это такие сегменты потока, что интервал между поступления их на маршрутизатор не меньше заданного порога флоулетов (flowlet gap), который определяется, вообще говоря, динамически. Порог флоулетов должен быть больше разницы максимальной и минимальной текущих задержек передачи данных от маршрутизатора до узла назначения потока через допустимые выходные порты передачи для этого потока [4].

Введем вероятностное пространство $\{\Omega, \mathfrak{A}, P\}$: Ω – множество элементарных исходов, $\mathfrak{A}$ – сигма-алгебра подмножеств Ω, P – мера вероятности. Элементарный исход $\omega \in \Omega$ – расписание пакетов, поступающих в маршрутизатор на интервале $\{t\}$, а также история значений сквозных задержек на маршрутах от маршрутизатора до узлов-получателей потоков в сети на этом интервале.

$\boldsymbol{a}(t) = \boldsymbol{a}(t)(\omega) \in \mathbb{R}_{\geq 0}^{m\times 1},\ \omega \in \Omega$ – векторный случайный процесс, обозначающий суммарный размер пакетов, поступивших на все входные порты в интервале $(t-1, t]$, для каждого потока ($c = \overline{1,m}$), $t \in \{t\}$. Элемент $a_c(t)$ вектора $\boldsymbol{a}(t)$ соответственно равен суммарному размеру пакетов потока c, поступивших в интервале $(t-1, t]$. Распределение $\boldsymbol{a}(t)$ отражает характеристики потоков в исследуемой сети.

Для каждого потока дано множество допустимых выходных портов для передачи пакетов потока. Зададим все эти множества в виде одной матрицы:

$W \in \{0,1\}^{m\times n}$ – матрица допустимых выходных портов ($\overline{1,n}$) для потоков ($\overline{1,m}$). $W_{cj} = 1$ тогда и только тогда, когда на выходной порт j можно передавать пакеты потока c.

Также дан следующий векторный случайный процесс:

$\boldsymbol{h}(t) = \boldsymbol{h}(t)(\omega) \in \mathbb{R}_{\geq 0}^{m\times 1}$ – вектор порогов флоулетов потоков $c = \overline{1,m}$ в момент t.

Определим $S^t = \langle \{\boldsymbol{a}(\tau)\}_{\tau=0}^{t}, W, \{\boldsymbol{h}(\tau)\}_{\tau=0}^{t} \rangle$ – история входных данных на интервале $[0, t]$.

Определим вспомогательные обозначения:

$\boldsymbol{q}(t) \in \overline{\mathbb{R}}^{m\times 1}$ – вектор последних моментов поступления пакетов потоков на момент t. Определяется через $\boldsymbol{a}(0), \dots, \boldsymbol{a}(t)$ следующим образом:

$$q_c(t) = \begin{cases} \max\limits_{\substack{\tau \leq t,\\ a_c(\tau)>0}} \tau,\ \exists \tau \leq t\colon a_c(\tau) > 0 \\ -\infty,\ \text{иначе} \end{cases}.$$

Заметим, что $\boldsymbol{q}(t)$ можно также выразить через $\boldsymbol{a}(t)$ и $\boldsymbol{q}(t-1), t \geq 1$.

$\boldsymbol{\nu}(t) \in \{0,1\}^{m\times 1}$ – вектор, обозначающий, прошло ли к моменту t время, равное текущему порогу флоулетов, с момента поступления последнего пакета соответствующего потока (0, если да, и 1, если нет):

$$\boldsymbol{\nu}(t) = \{\nu_1(t), \dots, \nu_m(t)\},$$

где $\nu_c(t) = \begin{cases} 0,\ t - q_c(t-1) \geq h_c(t) \\ 1,\ t - q_c(t-1) < h_c(t) \end{cases}, c = \overline{1,m}.$

Если в момент t поступил пакет потока c, то при $\nu_c(t) = 0$ он будет принадлежать новому флоулету потока, а при $\nu_c(t) = 1$ – последнему из флоулетов потока на данный момент.

За $O_{m,n}$ ($J_{m,n}$) обозначим матрицу $m \times n$ из нулей (соответственно единиц). За $\odot$ обозначим операцию поэлементного умножения матриц.

Введем следующую функцию:

$U(t, \boldsymbol{a}(t), S^t) \in \{0,1\}^{m\times n}$ – функция управления – отображение количества $\boldsymbol{a}(t)$ поступивших пакетов в момент t на матрицу распределения потоков по выходным портам, с учетом входных данных S^t от начального момента 0 до текущего момента t. Если $a_c(t) > 0$, то элемент $U_{cj}(t, \boldsymbol{a}(t), S^t)$ обозначает, передавать ли пакеты потока c на выходной порт j (1, если передавать, и 0, если нет).

Ограничения на функцию управления:

1) В рамках одного такта, для каждого потока должен быть определен единственный выходной порт для передачи:

$$\boldsymbol{g}_1(U, t, S^t) = U(t, \boldsymbol{a}(t), S^t) \cdot J_{n,1} - J_{m,1} = O_{m,1} \quad (1)$$

2) Пакеты должны быть распределены по допустимым выходным портам:

$$G_2(U, t, S^t) = U(t, \boldsymbol{a}(t), S^t) \odot \left(J_{m,n} - W\right) = O_{m,n} \quad (2)$$

3) Все пакеты потока в пределах текущего флоулета передаются на один и тот же выходной порт:

$$G_3(U, t, S^t) = U(\tau, \boldsymbol{a}(\tau), S^\tau)|_{\tau=t-1}^{t} \odot \left(\boldsymbol{\nu}(t) \cdot J_{1,n}\right) = O_{m,n},\ t \geq 1 \quad (3)$$

Определим $\{U\}$ – множество функций управления, удовлетворяющих ограничениям (1)—(3) при любых входных данных:

$$\{U\} = \left\{ U(t, \boldsymbol{a}, S) \middle| \begin{array}{c} \forall t \in \{t\},\ \forall \tilde{\boldsymbol{a}}^0, \dots, \tilde{\boldsymbol{a}}^t, \tilde{\boldsymbol{h}}^0, \dots, \tilde{\boldsymbol{h}}^t \in \mathbb{R}_{+}^{m\times 1}, \\ \forall W \in \{0,1\}^{m\times n}, \tilde{S}^t = \langle \{\tilde{\boldsymbol{a}}^\tau\}_{\tau=0}^{t}, W, \{\tilde{\boldsymbol{h}}^\tau\}_{\tau=0}^{t} \rangle \\ \boldsymbol{g}_1\left(U, t, \tilde{S}^t\right) = O_{m,1} \\ G_2\left(U, t, \tilde{S}^t\right) = O_{m,n} \\ G_3\left(U, t, \tilde{S}^t\right) = O_{m,n},\ t \geq 1 \end{array} \right\}.$$

Также дано: $\boldsymbol{r} \in \mathbb{R}_{\geq 0}^{n\times 1}$ – вектор пропускных способностей (размер данных, передаваемый за один такт) выходных каналов, смежных с выходными портами $\overline{1,n}$.

$\boldsymbol{b}(t) \in \mathbb{R}_{\geq 0}^{n\times 1}$ – случайный векторный процесс, обозначающий длины очередей (суммарный размер пакетов) на выходных портах $\overline{1,n}$ к моменту t. Определяется соотношениями:

$$\boldsymbol{b}(t+1) = [\boldsymbol{b}(t) + U^T(t, \boldsymbol{a}(t), S^t) \cdot \boldsymbol{a}(t) - \boldsymbol{r}]^+,$$

$t \in \{0, \dots, T-1\}$;

$$\boldsymbol{b}(0) = O_{n,1},$$

где $[x]^+ = \max(0, x)$.

Введем вспомогательные обозначения:

$\mu(t) = \frac{1}{n}\sum_{i=1}^{n} b_i(t)$ – средняя по выходным портам длина очереди.

$\delta(t) = \frac{1}{n}\sum_{i=1}^{n}\left(b_i(t) - \mu(t)\right)^2$ – средний (по портам) квадрат отклонения длины очереди от средней (по портам) длины очереди в момент t.

B. Постановка задачи балансировки загрузки очередей

Дано:

- Модель входных потоков (случайный процесс $\boldsymbol{a}(t)$, матрица допустимых выходных портов W, случайный процесс $\boldsymbol{h}(t)$);
- Модель работы маршрутизатора.

Найти: функцию управления $U(t, \boldsymbol{a}(t), S^t)$:

- удовлетворяющую ограничениям (1)—(3);
- минимизирующую средние отклонения длин очередей:

$$U^* = \underset{U\in\{U\}}{\operatorname{argmin}} F(U),\ F(U) = \mathbb{E}\left(\frac{1}{T+1}\sum_{t=0}^{T}\delta(t)\right).$$

III. Обзор существующих методов балансировки нагрузки

Существует ряд методов, которые балансируют входную нагрузку в маршрутизатор по маршрутам для её передачи. Для равномерной загрузки очередей выходных портов требуется высокая адаптируемость метода балансировки нагрузки к изменяющемуся профилю входного трафика.

Выбор единицы балансировки также влияет на адаптируемость. Методы балансировки потоков не обладают достаточно высокой адаптируемостью, т.к. для потоков нельзя динамически изменять выбор выходного порта для передачи, в том числе для значительно длительных и интенсивных потоков. Для решения поставленной задачи подходят только методы балансировки флоулетов.

Для использования в глобальных вычислительных сетях (WAN) также необходимо, чтобы метод не требовал модифицировать заголовки пакетов.

Для применимости к поставленной задаче метод должен иметь аналогичный критерий оптимизации, напрямую связанный с вариацией загруженности очередей.

A. Методы, основанные на эвристиках

В сетях традиционно используется метод ECMP (Equal-cost multi-path) [2], который равномерно распределяет потоки по маршрутам одинаковой стоимости. Также используется UCMP (Unequal-cost multi-path) [3], в котором балансировка осуществляется по маршрутам неравной стоимости. В версии Cisco метода UCMP балансировка осуществляется в соответствии с весами, зависящие от входной нагрузки [3]. Существует метод Let It Flow [4], который равномерно распределяет флоулеты по маршрутам. Все эти методы не имеют критерия оптимизации, поэтому они не подходят для решения поставленной задачи. Метод CONGA [5] балансирует флоулеты с целью минимизации перегрузок в сети, но он требует модифицировать заголовки пакетов для учета статистики перегрузок. Из-за этого он может быть применим в сетях центров обработки данных, но не подходит для балансировки нагрузки в глобальных вычислительных сетях.

B. Обучение с подкреплением

Для достижения максимальной адаптируемости к изменяющемуся профилю трафика целесообразно рассмотреть методы на основе машинного обучения. Для решения задач управления в этом случае используются методы обучения с подкреплением.

В ходе обучения с подкреплением агент обучается, взаимодействуя со средой. В каждый момент дискретного времени агент наблюдает *состояние* среды, выбирает и совершает одно из доступных *действий*. По выбранному агентом действию среда переходит в новое состояние и возвращает агенту вещественное число – *награду*. Распределение вероятностей переходов в новые состояние и награда полностью определяются текущим состоянием и выбранным в текущий момент действием. Агент обучается выбирать действия в зависимости от состояния среды, максимизируя получаемую награду. Среда обычно описывается в форме марковского процесса принятия решений (МППР), в том числе – частично наблюдаемого МППР, при котором агент наблюдает лишь часть состояния среды, называемую *наблюдением*.

Существует ряд методов балансировки флоулетов при помощи обучения с подкреплением [6, 7, 8], однако в них не рассматривается целевая функция, напрямую связанная с загруженностью очередей маршрутизаторов, поэтому они не подходят для решения поставленной задачи.

C. Выводы

На основе обзора решено разработать метод решения поставленной задачи балансировки флоулетов на основе обучения с подкреплением, т.к. в этом случае можно достичь максимальной адаптируемости к входной нагрузке.

IV. Предлагаемый метод

Предлагается применить к поставленной задаче один из методов обучения с подкреплением, допускающих непрерывные пространства наблюдений и действий. Для этого нужно определить пространство наблюдений, пространство действий и награду, на основе которых алгоритм обучения с подкреплением будет обучаться балансировать флоулеты по выходным портам.

Уточним, что при конфигурации алгоритма обучения с подкреплением заранее известно и задано статически количество маршрутов до каждого узла от рассматриваемого маршрутизатора. Узлы сети обозначим $1, \dots, V$, маршруты от рассматриваемого маршрутизатора до узла v обозначим $e_{v,i}$, $i = \overline{1, E_v}$. Для любого потока c с узлом назначения v маршруты $e_{v,i}$, $i = \overline{1, E_v}$ взаимно-однозначно соответствуют допустимым портам $\{j | W_{cj} = 1\}$ для передачи этого потока.

В каждый момент времени t алгоритм балансировки нагрузки хранит набор значений весов распределения входного трафика по маршрутам: $w_{v,i}(t) \geq 0$, $v = \overline{1, V}$,

$i = \overline{1, E_v}$, таких что $\forall v \sum_{i=1}^{E_v} w_{v,i}(t) = 1$ (условие нормировки). При поступлении новых флоулетов любого потока с узлом назначения v, порт для него выбирается из множества допустимых портов потока с вероятностями $w_{v,i}(t), i = \overline{1, E_v}$.

Действие в момент t в предлагаемом методе балансировки – вектор из новых значений весов $w^*_{v,i}$, $v = \overline{1, V}$, $i = \overline{1, E_v}$. В результате действия в следующий момент времени весам распределения $w_{v,i}(t+1)$ присваиваются значения $w^*_{v,i}$, прошедшие нормировку. Действие выполняется каждые p_a тактов дискретного времени, p_a – параметр алгоритма. Определим число маршрутов $n_R = \left|\{(v,i) | v = \overline{1, V},\ i = \overline{1, E_v}\}\right|$.

Наблюдение в момент t составляется следующим образом:

1) Для каждого маршрута i до каждого узла v ($v = \overline{1, V}$, $i = \overline{1, E_v}$) вычисляется объем данных, переданный по маршруту за последний такт времени, разделённый на пропускную способность соответствующего выходного канала.

2) Для каждого выходного порта вычисляется текущая длина его очереди, разделённая на пропускную способность соответствующего канала. Из получившегося вектора вычитается значение его минимального элемента.

3) Получившиеся два вектора конкатенируются, получается вектор длины $n_R + n$.

4) Определим Q, q – параметры алгоритма (Q делится нацело на q): q определяет временную размерность наблюдения, Q определяет, сколько последних тактов времени учитывается в наблюдении. Набирается история значений вектора, получаемого на шагах 1—3, за последние Q тактов времени $t - Q + 1, \dots, t$. Значения на каждых Q/q идущих подряд тактах усредняются, получается q векторов длины $n_R + n$.

5) Конкатенация полученных q векторов является наблюдением.

Таким образом, размерность наблюдения равна $(n_R + n)q$. Если число E_v маршрутов до каждого узла v не превышает K_R , то размерность пространства наблюдений не превышает $(K_R|V| + n)q$, а размерность пространства действий не превышает $K_R|V|$.

Награда в момент t определяется как разница вариации загрузок очередей (аналогично формуле в целевой функции задачи) до и после действия: $\delta(t - p_a) - \delta(t)$.

V. Экспериментальное исследование

A. Методика экспериментального исследования

Из алгоритмов обучения с подкреплением выбраны PPO (Proximal Policy Optimization) [9], SAC (Soft Actor-Critic) [10], TD3 (Twin Delayed Deep Deterministic Policy Gradient) [11], A2C (Advantage Actor-Critic) [12] (последний – в синхронной реализации). Результаты сравнения будут представлены лишь для одного из этих алгоритмов, который покажет наилучший результат.

Проведено сравнение предложенного метода с методами ECMP [2] (равномерная балансировка потоков по маршрутам) и Let It Flow [4] (равномерная балансировка флоулетов по маршрутам) по критерию вариации загрузки очередей (т.е. целевой функции задачи).

Сравнение методов проведено на имитационной модели маршрутизатора в соответствии с описанной математической моделью. Входной трафик в маршрутизатор генерируется искусственно, моделируется попакетно.

Поступление пакетов каждого потока в маршрутизатор – процесс Пуассона. Для каждого узла сети инициация новых потоков с получателем в этом узле моделируется процессом Пуассона, а продолжительность этих потоков имеет геометрическое распределение. Значения сквозных задержек на маршрутах от маршрутизатора до узлов сети моделируются как константы, взятые из одного и того же равномерного распределения. Порог флоулета для потока вычисляется как разница максимальной и минимальной сквозных задержек на маршрутах для передачи потока, умноженная на 2. Значения остальных параметров генерации входного трафика берутся из равномерных распределений.

В экспериментах используется маршрутизатор с $n = 8$ выходными портами и одинаковыми пропускными способностями каналов. В маршрутизатор поступает трафик до $|V| = 20$ узлов-получателей, и до каждого узла-получателя задано от 1 до $K_R = 3$ маршрутов. Входной трафик генерируется так, что матожидание его суммарной нагрузки составляет 60% от пропускной способности маршрутизатора. Значения матожидания длины потока для разных узлов-получателей потока – от 2 до 20 с.

Параметры предложенного метода балансировки следующие: действие принимается каждые 2,4 с, в наблюдении учитывается история за последние 7,2 с, временная размерность наблюдения $q = 3$.

Проведено 12 экспериментов на 3 разных профилях входного трафика (наборах значений параметров вероятностных распределений для его генерации). В рамках каждого эксперимента модель каждого выбранного алгоритма обучения с подкреплением обучается на 30 эпизодах (соответствуют трассам входного трафика) длительностью 4 минуты модельного времени (за которые принимается 100 действий), и работа всех сравниваемых методов проверяется на 10 эпизодах той же длительности. Проверка заключается в измерении значения целевой функции на эпизоде. После каждого эпизода обучения модели замеряется средняя суммарная награда за последние несколько эпизодов, и наилучшая по этому критерию версия модели будет использована для проверки метода. За каждый эпизод маршрутизатор обрабатывает в среднем по 1400 потоков и 6000 флоулетов.

Данная методика также применима для других классов распределений и их параметров для генерации трафика (направление дальнейшего исследования – их уточнение на основе анализа реальных трасс сетевого трафика).

B. Результаты

Из выбранных алгоритмов обучения с подкреплением наилучший результат на входных данных дал алгоритм

SAC (Soft Actor-Critic), его результат изображен на Рис. 1. Для каждого алгоритма достигнутые им значения целевой функции на 120 проверочных эпизодах изображены в виде графика «ящик с усами», который отображает медиану значений по эпизодам (её значение подписано на графике), нижний и верхний квартили и др. Значение целевой функции умножено на константу, чтобы результат был равен вариации задержек в очередях (средний квадрат отклонения от среднего по портам).

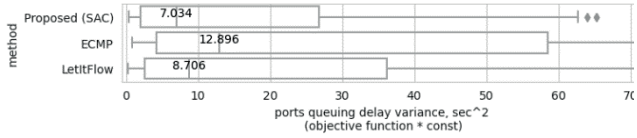

Рис. 1. Сравнение методов балансировки нагрузки.

На описанных входных данных предложенный метод балансировки флоулетов на основе алгоритма обучения с подкреплением SAC (Soft Actor-Critic) достигает значения целевой функции по медиане на 20% меньше, чем метод балансировки флоулетов LetItFlow, и на 45% меньше, чем метод балансировки потоков ECMP, а также уменьшает разброс значений целевой функции по сравнению с обоими методами.

VI. Заключение

Предложен метод балансировки флоулетов с целью равномерной загрузки очередей маршрутизатора, использующий обучение с подкреплением для учета истории входных потоков. Сравнение показало, что разработанный метод уменьшает значение целевой функции на 20% по сравнению с методами ECMP и Let It Flow.

Направления дальнейшего исследования:

- на основе сетевых трасс уточнить методику экспериментального исследования;
- разработать метод решения аналогичной задачи для сети из множества маршрутизаторов, используя многоагентные методы обучения с подкреплением.

Список литературы

[1] Smeliansky R. L. Network powered by computing // 2022 International Scientific and Technical Conference on Modern Computer Network Technologies (MoNeTeC). Moscow: IEEE, 2022. P. 1–5.

[2] ECMP Load Balancing [Электронный ресурс] // URL: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/mp_l3_vpns/configuration/xe-3s/asr903/mp-l3-vpns-xe-3s-asr903-book/mp-l3-vpns-xe-3s-asr903-book_chapter_0100.pdf (дата обращения: 19.06.2024).

[3] UCMP Load Balancing [Электронный ресурс] // URL: https://www.cisco.com/c/en/us/td/docs/ios-xml/ios/mp_l3_vpns/configuration/xe-3s/asr903/17-1-1/b-mpls-l3-vpns-xe-17-1-asr900/m-ucmp.pdf (дата обращения: 19.06.2024).

[4] Let it Flow: Resilient Asymmetric Load Balancing with Flowlet Switching / Vanini E., Pan R., Alizadeh M., Taheri P., Edsall T. // Proceedings of the 14th USENIX Symposium on Networked Systems Design and Implementation. Boston, MA, USA: NSDI, 2017. P. 407–420.

[5] CONGA: distributed congestion-aware load balancing for datacenters / M. Alizadeh, T. Edsall, S. Dharmapurikar, R. Vaidyanathan, K. Chu, A. Fingerhut, V. T. Lam, F. Matus, R. Pan, N. Yadav, G. Varghese // Proceedings of the 2014 ACM conference on SIGCOMM. Chicago, Illinois, USA: ACM, 2014. P. 503–514.

[6] Liu P. Using random neural network for load balancing in data centers //Proceedings on the International Conference on Internet Computing (ICOMP). – The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp), 2015. – P. 3.

[7] Lin Q. et al. Rilnet: A reinforcement learning based load balancing approach for datacenter networks //Machine Learning for Networking: First International Conference, MLN 2018, Paris, France, November 27–29, 2018, Revised Selected Papers 1. – Springer International Publishing, 2019. – P. 44–55.

[8] Lim J., Yoo J. H., Hong J. W. K. Reinforcement learning based load balancing for data center networks //2021 IEEE 7th International Conference on Network Softwarization (NetSoft). – IEEE, 2021. – P. 151—155.

[9] Schulman J. et al. Proximal policy optimization algorithms // arXiv preprint arXiv:1707.06347. – 2017.

[10] Haarnoja T. et al. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor // International conference on machine learning. – PMLR, 2018. – P. 1861–1870.

[11] Fujimoto S., Hoof H., Meger D. Addressing function approximation error in actor-critic methods // International conference on machine learning. – PMLR, 2018. – P. 1587–1596.

[12] Mnih V. et al. Asynchronous methods for deep reinforcement learning // International conference on machine learning. – PMLR, 2016. – P. 1928–1937.

Калибровка метода оценки плотности транспортного потока с применением широкоугольной камеры

Кутейников И.А.
Кафедра высшей математики
Московский автомобильно-дорожный государственный технический университет (МАДИ);
Кафедра математической кибернетики и информационных технологий
Московский технический университет связи и информатики (МТУСИ)
Москва, Россия
ivankuteynikov09@gmail.com

Яшина М.В.
Кафедра высшей математики Московского автомобильно-дорожного государственного технического университета (МАДИ) и ФГУП «НАМИ» Государственного научного центра Российской Федерации
Москва, Россия
mv.yashina@madi.ru

***Abstract*—В последние годы развитие вычислительных технологий и доступность высокоточных компонентов способствуют созданию систем для мониторинга транспортных потоков. Одной из таких систем является разработанное программное обеспечение – Virtual Detectors System (ViDES), предназначенное для оценки характеристик транспортного потока с использованием видеокамер. В статье рассматривается калибровка метода оценки плотности транспортного потока, основанная на данных, полученных с широкоугольных камер, с учетом гидродинамической модели. Представлены особенности алгоритмов обработки видеоизображений, позволяющие детектировать и анализировать транспортные потоки в режиме реального времени. Уделено внимание калибровке системы для повышения точности измерения ключевых параметров транспортного потока: плотности, интенсивности и скорости. Также рассматриваются методологические подходы к учету и минимизации ошибок, вызванных различиями в углах обзора и разрешении камер. Результаты исследования показывают, что разработанная методика позволяет улучшить точность оценки транспортной плотности и может быть полезной для оптимизации управления дорожным движением, предотвращения пробок и повышения безопасности на дорогах.**

Keywords*— *метод виртуальных детекторов, гидродинамическая модель, плотность транспортного потока, алгоритмы компьютерного зрения, анализ данных

I. Введение

Традиционные методы мониторинга транспортных потоков, такие как использование индуктивных петель, микроволновых датчиков и дорожных сенсоров, остаются основой для оценки дорожного движения в различных странах. Однако эти методы имеют значительные ограничения, главным образом связанные с высокой стоимостью установки и обслуживания, необходимостью развертывания сложной инфраструктуры для передачи и обработки данных. В условиях быстрого роста городов и увеличения автомобилизации современного общества все более актуальной становится необходимость поиска более эффективных и гибких решений для мониторинга транспортных потоков [1].

Одним из таких решений является использование метода виртуальных детекторов, основанного на системах видеонаблюдения [2]. Данный метод использует существующие камеры наблюдения и технологии компьютерного зрения для анализа видеоизображений, что позволяет извлекать информацию о характеристиках транспортных потоков без применения физических датчиков. Важным преимуществом виртуальных детекторов является их экономичность, масштабируемость и гибкость, что делает их особенно полезными в условиях динамично меняющейся дорожной среды. Камеры могут быть легко перепрофилированы для мониторинга дорожного движения с минимальными затратами, что снижает необходимость установки нового оборудования.

Особое внимание в мониторинге транспортных потоков уделяется плотности, так как она является ключевым показателем для оценки состояния дорожной сети. Плотность напрямую влияет на уровень загруженности дорог, риск возникновения пробок и, в конечном итоге, на безопасность и комфорт водителей. Для более точного и надежного измерения плотности транспортного потока в современных системах мониторинга вводится гидродинамическая модель движения. Данная модель позволяет описывать взаимодействие транспортных средств в потоке, учитывая изменение плотности в зависимости от условий движения, таких как скорость, интенсивность и взаимодействие автомобилей. Гидродинамическая модель помогает лучше понять поведение транспортного потока и улучшить управление дорожной ситуацией.

Широкоугольные камеры играют ключевую роль в применении методов виртуальных детекторов, поскольку они позволяют захватывать более широкие участки

дорожного полотна. Это обеспечивает более точное и полное представление о дорожной обстановке, что особенно важно при анализе плотности транспортного потока. Широкий угол обзора позволяет минимизировать слепые зоны и улучшить точность оценки количества транспортных средств на дороге, что повышает эффективность мониторинга.

Разработанная система на базе метода виртуальных детекторов Virtual Detectors System (ViDES) — использует преимущества широкоугольных камер и гидродинамической модели для точного мониторинга транспортных потоков [3]. ViDES позволяет не только оценивать такие параметры, как плотность, интенсивность и средняя скорость движения, но и адаптироваться к изменениям дорожной обстановки в режиме реального времени. Это делает систему полезной для предотвращения заторов, оптимизации управления дорожным движением и повышения безопасности на дорогах.

II. Гидродинамическая модель транспортных потоков

Одним из макроскопических подходов к моделированию трафика с середины пятидесятых годов прошлого столетия становится гидродинамическая модель и ее различные модификации, связанные с уподоблением потока автомобилей текущей жидкости.

В работе 1955 года Лайтхилла-Уизема [4] зависимость функции от плотности была уподоблена течению жидкости с плотностью $\rho(x,t)$, равной числу машин на единицу длины дороги, и интенсивностью $q(x,t)$, равной количеству автомобилей, пересекающих сечение x за единицу времени. Учитывая закон сохранения массы, имеем сохранение числа машин на участке дороги в замкнутой системе. В случае открытой системы имеем $q_x + \rho_t = 0$ и $q_x + \rho_t = g(x,t)$. Функция $g(x,t)$ отражает интенсивность потока въезжающих/выезжающих транспортных средств. Такая модель соответствовала реальным наблюдениям, особенно при низких интенсивностях потоков.

Пусть $R(t,x)$ — количество автомобилей, которые в момент времени t приближаются к сечению $x \in G$ по одной полосе.

Тогда интенсивность транспортного потока — q определяется как:

$$q(t,x) = \frac{\partial R}{\partial t}.$$

Плотность транспортного потока — ρ определяется как:

$$\rho(t,x) = -\frac{\partial R}{\partial x}.$$

Условие регулярности (уравнение непрерывности) имеет вид:

$$\frac{\partial \rho}{\partial t} = \frac{\partial q}{\partial x} \Leftrightarrow \frac{\partial^2 R}{\partial t \partial x} = \frac{\partial^2 R}{\partial x \partial t}.$$

Скорость транспортного потока — v в точке $x \in G$ в момент времени t:

$$v(t,x) = \frac{q(t,x)}{\rho(t,x)}.$$

Для подсчета плотности транспортного потока на протяжении 1 км при средней длине транспортного средства 5 м потребуется $k = 200$ дискретных областей для детектирования.

Обозначим как N — количество областей, в которых находятся транспортные средства, а L — как общее количество областей на полосе, получим

$$\rho = \frac{N}{L},\ 0 \le \rho \le 1.$$

Для перехода к единицам измерения авт./км

$$\rho = \frac{N}{L} * k,\ 0 < \rho < 200.$$

Обозначив область детектирования как d (рис. 1), получим соотношение $d = \frac{L}{N}$ откуда следует $d \sim \frac{1}{\rho}$.

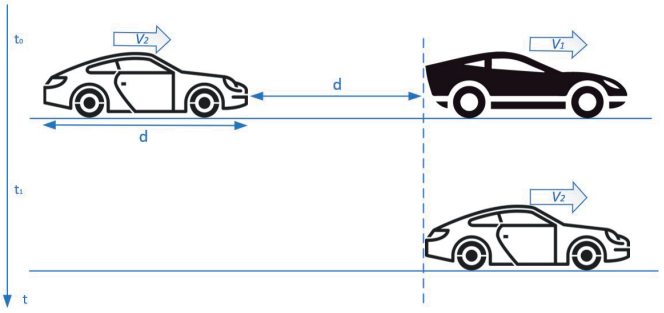

Рис. 1. Разбиение полосы на дискретные фрагменты.

В рамках исследования гидродинамической модели коллективом авторов под руководством профессора А.П. Буслаева была разработана гидродинамическая транспортная модель города Москвы, реализованная как вычислительная схема Годунова для графа [5].

III. Описание входных данных

Использование широкоугольных камер в системах мониторинга транспортных потоков приобретает все большую актуальность благодаря их способности охватывать более широкие участки дорожного полотна. В отличие от камер с узким углом обзора, широкоугольные камеры могут фиксировать сразу несколько дорожных полос, перекрестков и прилегающих зон, что значительно увеличивает объем данных для анализа. Это особенно важно при оценке плотности транспортных потоков, так

как позволяет более точно учитывать количество транспортных средств и их распределение на дороге. В условиях плотного движения, где высока вероятность заторов и пробок, широкий угол обзора минимизирует слепые зоны и снижает вероятность пропуска ключевых событий.

Рассмотрим поток кадров в виде матриц размера $n \times m$, где n — количество пикселей (pixel|p) по горизонтали, m — количество пикселей по вертикали, полученных с использованием цифровой видеокамеры iVideonCute 2 (рис.2).

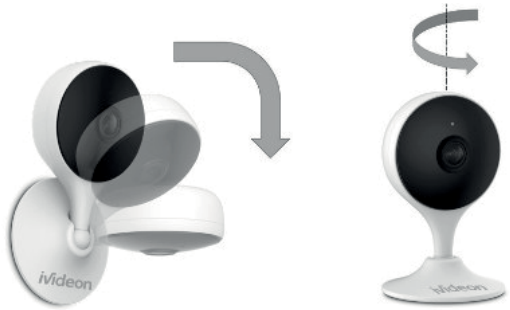

Рис. 2. Способы размещения камеры iVideonCute 2

Камера установлена в лаборатории кафедры «Высшая математика» Московского Автомобильно-Дорожного Института (МАДИ) и транслирует движение автотранспорта по Ленинградскому шоссе (рис.3) в разрешении 1920x1080 - Full HD в профиль .

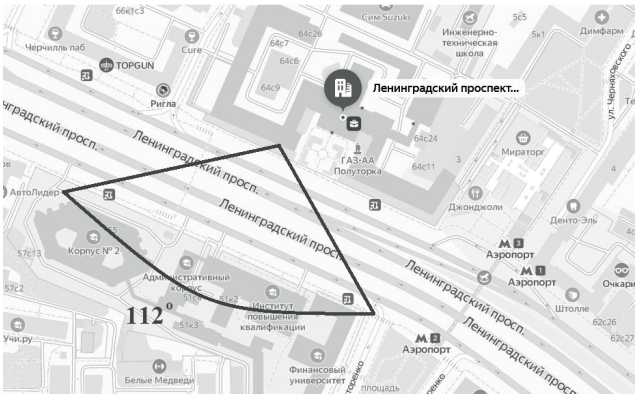

Рис. 3. Схема размещения камеры на карте

Она размещена на высоте ~ 18 м под углом ~ 60°. Угол обзора объектива камеры:

- горизонтальный: 112°;
- вертикальный: 58°;
- диагональный: 131°.

Данный угол обзора позволяет осуществлять захват ~ 50 м магистрали.

Измерение длины магистрали может осуществляться с использованием спутниковых систем глобального позиционирования (ГЛОНАСС, GPS), априорном знании о размере объектов на изображении (разметка, автомобили), лазерных измерителях и других технических средствах [6].

Длина одного штриха разметки на исследуемом участке магистрали со скоростным ограничением 60 км/ч равна 2 метрам (рис.4).

Рис. 4. Захват видеоданных с камеры

IV. Метод виртуальных детекторов

Метод виртуальных детекторов позволяет собирать данные практически с любого ракурса съемки благодаря гибкости в настройке размера, форме и расположению детекторов.

Детектор (detector|det) – прямоугольная область на экране размера $n_{\text{det}} \times m_{\text{det}}$, где $n_{\text{det}} \le m, m_{\text{det}} \le m$, для регистрации и накопления цветовых изменений.

Ширина и высота детектора выбирается таким образом, чтобы АТС, движущиеся по полосе движения, обязательно попадало корпусом в область детектора (рис. 5). Если учесть, что заполнение контрольной области детектора корпусом АТС должно быть минимум 30%, средняя ширина дорожной полосы движения ТС составляет 3,75 м., а средняя ширина самого ТС 1,8 м. при средней длине 5 м., то рекомендуемая высота контрольной области детектора вычисляется как (1,8*30%)*2+0,15 = 1,23 м, а ширина (5*30%)*2+0,15 = 3,15 м.

Рис. 5. Расстановка детекторов на полосе

После расстановки для области каждого детектора на каждом кадре видеопотока происходит вычисление

выбранной метрики, связанной с интенсивностью цвета, в момент времени t. После чего для каждого детектора происходит бинаризация значения метрики по заданному порогу ε, где 1 – соответствует наличию транспортного средства в области детектора, а 0 – его отсутствию.

В дальнейшем бинаризованные значения усредняются и переводятся в секунды в соответствии с частотой кадров видеосъемки. Дальше происходит вычисление характеристик транспортных потоков. Алгоритм работы метода виртуальных детекторов представлен на рис. 6.

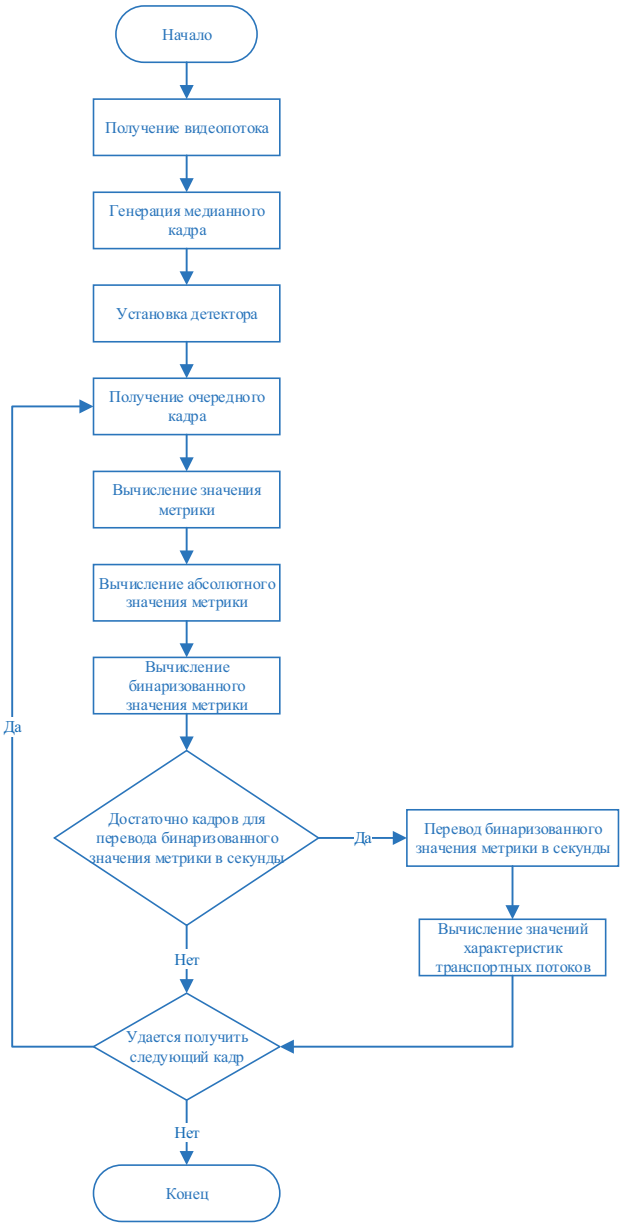

Рис. 6. Блок-схема алгоритма работы метода виртуальных детекторов

V. Алгоритм оценки плотности транспортного потока

В процессе измерений фиксируется количество автомобилей на каждой полосе участка дороги в каждый момент времени. Для этого производится суммирование значений активированных детекторов в каждой строке таблицы в .csv файле, после чего полученная сумма делится на общее количество детекторов на полосе (рис.7).

Таким образом, плотность транспортного потока определяется на участке дороги в каждый момент времени t. При измерении плотности, зная размер каждого детектора, единицы измерения переводятся в километры (авт./км), что позволяет получить более наглядные для визуализации и анализа результаты.

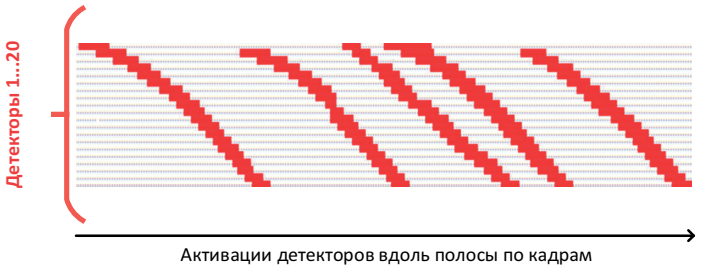

Рис. 7. Визуализакция активации детекторов вдоль полосы

VI. Полученные результаты

В результате обработки данных за 25 июня 2024 года, полученных с виртуальных детекторов, рассчитанные плотности транспортного потока за 24 часа представлены на рис.8-9.

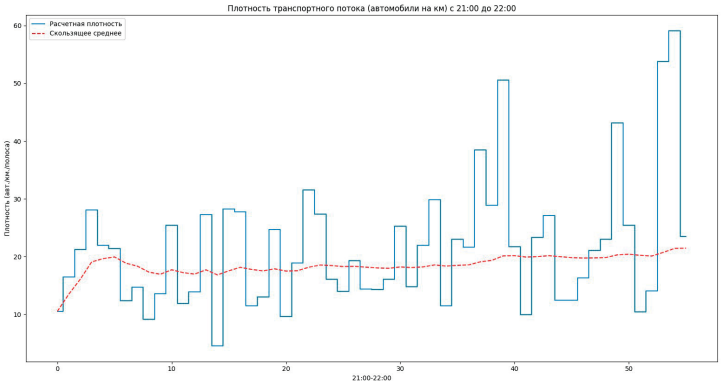

Рис. 8. Плотность транспортного потока с 21:00 по 22:00

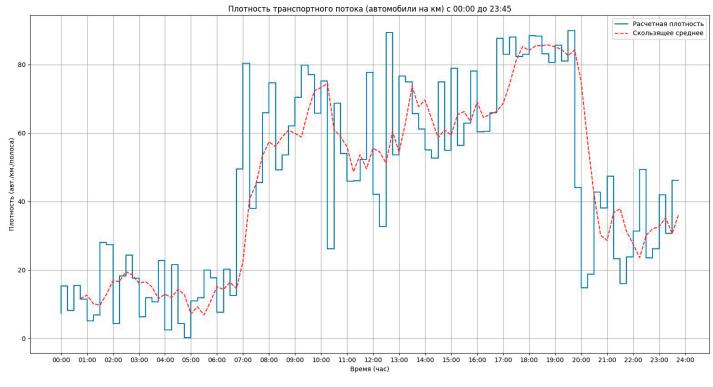

Рис. 9. Плотность транспортного потока с 00:00 по 23:45

Из графиков видно, что рассчитанная плотность отражает колебания дорожной обстановки в течение дня с ростом утреннего, обеденного и вечернего пика. Разброс в соседних значениях, особенно в промежуток с вечера до утра может быть вызван следами от фар, что затрудняет определение особенностей транспортных средств. Это явление является распространенной проблемой в компьютерном зрении. Применение адаптивных метрик для различных погодных и временных сценариев позволит сгладить выбивающиеся из общей статистики результаты. В целом, результат оценки в большинстве случаев близок к истинному положению дел, что свидетельствует о том, что предложенный подход может обеспечить точное обнаружение.

VII. Выводы

В рамках статьи представлен метод подсчета плотности транспортного потока с использованием широкоугольной камеры на основе метода виртуальных детекторов. Широкоугольная камера, позволяет охватить большую площадь дорожного полотна, что повышает точность и полноту анализа транспортных потоков. В сочетании с алгоритмами компьютерного зрения и гидродинамической моделью, предложенный подход обеспечивает более точное измерение плотности транспортного потока в режиме реального времени, что является ключевым фактором для управления дорожным движением и предотвращения пробок.

Применение метода виртуальных детекторов позволяет избежать необходимости в установке дорогостоящего физического оборудования и использовать существующую инфраструктуру камер видеонаблюдения. Это не только снижает затраты на реализацию системы, но и обеспечивает гибкость в ее адаптации к различным дорожным условиям. Представленная методика показала свою эффективность в различных сценариях движения, что подтверждает ее практическую значимость для использования в городских условиях с высокой интенсивностью движения.

Таким образом, предложенный метод может стать важным инструментом для повышения точности мониторинга транспортных потоков и улучшения управления дорожной ситуацией. В будущем данная методика может быть дополнительно улучшена с учетом возможных изменений дорожных условий, погодных факторов и других переменных, что позволит еще более эффективно контролировать плотность транспортного потока и минимизировать заторы.

Список литературы

[1] A.P. Buslaev, M.V. Yashina, I.S. Kotovich. On problems of intelligent monitoring for traffic. Logic Journal of the IGPL. 2011, no. 19(2), pp. 384-394.

[2] M. V. Yashina, A. S. Bugaev, I. A. Kuteynikov, M. A. Kuznetsov. Evaluation of Recovery Accuracy of Vehicles Flow Characteristics by Video Sensor Views. 2022 Systems of Signals Generating and Processing in the Field of on Board Communications. – IEEE, 2022. – P. 1-4. - DOI 10.1109/IEEECONF53456.2022.9744271

[3] Kuteynikov I. A. Development of a System «ViDeS» for Monitoring the Dynamics of Traffic Flows Based on the Virtual Detectors Method. 2022 Intelligent Technologies and Electronic Devices in Vehicle and Road Transport Complex (TIRVED), 2022, P. 1-7, DOI 10.1109/TIRVED56496.2022.9965453

[4] Lighthill M. J., Whitham G. B. On kinematic waves II. A theory of traffic flow on long crowded roads //Proceedings of the royal society of london. series a. mathematical and physical sciences. – 1955. – Т. 229. – №. 1178. – С. 317-345.

[5] Луканин, В. Н., Буслаев, А. П., Трофименко, Ю. В., Яшина, М. В. (1998). Автотранспортные потоки и окружающая среда.

[6] Яшина М.В., Розентблат Г.М., Солиев Ю.С., Кутейников И.А., Доткулова А.С. Математические методы для управления дорожной инфраструктурой : коллективная монография в 2-х частях. Часть I. – Москва : Московский автомобильно-дорожный государственный технический университет (МАДИ), 2021. – 164 с. – ISBN 978-5-7962-0280-7.

Эффективные методы сжатия данных при формировании луча в сетях LTE и 5G NR

Д.Р. Лысенко
ВМК МГУ им. М.В. Ломоносова
Москва, Россия
lysenkodr@my.msu.ru

В.О. Писковский
ВМК МГУ им. М.В. Ломоносова
Москва, Россия
vpiskovski@lvk.cs.msu.ru

Аннотация—Системы связи пятого поколения 5G NR используют антенные решётки для направленной передачи и приёма, что в свою очередь способствует повышению производительности и эффективности связи. Для дуплексных систем с частотным разделением каналов важна обратная связь о состоянии канала. Выбор кодовой страницы для конфигурации антенной решетки базовой станции и антенных портов клиентского устройства происходит при помощи обмена отчетами. Суть работы состоит в оптимизации процедуры обмена, применении и исследовании методов сжатия данных.

Index Terms—5G NR, MIMO, CSI, беспроводная связь, Beamforming, сжатие данных, базовая станция, антенная решётка, пользовательское оборудование, дискретное преобразование Фурье, дискретное вейвлет-преобразование, сингулярное разложение.

I. Введение

В системах MIMO (Multiple Input Multiple Output) информация о состоянии канала (Channel State Information, CSI) на базовой станции необходима для обеспечения качественной работы прекодирования, настройки антенной решётки при формировании луча. Однако, по мере роста числа антенн, установленных на базовой станции, обеспечение обратной связи становится сложной задачей из-за увеличения матрицы CSI.

Необходимо эффективно сжимать информацию о состоянии канала на пользовательском оборудовании перед передачей и реконструировать его на базовой станции. В матрице CSI большое количество элементов близких к нулю. Это связано с разреженностью в MIMO-каналах, что может быть использовано для сжатия. В [1], [2] матрица преобразуется с помощью дискретного преобразования Фурье, в этой области сохраняется свойство разреженности. Учитывая, что временные задержки между сигналами лежат в ограниченном интервале, только некоторые строки содержат значения, не близкие к нулю, остальные можно не учитывать.

Для решения задачи потребовалась реализация модели канала на основе библиотеки Quadriga [3]. Были предложены алгоритмы из области сжатия изображений и снижения размерности матриц, основанные на: дискретном преобразовании Фурье (Discrete Fourier Transform, DFT), дискретном вейвлет-преобразовании (Discrete Wavelet Transform, DWT) на основе вейвлетов Добеши, сингулярном разложении (Singular Value Decomposition, SVD).

II. Постановка задачи

A. Формальная постановка задачи

Дано: Система MIMO с N_{tx} передающими антеннами на стороне базовой станции и N_{rx} приёмными антеннами на стороне мобильного терминала. Применяется цифровая модуляция OFDM (Orthogonal frequency-division multiplexing), где сигнал формируется N_{sc} поднесущими. При $N_{\text{tx}} \geq 1$, $N_{\text{rx}} = 1$ принимаемый сигнал на i-й поднесущей:

$$y_i = \mathbf{h}_i^{\mathsf{H}} \mathbf{v}_i x_i + n_i, \text{ где:}$$

$\mathbf{h}_i^{\mathsf{H}} \in \mathbb{C}^{N_{\text{tx}} \times 1}$ - коэффициент передачи канала
$\mathbf{v}_i \in \mathbb{C}^{N_{\text{tx}} \times 1}$ - вектор прекодирования
$x_i \in \mathbb{C}$ - вектор данных
$n_i \in \mathbb{C}$ - шум на приёмной антенне
$\mathbf{H} = [\mathbf{h}_1 \dots \mathbf{h}_i]^{\mathsf{H}} \in \mathbb{C}^{N_{\text{tx}} \times N_{\text{sc}}}$ - матрица канала
Найти: $\widetilde{\mathbf{H}}$ - сжатая матрица канала

III. Решения задачи

A. Моделирование канала и создание набора данных

Данные для исследования и анализа каналов связи часто получаются либо путем компьютерного моделирования с использованием специализированных инструментов, либо путем проведения реальных измерений в соответствующей среде.

В данной работе мы используем программное обеспечение QuaDRiGa для генерации матриц CSI в различных условиях, таких как прямая видимость (LOS) и отсутствие прямой видимости (NLOS). Это позволяет нам создавать реалистичные наборы данных для анализа поведения каналов связи в различных сценариях.

Определяем основные параметры моделирования, такие как конфигурация антенн, частотные диапазоны, количество поднесущих и т.д. Затем задаем параметры сценария, такие как характеристики среды (тип местности и зону покрытия) и движение мобильного терминала (линейное или криволинейное движение) I.

Таблица I
Параметры сценария моделирования

Характеристика	Значение (Indoor/Outdoor)
Базовая станция	32 антенны
Мобильный терминал	1 антенна
Частота	300 МГц, 5.3 ГГц
Ширина полосы	20 МГц
Количество поднесущих	128...1024
Интервал передачи CSI	5 с
Скорость абонента	0.9 м/с, 0.001 м/с
Путь	линейный, кривой
Сценарий среды	LOS/NLOS

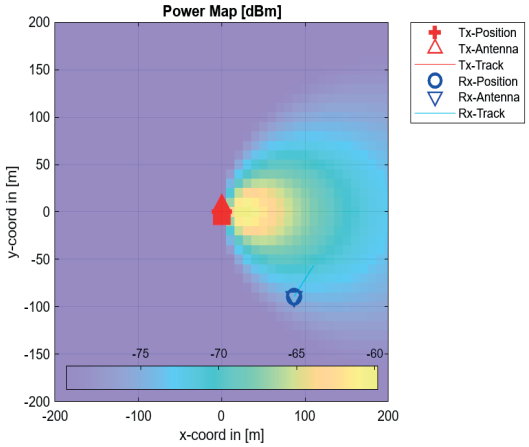

Рис. 1. Тепловая карта мощности сигнала, позиционирование базовой станции и мобильного терминала в пространстве

После установки параметров происходит генерация матриц CSI для каждого сценария. Эти матрицы оценивают характеристики канала, такие как амплитуда и фаза сигнала, в зависимости от параметров среды и расположения антенн. Строим график, чтобы визуально представить распределение сигнала в пространстве (см. Рисунок 1). Полученные данные сохраняются для последующего анализа.

B. Методы сжатия

a) Методы на основе DFT: Вход: Матрица канальной передачи $\mathbf{H}$ и порог усечения k.

Выход: Сжатая матрица $\hat{\mathbf{H}}$ и метрики (корреляция, SNR, NMSE).

- Метод №1 исключает центральные строки спектра.
- Метод №2 сохраняет элементы спектра с наибольшими амплитудами, отсортированными по значению.
- Метод №3 сохраняет строки спектра с наибольшей суммарной амплитудой.

b) Методы на основе SVD: Вход: Матрица канальной передачи $\mathbf{H}$ и порог усечения k.

Выход: Приближенная матрица $\hat{\mathbf{H}}$ и метрики (корреляция, SNR, NMSE).

Шаги: - Выполнить сингулярное разложение: $\mathbf{H} = U\Sigma V^*$. - Обрезать U, Σ, и V^* до первых k столбцов. - Вычислить приближенную матрицу: $\hat{\mathbf{H}} = U_k \Sigma_k V_k^*$.

- Метод №1: Выполняется прямое сингулярное разложение матрицы $\mathbf{H}$ и обрезка до k сингулярных значений.
- Метод №2: Сначала применяется DFT к матрице $\mathbf{H}$, затем сингулярное разложение для частотной области.
- Метод №3: Матрица $\mathbf{H}$ разделяется на действительную и мнимую части, каждая из которых обрабатывается отдельно с использованием сингулярного разложения.

c) Методы на основе DWT: Вход: Матрица канальной передачи $\mathbf{H}$ размера $N_{\text{tx}} \times N_{\text{sc}}$.

Выход: Приближенная матрица $\mathbf{H}_{\text{rec}}$ и метрики корреляции, SNR и NMSE между $\mathbf{H}$ и $\mathbf{H}_{\text{rec}}$.

Шаги: 1. Выполнить DWT для матрицы $\mathbf{H}$. 2. Сохранить наибольшие k значений вейвлет-коэффициентов. 3. Сформировать вектор позиций $\mathbf{t}$. 4. Восстановить вектор вейвлет-коэффициентов с нулями на отброшенных местах. 5. Выполнить обратное DWT для восстановления матрицы $\mathbf{H}_{\text{rec}}$.

- Метод №1: Применяет DWT к исходной комплексной матрице $\mathbf{H}$ в целом.
- Метод №2: Применяет DWT отдельно к действительной (Re) и мнимой (Im) частям матрицы $\mathbf{H}$.

C. Метрики

Для подсчёта разницы между восстановленной матрицей $\hat{\mathbf{H}}$ и исходной матрицей канала $\mathbf{H}$ используется:

Нормированная среднеквадратичная ошибка (NMSE):

$$\text{NMSE} = \mathbb{E}\left[\frac{\|\mathbf{H} - \hat{\mathbf{H}}\|_2^2}{\|\mathbf{H}\|_2^2}\right].$$

Где $\mathbb{E}$ обозначает математическое ожидание, $\|\cdot\|_2$ обозначает квадратную норму ℓ_2 матрицы.

Корреляция (Косинусное сходство):

$$\rho = \mathbb{E}\left[\frac{1}{N_{\text{sc}}}\sum_{i=1}^{N_{\text{sc}}}\frac{\hat{\mathbf{h}}_i^H \mathbf{h}_i}{\|\hat{\mathbf{h}}_i\|_2 \|\mathbf{h}_i\|_2}\right]$$

Где $\mathbf{h}_i$ и $\hat{\mathbf{h}}_i$ — i-ые строки матриц $\mathbf{H}$ и $\hat{\mathbf{H}}$ соответственно, $\cdot^H$ - сопряженно-транспонированная матрица, $\|\cdot\|_2$ обозначает квадратную норму ℓ_2.

Отношение сигнал/шум (SNR)

Отношение мощности полезного сигнала к мощности шума:

$$SNR = \frac{P_{\text{signal}}}{P_{\text{noise}}} = \frac{A_{\text{signal}}^2}{A_{\text{noise}}^2},$$

где P — средняя мощность, A — среднеквадратичное значение амплитуды. Оба сигнала измеряются в полосе пропускания системы.

Обычно отношение сигнал/шум выражается в децибелах (дБ):

$$SNR(\text{dB}) = 10\log_{10}\left(\frac{P_{\text{signal}}}{P_{\text{noise}}}\right).$$

IV. Практическая часть

A. Описание стенда

a) Используемые аппаратные средства: ASUS Vivobook K6500ZC, процессор Intel Core i5-12500H 2,5 ГГц, видеокарта: NVIDIA GeForce RTX 3050 Laptop GPU 4GB, оперативная память: 16.0 ГБ, постоянная память: 512,0 ГБ SSD.

b) Используемое программное обеспечение: операционная система: Microsoft Windows 11 Pro, версия: 10.0.22631, среда моделирования: MATLAB Version: R2019, библиотеки: QuaDRiGa v2.8.1.

B. Эксперименты

Мы генерируем два типа матриц каналов на центральной частоте 300 МГц и полосе пропускания 20 МГц с 128 и 1024 поднесущими. Предполагаем сценарий Berlin Urban Macro (UMa) с прямой видимостью (LOS) в модели канала QuaDRiGa, где количество передающих антенн равно M = 32, а UE движется по окружности или по линейному пути в разных направлениях.

В экспериментах во всех реализованных методах мы сохраняем наибольшие элементы матрицы канала. Порог обрезания устанавливается на уровне 0-99%. Разница между реконструированным каналом $\hat{\mathbf{H}}$ и входным $\mathbf{H}$ определяется с помощью нормализованной MSE, измеряемой в 10log10(дБ), корреляции (косинусного сходства) и отношения сигнал-шум, измеряемого в 10log10(дБ).

C. Выводы

В статье [12] утверждается, что корреляция действительной части значений CSI эквивалентна корреляции мнимой части значений матрицы канала. На основании этого предположения были проведены эксперименты с методами SVDmethod3 и DWTmethod1, которые опровергли данное утверждение, указав на несоответствие с физическими свойствами матрицы канала. В качестве дополнительного эксперимента было выполнено сравнение сжатия матрицы методом SVD в области Фурье с матрицей канала. Это сравнение не выявило значительных отличий, что указывает на корректность использования области Фурье для анализа матрицы канала.

a) Допустимые значения метрик: Значения варьируются в зависимости от конкретного применения и контекста, но существуют общие ориентиры:

NMSE < -20 дБ, SNR > -1 дБ, Corr > 0.9: В большинстве случаев считается хорошим, так как ошибка восстановления очень мала.

NMSE > -10 дБ, SNR < -3 дБ, Corr < 0.7: Плохое качество восстановления. Ошибка слишком велика, что может быть неприемлемо для беспроводного канала передачи данных.

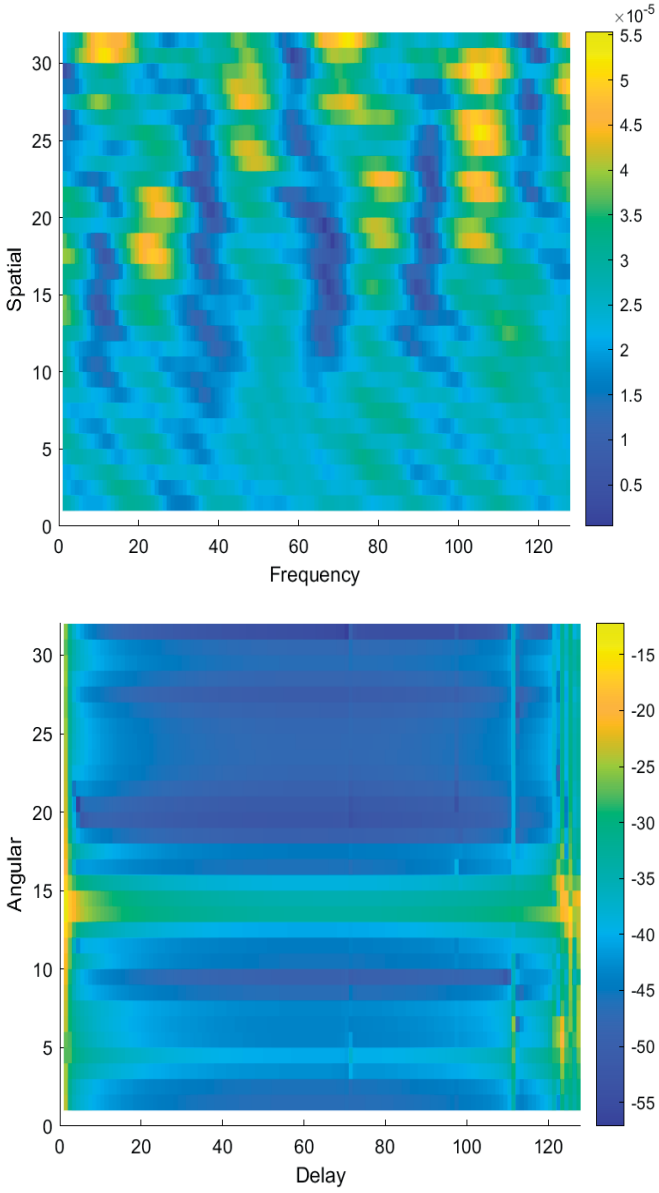

Рис. 2. Модуль матрицы канала и DFT-преобразованной матрицы канала |H|

b) Общие выводы для всех методов:

1) Большинство методов достигают значения NMSE около -20 дБ при сохранении 10-20% данных. DFTmethod2 и DWTmethod2 демонстрируют наименьшие значения NMSE при том же уровне сохранения данных, что указывает на их высокую эффективность в восстановлении сигнала.
2) У всех методов ухудшается отношение SNR приблизительно на 1 дБ и меньше, при уровне сохранения данных около 10%. DFTmethod2 и DFTmethod3 здесь показывают лучшие результаты, достигая малых значений SNR при небольшом проценте сохранения данных.
3) Все методы достигают коэффициента корреляции, близкого к 1, при уровне сохранения данных

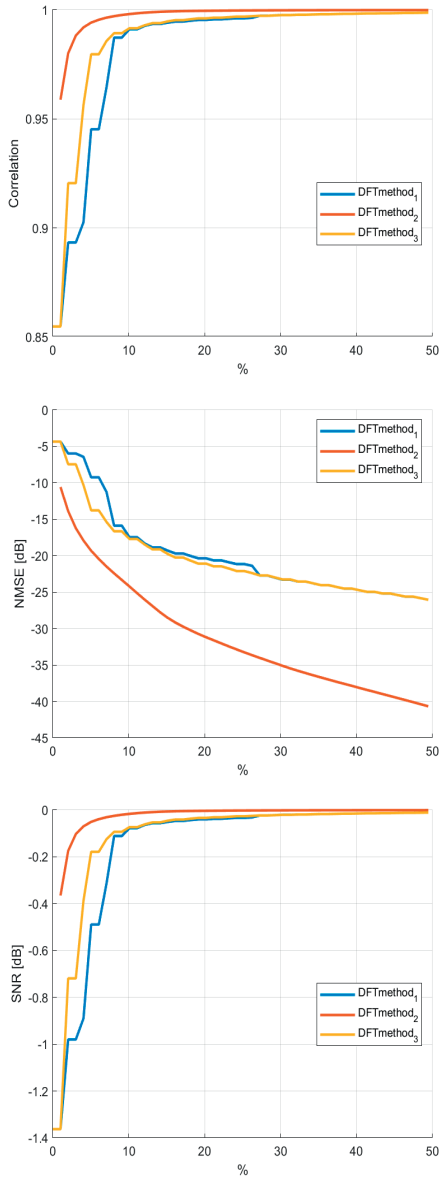

Рис. 3. Полученная корреляция, среднеквадратичная ошибка, отношение сигнал/шум в методах сжатия на основе DFT.

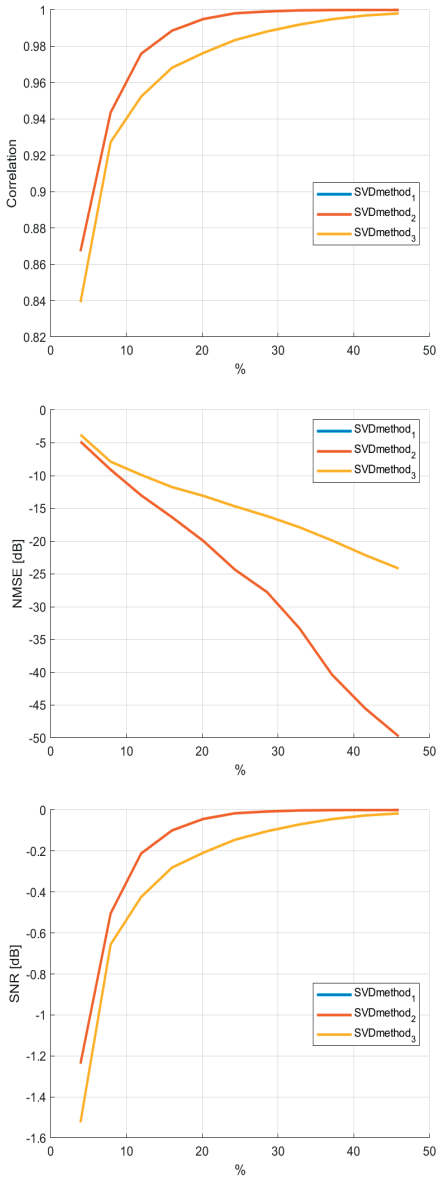

Рис. 4. Полученная корреляция, среднеквадратичная ошибка, отношение сигнал/шум в методах сжатия на основе SVD.

около 15-20%. Это указывает на высокое сходство векторов после восстановления сигнала, однако не отражает качества данных. DFTmethod2, DFTmethod2 и DFTmethod3 достигают высоких значений корреляции быстрее, чем другие методы, что указывает на их высокую эффективность.

c) Рекомендации: Для приложений, требующих минимальной потери данных и высококачественного восстановления сигнала, рекомендуется использовать DFTmethod2 или DFTmethod3. Для приложений, где допустимы начальные потери в качестве, можно рассмотреть DWTmethod2 или SVDmethod1. В целом, все методы показывают улучшение качества восстановления сигнала, если учитывать физические свойства матрицы канала и рассматривать методы, подходящие для комплексных матриц.

Заключение

Анализ полученных результатов показал, что методы сжатия данных, основанные на дискретном преобразовании Фурье, сингулярном разложении и дискретном вейвлет-преобразовании, демонстрируют допустимые характеристики восстановления сигнала при различном уровне сжатия. В результате работы методы DFTmethod2, DFTmethod3 и SVDmethod1 показывают значения NMSE около -20 дБ при сохранении 10-20% данных, при этом отношение SNR ухудшается на 1 дБ, что является хорошим показателем качества восстановления.

Список литературы

[1] C.-K. Wen, W.-T. Shih, and S. Jin, “Deep learning for massive MIMO CSI feedback,” IEEE Wireless Communications Letters, vol. 7, no. 5, pp. 748–751, 2018.

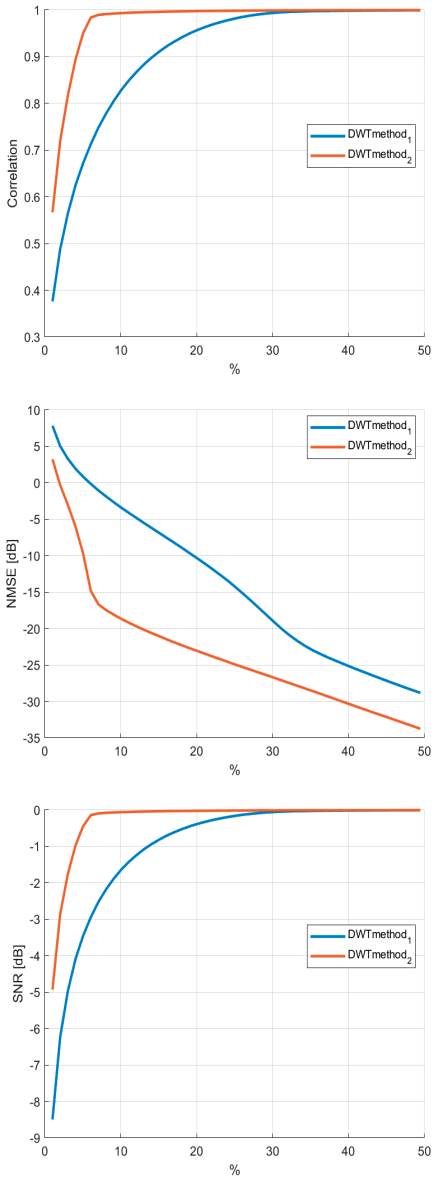

Рис. 5. Полученная корреляция, среднеквадратичная ошибка, отношение сигнал/шум в методах сжатия на основе DWT.

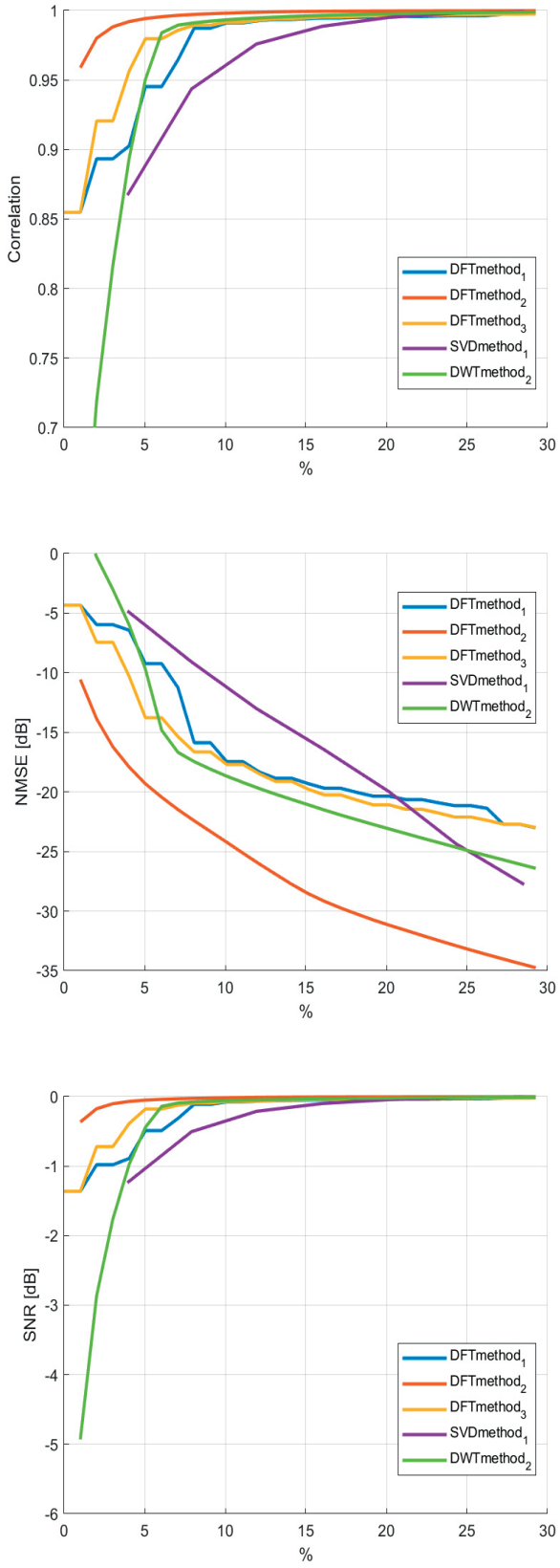

Рис. 6. Полученные метрики корреляции, среднеквадратичной ошибки, отношения сигнал/шум в методах сжатия, показавших лучшие результаты.

[2] J. Guo, C.-K. Wen, S. Jin, and G. Y. Li, "Overview of deep learning-based CSI feedback in massive MIMO systems," IEEE Transactions on Communications, vol. 70, no. 12, pp. 8017–8045, 2022.

[3] S. Jaeckel, L. Raschkowski, K. Börner, and L. Thiele, "QuaDRiGa: A 3-D multi-cell channel model with time evolution for enabling virtual field trials," IEEE Transactions on Antennas and Propagation, vol. 62, no. 6, pp. 3242–3256, 2014.

[4] T. Peters, Data-driven science and engineering: machine learning, dynamical systems, and control: by S.L. Brunton and J.N. Kutz, Cambridge, Cambridge University Press, 2019, vol. 60, no. 4.

[5] T. Wang, C.-K. Wen, S. Jin, and G. Y. Li, "Deep learning-based CSI feedback approach for time-varying massive MIMO channels," IEEE Wireless Communications Letters, vol. 8, no. 2, pp. 416–419, 2018.

[6] S. Mourya, S. Amuru, and K. K. Kuchi, "Multi-Task Learning for Multi-User CSI Feedback," arXiv preprint arXiv:2211.08173, 2022.

[7] X. Han, Z. Wang, D. Li, W. Tian, X. Liu, W. Liu, S.

Jin, S. Jia, Z. Zhang, and N. Yang, "AI enlightens wireless communication: A transformer backbone for CSI feedback," China Communications, 2024.
[8] 3GPP, "5G System Overview," [Online]. Available: https://www.3gpp.org/technologies/5g-system-overview. [Accessed: 13-May-2024].
[9] M. Shafi, A. F. Molisch, P. J. Smith, T. Haustein, P. Zhu, P. De Silva, F. Tufvesson, A. Benjebbour, and G. Wunder, "5G: A tutorial overview of standards, trials, challenges, deployment, and practice," IEEE Journal on Selected Areas in Communications, vol. 35, no. 6, pp. 1201–1221, 2017.
[10] 3GPP, "Release 18," [Online]. Available: https://www.3gpp.org/specifications-technologies/releases/release-18. [Accessed: 13-May-2024].
[11] 3GPP, "3rd Generation Partnership Project; Technical Specification Group Radio Access Network; NR; Physical layer procedures for data," [Online]. Available: https://atisorg.s3.amazonaws.com/archive/3gppdocuments/Rel16/ATIS.3GPP.38.214.V1620.pdf. [Accessed: 13-May-2024].
[12] Y. Sun, W. Xu, L. Liang, N. Wang, G. Y. Li, and X. You, "A lightweight deep network for efficient CSI feedback in massive MIMO systems," IEEE Wireless Communications Letters, vol. 10, no. 8, pp. 1840–1844, 2021.

Прогнозирование временных характеристик прикладных сетевых сервисов

В.О. Писковский
ВМК МГУ им. М.В. Ломоносова
Москва, Россия
vpiskovski@lvk.cs.msu.ru

Е.О. Лычева
ВМК МГУ им. М.В. Ломоносова
Москва, Россия
ders.1981@mail.ru

В.М. Могиленец
ВМК МГУ им. М.В. Ломоносова
Москва, Россия
mogilinez17@yandex.ru

Аннотация: Функционирование распределенных вычислительных систем в соответствии с предъявляемыми к ним техническими требованиями определяется эффективностью размещения сетевого сервиса на оборудовании в сети.

При этом пользователям сети предоставляется набор информационных услуг с заданным уровнем качества и надёжности:

1. доступ к информации «в любое время в любом месте», то есть любому участнику процесса управления при наличии прав доступа при возникновении потребности и вне зависимости от места его нахождения;
2. информационное взаимодействие между автоматизированными системами;
3. своевременный мониторинг и анализ данных источников различного типа;
4. переход от централизованной схемы «загрузка с очисткой данных — анализ — распространение» к схеме «распределенные размещение и предобработка данных, и по необходимости — последующие загрузка, анализ и распространение».

Одним из основных факторов обеспечения приведенных выше типов услуг с заданными вероятностно-временными характеристиками являет прогноз временных характеристик выполнения прикладных сетевых сервисов на широком спектре оборудования и платформ виртуализации.

В работе исследуются методы прогноза временных характеристик прикладных сетевых сервисов в зависимости от данных эксплуатации сетевого сервиса на используемых сетевых ресурсах, текущего и прогнозируемого состояния аппаратного и программного обеспечения:

5. Обучение моделей случайного леса
6. Усеченное сингулярное разложение
7. Анализ главных компонент
8. Нейронные сети на основе многослойного персептрона, свёрточной сети.

Ключевые слова: прогноз временных характеристик, обучение моделей случайного леса, усеченное сингулярное разложение, метод главных компонент, нейронные сети на основе многослойного персептрона, свёрточные сети

1. ВВЕДЕНИЕ

Развитие технологии распределенных облачных вычислений, реализация отказоустойчивых сетевых сервисов в виде распределенных сетевых услуг приводит к необходимости перехода от атомарных систем, локализованных на выделенных для этого аппаратных единицах, к так называемым «логическим/сетевым» распределенным системам, которые «живы» пока есть обеспечивающая их существование сеть с достаточной степенью избыточности. Такие системы используют преимущества технологии виртуализации и распределены по различным серверным мощностям. Если какой-то экземпляр сервиса оказался недоступен, то управление сетью активизирует его резервную копию или соответствующий клон, в зависимости от уровня обслуживания. Иными словами, сетевые сервис, функция, услуга существуют и работают не на конкретной единице оборудования, а в сети и пока существует сеть, опирающаяся на определенный уровень избыточности в ИКТ инфраструктуре.

Обеспечение функционирования распределенных вычислительных систем в соответствии с предъявляемыми к ним техническими требованиями определяется эффективностью размещения сетевого сервиса на оборудовании в сети. Программируемость полученной таким образом вычислительной среды — один из ключевых факторов развития информационных инфраструктур. Широкий спектр приложений, таких как службы Интернета вещей, методы машинного обучения, анализа данных и научных вычислений и прочие, используют модель вычислений «Функции как услуга» (Function as a Service, FaaS) облачных платформ Amazon Lambda, Azure, Google Functions и другие. Широкое распространение получают «бессерверные вычисления», что предполагает разбиение приложения на небольшие «функции», локализованные на разных узлах сети. Работа таких приложений управляется единой облачной платформой.

Ключевым вопросом для организации работы распределенных приложений, особенно в труднодоступных малонаселенных районах, например, Крайнего севера или горных массивов, является решение задачи локализации прикладных сервисов, сетевых функций по узлам сети. Задача такой

локализации должна учитывать доступные вычислительные ресурсы, прогноз времени выполнения, характеристики энергоснабжения, вопросы охлаждения, состояние сетевых соединений, принятые требования качества сервиса (QoS, SLA), а также другие характеристики.

В докладе рассматривается построение прогнозирования времени выполнения задачи на узлах сети с учетом указанных выше характеристик. В качестве исходных данных для построения и тестирования методов прогнозирования используются опубликованные данные трасс работы распределенных сетевых сервисов на типовом оборудовании облачных сетевых структур и сетей доступа к ним [3].

2. ПОСТАНОВКА ЗАДАЧИ

Задача прогнозирования временных характеристик прикладных сетевых сервисов - найти функцию $F_k(\vec{x})$, наилучшим образом приближающей данные на тестовой выборке: $F_k(\vec{x}) = \mathrm{argmin}_{F_k} \| F_k(\vec{x}) - y_k \|$, где $\vec{x} \in R^D$ – заданные D метрик, по которым прогнозируется k численных характеристик. Качество прогноза оценивается в норме l_2.

В работе рассматривается задача прогнозирования временных характеристик распределенных сетевых сервисов по предоставленным данным журнала эксплуатации аппаратных ресурсов, на которых расположены сервисы. В качестве входных данных рассматривались 45 параметров системы управления заданиями Slurm [1]. Для восполнения пропущенных данных применяются методы на основе матричных разложений. Полученные данные подаются на вход методам, оптимизированным и отлаженным на полных наборах информации журналов эксплуатации доступных из открытых источников [2].

Решение поставленной задачи содержит два этапа. На первом, предварительном этапе разрабатываются методы прогнозирования на полных наборах. Разрабатываемые методы и наборы, на которых проводилась их отладка, изложены в разделе 3.

3.МЕТОДЫ РЕШЕНИЯ

На первом этапе рассматривались методы прогнозирования параметров, описывающих состояние сервера и уровень качества предоставляемого сервиса, на основе имеющихся замеров. Набор данных содержал сведения о работе серверов тестового стенда во время работы приложений видео по запросу и системы управления базами данных [4].

Тестовый стенд состоит из шести серверов и одной клиентской машины, связанных OpenFlow сетью, часы машин синхронизируются с применением протокола NTP.

Сервис видео по запросу распределяет роли между серверами следующим образом: два сетевых хранилища, три веб-сервера и кодирующих машины, один HTTP-сервер. В хранилищах расположены 10 самых просматриваемых видеоролика 2013 года длительностью от 33 секунд до 5 минут. Два из шести вычислителей выполняли роль хранилища информации для работы сервиса, на трёх запускалось веб-приложение, последний использовался в качестве балансировщика нагрузки. Во время работы приложения собирались показатели с каждого из вычислителей. Также собиралась информация из openflow сети, а именно: количество полученных и переданных байт с каждого порта и количество переданных и полученных пакетов на каждый порт. В итоге получилось 1256 показателя, которые были занесены в таблицу пространства признаков.

Для базы данных все сервера выполняют одинаковые роли. Сервис запускался на всех шести машинах. Аналогично сервису видео по запросу регистрировались показателей вычислителей и сети (1724 метрики), а количество регистрируемых записей составило порядка 20 тысяч. В базе данных хранится 10 миллионов уникальных ключей, равномерно распределённых в 32-битном пространстве. Хранимые данные имеют размер 1024 бит. Каждая пара ключ-значение хранится на трёх серверных машинах, для идентификации которых используется согласованное хэширование. Подробно о стенде и данных см. [3].

Параметры работы устройств, на которых запущены сервисы, получены из ядра операционной системы Linux, работающей на серверах [4]. Ядро Linux (Ubuntu Server 14.04 64 бит) предоставляет приложениям доступ к ресурсам, таким как центральный процессор (ЦП), память и сеть, а также планирует запросы к этим ресурсам. Для доступа к структурам данных ядра используется procfs. Procfs основан на абстракции файловой системы Unix. Таким образом, к данным ядра можно получить доступ, как если бы они были структурированы в каталогах, и хранились в файлах. Например, для каждого процесса procfs включает каталог, имя которого соответствует идентификатору процесса.

В работе для каждой машины используется набор функций Xsar, созданный на основе отчета об активности системы (SAR). При чтении данных через procfs SAR вычисляет различные системные показатели в течение настраиваемого интервала. Примерами таких показателей являются использование ядра ЦП, памяти и пространства подкачки, статистика дискового ввода-вывода и сетевого интерфейса. Набор функций Xsar включает только числовые функции, возвращаемые инструментом SAR.

В работе [4] для предоставления услуги VoD используется медиаплеер VLC 2.1.6, а для работы базы данных выбрана система Voldemort 1.10.22. Показатели уровня обслуживания измеряются на клиентском устройстве.

Для сервиса VoD исследуются следующие характеристики: (1) частота кадров видео (кадров/сек), представляет собой количество отображаемых видеокадров за единицу времени, (2) скорость буферизации аудио (буферов/сек), то есть количество воспроизводимых аудиобуферов за единицу времени, (3) количество прочитанных из сети байтов, (4) средняя задержка отображения), (5) средняя задержка воспроизведения звука.

Эти метрики не измеряются напрямую, а вычисляются на основе событий VLC, таких как отображение видеокадра на дисплее клиента. Эти параметры выбраны исходя из того, что интерес представляют показатели уровня обслуживания, измеряемые на стороне клиента.

Для сервиса БД типа «Ключ-значение» (Key-Value, KV, Voldemort) в ходе эксперимента измеряются задержки операции чтения и записи, инициированных клиентом. Метрика вычисляется на основе среднего значения задержек нескольких запросов, отправленных в хранилище.

Был проведён обзор методов прогнозирования, в результате которого, на основе работы [5] сделан выбор в пользу модели случайного леса (Рис. 1)

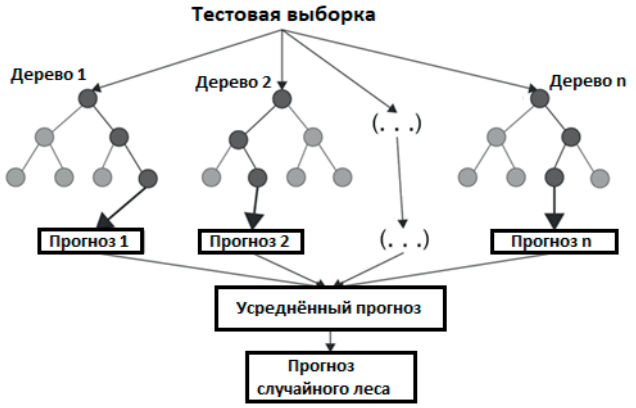

Рисунок 1. Модель случайного леса

Существенной частью работ было понизить размерность входных данных без значительной потери точности, применить наиболее эффективные методы прогнозирования данных. В качестве методик первого этапа рассматривались анализ главных компонент (Principal Component Analysis, PCA) и усечённое сингулярное разложение (Truncated Singular Value Decomposition, TSVD).

2.1 Анализ главных компонент

Рассмотрим набор данных наблюдений $\{x_n\}$, где n = 1,...,N, а x_n – Евклидова переменная размерностью D. Цель метода — спроецировать данные на пространство, имеющее размерность M < D, при этом максимизируя объяснённую дисперсию прогнозируемых данных (в альтернативной постановки – минимизировать ошибку проецирования). Для начала рассмотрим проекцию на одномерное пространство (M = 1).

Направление нового пространства определяется D-мерным вектором u_1, который для удобства (и без ограничения общности) считается единичным, так чтобы $u_1^T u_1 = 1$ (1). Каждая точка данных x_n затем проецируется на значение $u_1^T x_n$. Среднее значение прогнозируемых данных может быть вычислено как $u_1^T \bar{x}$, где

$$\bar{x} = \frac{1}{N}\sum_{n=1}^{N} x_n \quad (2),$$

а дисперсия прогнозируемых данных определяется выражением

$$\frac{1}{N}\sum_{n=1}^{N}\{u_1^T x_n - u_1^T \bar{x}\}^2 = u_1^T S u_1 \quad (3)\ ,$$

где S – матрица ковариации:

$$S = \frac{1}{N}\sum_{n=1}^{N}(x_n - \bar{x})(x_n - \bar{x})^T \quad (4).$$

Теперь необходимо максимизировать прогнозируемую дисперсию $u_1^T S u_1$ относительно u_1. Для соблюдения наложенного ранее ограничения (1) введём множитель Лагранжа:

$$u_1^T S u_1 + \lambda_1(1 - u_1^T u_1) \quad (5).$$

Полагая производную по u_1 равной нулю, стационарной точкой окажется

$$S u_1 = \lambda_1 u_1 \quad (6).$$

Отсюда, u_1 – собственный вектор ковариационной матрицы. Также заметим, что

$$u_1^T S u_1 = \lambda_1 \quad (7).$$

Следовательно, наибольшая дисперсия достигается при u_1, соответствующем наибольшему собственному значению ковариационной матрицы.

Остальные главные компоненты определяются поэтапно: выбирая каждое новое направление так, чтобы оно максимизировало прогнозируемую дисперсию среди всех возможных направлений, ортогональных уже рассмотренным. В общем случае, рассматривая размерность M, главные компоненты будут представлять собой M собственных векторов ковариационной матрицы, соответствующих её M наибольшим собственным значениям.

Описанный способ понижения размерности пространства называется анализом главных компонент [6].

2.2 Усечённое сингулярное разложение

Пусть матрица M порядка $m \times n$ состоит из элементов из поля K, где K —— либо поле вещественных чисел, либо поле комплексных чисел.

Неотрицательное вещественное число σ называется сингулярным числом [7] матрицы M тогда и только тогда, когда существуют два вектора единичной длины $u \in K^m$ и $v \in K^n$ такие, что:

$Mv = \sigma u$ (8),

$M^*u = \sigma v$ (9).

Векторы u и v называются левым и правым сингулярными векторами, соответствующими сингулярному числу σ. Сингулярным разложением матрицы M порядка $m \times n$ называется разложение следующего вида:

$M = U\Sigma V^*$ (10),

где Σ – матрица размера $m \times n$, в которой элементы, лежащие на главной диагонали — это сингулярные собственные числа, а остальные элементы нулевые, а матрицы U (порядка m) и V (порядка n) —— это две унитарные матрицы, состоящие из левых и правых собственных сингулярных векторов. Причем диагональные элементы σ_i матрицы Σ упорядочены следующим образом: $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_n)$ $\sigma_1 > \sigma_2 > \ldots > \sigma_r = \ldots = \sigma_n = 0$, где $r = \mathrm{rank}(A)$.

Для уменьшения размерности данных требуется приближение заданной матрицы M некоторой другой матрицей M_k с заранее заданным рангом k.

Теорема Эккарта-Янга [7]:

Если потребовать, чтобы такое приближение было наилучшим в том смысле, что норма Фробениуса разности матриц M и M_k минимальна при ограничении $\mathrm{rank}(M_k) = k$, то оказывается, что наилучшая такая матрица M_k получается из сингулярного разложения матрицы M по формуле: $M_k = U\Sigma_k V^*$, где Σ_k – матрица Σ, в которой заменили нулями все диагональные элементы, кроме k наибольших элементов: $\Sigma_k = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_k, 0, \ldots, 0)$. Выражение для матрицы M_k можно переписать в такой форме: $M_k = U_k\Sigma_k V_k^*$ (19), где матрицы U_k, Σ_k и V_k получаются из соответствующих матриц в сингулярном разложении матрицы M обрезанием до k первых столбцов – такое разложение называется усечённым сингулярным разложением.

Таким образом, приближая матрицу M матрицей меньшего ранга, выполняется сжатие информации, содержащейся в M: матрица M размера $m \times n$ заменяется меньшими матрицами размеров $m \times k$ и $k \times n$ и диагональной матрицей с k элементами. При этом сжатие происходит с потерями — в приближении сохраняется лишь наиболее существенная часть матрицы M.

2.3 Нейронные сети

Для сравнительного анализа были выбраны два класса нейронных сетей: многослойный персептрон (multilayer perceptron, MLP)[8] и свёрточная нейронная сеть (convolution neural network, CNN)[8].

2.3.1 Многослойный персептрон, краткое описание работы

Нейронную сеть, построенную на основе MLP, можно представить в виде связного ориентированного графа, разделенного на слои. Вершины этого графа --- нейроны. Наличие ребра означает, что выход одного нейрона подаётся на вход другому. Все слои, из которых состоит граф, можно классифицировать на входной, выходной и скрытые слои (Рис.2).

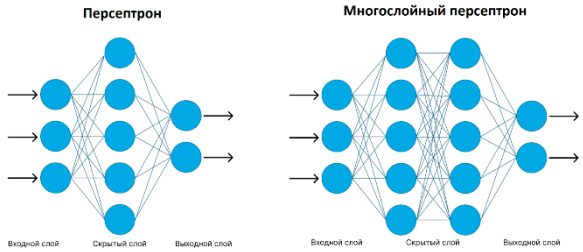

Рисунок 2. Графовое представление нейронной сети

Каждый нейрон представляет собой функцию вида:

$$\alpha(\vec{x}, \vec{\omega}) = f\left(\sum_{i=1}^{n} \omega_i * x_i\right)$$

Где x_i – значение сигнала, которое поступило на вход нейрону, $\vec{\omega}$ – набор весов соответствующий данному нейрону, f – функция активации, принимающая решение о передаче сигнала дальше.

Процесс обучения представляет собой подбор весов для каждого нейрона при помощи метода градиентного спуска для минимизации функции потерь, имеющей вид:

$$F(y, \hat{y}) = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}_i|$$

Где N – количество прогнозов, y и $\hat{y}$ – вектора истинных и спрогнозированных значений соответственно.

2.3.2 Сверточная нейронная сеть, краткое описание работы

Строение сверточной нейронной сети похоже на многослойный персептрон. В MLP каждому нейрону соответствует свой набор весов, в CNN каждому слою нейронов соответствует свой набор весов. Это позволяет уменьшить количество параметров модели. Для этого вместо операции суммы в функции нейрона используют операцию свертки матриц.

В данной работе использовалась одномерная свертка, то есть каждый нейрон слоя реализовывал линейную комбинацию части признаков:

$$\alpha_k(\vec{x}, \vec{\omega}) = f\left(\sum_{i=0}^{l-1} \omega_{i+1} * x_{k+i}\right)$$

Где k – номер нейрона в слое, $\vec{\omega}$ – вектор весов слоя, $\vec{x}$ – вектор сигналов предыдущего слоя, l – длина вектора весов слоя, f -- функция активации.

2.3.3 Критерии сравнения

Основными критериями для сравнения предсказательных способностей алгоритма является его точность и время, необходимое для достижения заданной точности. В данной работе под точностью будет пониматься средняя абсолютная ошибка

mae = $\frac{1}{N}|y_k - \widehat{y_k}|$, где $\widehat{y_k}$ – результат предсказания k-й метрики на проверочной выборке и y_k – истинные значения k-й метрики, а под временем – количество эпох обучения, затраченного на достижение заданной точности.

Для сравнения алгоритмов в течении процесса обучения были сформулированы следующие критерии:

9. Склонность к переобучению: модель склонна к переобучению если:

$$\exists L: \forall\, n > L: mae_{n,y_k}^{train} < mae_{L,y_k}^{train} \Rightarrow mae_{n,y_k}^{test} > mae_{n,y_k}^{test}$$

10. Стабильность этапа обучения:

$$Stab = \frac{\frac{1}{N}\sum_{i=1}^{N}(sign(y_k^{i-1} - y_k^i)) + 1}{2}$$

где N – количество эпох обучения, $sign$(y) – возвращает 1 если y ≥ 0, 0 в противном случае, y_k^i – значение k-й точности на j-й эпохе обучения.

2.3.3 Описание экспериментов

Исходная выборка была разбита на проверочную и обучающую в соотношении 1:3. Каждая нейронная сеть обучалась 150 эпох на обучающей выборке. После каждой эпохи производилось измерение точности для проверочной выборки. Для уменьшения вероятности переобучения алгоритмов на каждом слое использовались алгоритмы регуляризации.

Для экспериментов с MLP была построена сеть из одного входного слоя, пяти скрытых (с количеством нейронов: 1024, 512, 16, 8 и 4 соответственно) и одного выходного слоя. На каждом скрытом слое использовалась L1 регуляризация с весом 0,01.

Для экспериментов с CNN была построена сеть из одного входного слоя, четырёх сверточных слоёв, с длинами векторов весов 17, 9, 3, 21 соответственно, двух полно связных слоёв, содержащих 16 и 4 нейрона соответственно, и одного выходного слоя. На каждом скрытом слое использовалась L1L2 регуляризация с весами 0,1 и 0,1.

3. РЕЗУЛЬТАТЫ ЭКСПЕРИМЕНТОВ

3.1. Методы случайного леса, PCA, TSVD

В данном разделе представлены результаты обучения моделей случайного леса [8]. В качестве критерия точности модели используется нормализованная средняя абсолютная ошибка [4]:

$$NMAE = \frac{1}{\bar{y}}\left(\frac{1}{m}\sum_{i=1}^{m}|yi - \hat{y}i|\right)*100\% \quad (20)$$

N – количество точек в выборке, y_i – реальное значение прогнозируемой величины в i-ой точке, y_i с крышечкой – значение, предсказанное моделью, y с крышечкой – среднее значение параметра y.

Количество компонент в методах, описанных в разделе 2, выбиралось исходя из необходимого их числа для сохранения 90% объяснённой дисперсии данных (Рис. 2, 3, 4, 5, 6, 7, 8).

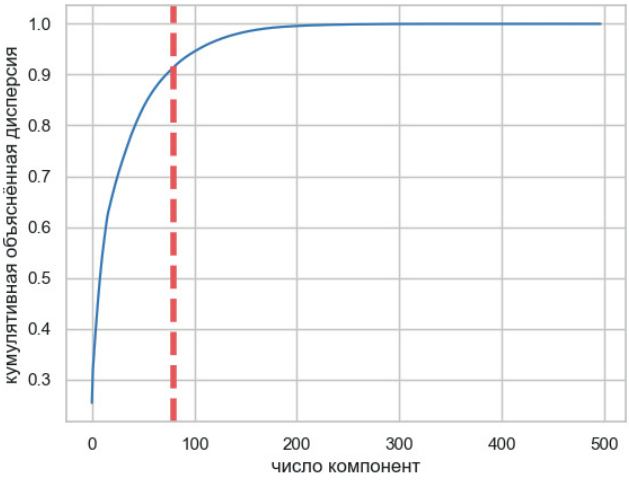

Рисунок 3. Объяснённая дисперсия PCA для средней скорости чтений из базы данных

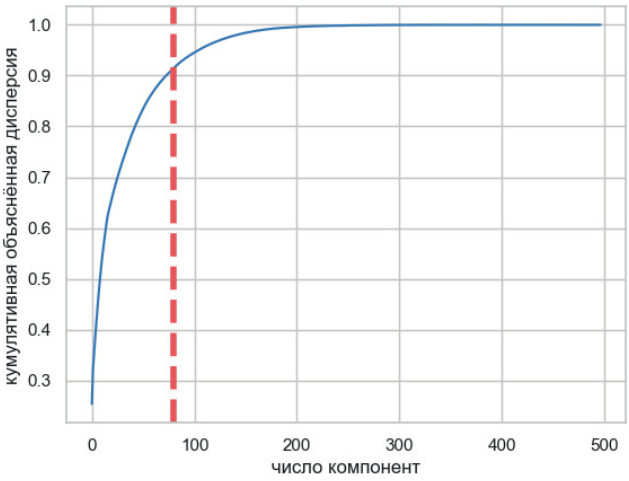

Рисунок 4. Объяснённая дисперсия PCA для средней скорости записи в базу данных

Рисунок 5. Объяснённая дисперсия TSVD для базы данных

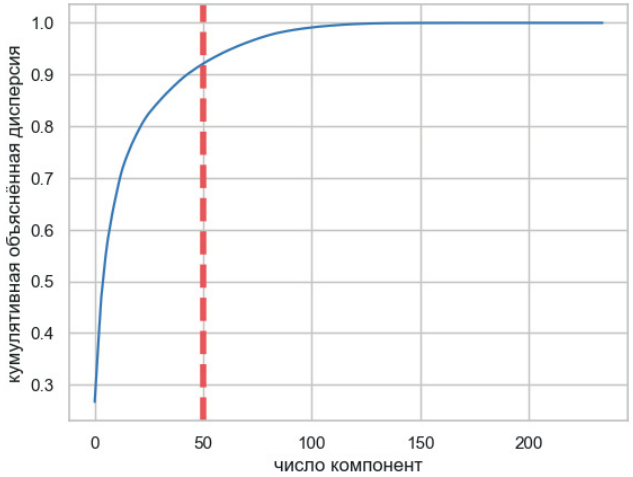

Рисунок 6. Объяснённая дисперсия PCA для задержки воспроизведения аудио

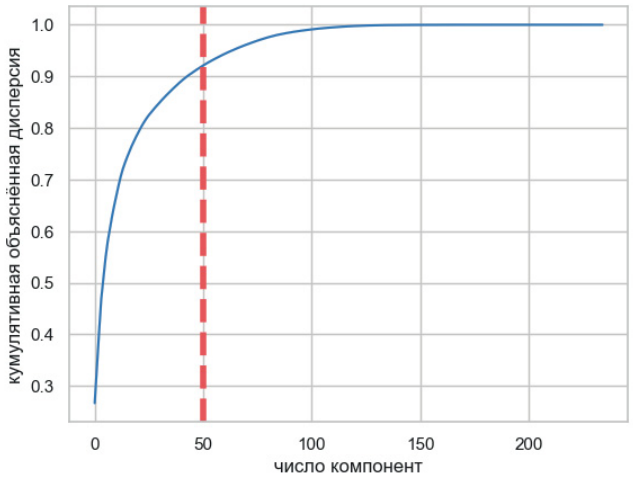

Рисунок 7. Объяснённая дисперсия PCA для числа прочитанных из сети байтов

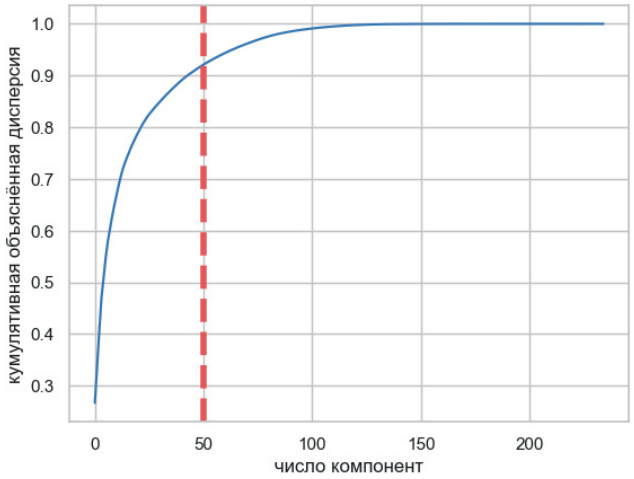

Рисунок 8. Объяснённая дисперсия PCA для количества кадров в секунду

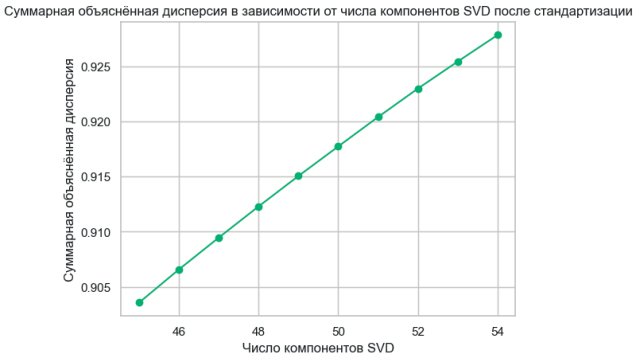

Рисунок 9. Объяснённая дисперсия TSVD для видео по запросу

При обучении моделей случайного леса были получены следующие результаты:

	Задержка операций чтения, NMAE %	Задержка операций записи, NMAE %
Случайный лес	1.86	2.03
Случайный лес + PCA (90 компонент)	1.82	1.99
Случайный лес + TSVD (76 компонент)	1.91	2.09

Таблица 1. Точность моделей для базы данных

	Задержка воспроизведения аудио, NMAE %	Количество прочитанных из сети байт, NMAE %	Число кадров в секунду, NMAE %
Случайный лес	42.5	28.5	9.86
Случайный лес + PCA (50 компонент)	35	35.8	12.7

Случайный лес + TSVD (46 компонент)	35	35.9	12.7

Таблица 2. Точность моделей для видео по запросу

Результаты показывают, что прогнозирование с использованием модели случайного леса на основе набора параметров, полученного при помощи анализа главных компонент, в 4 из 6 случаев показало улучшение результата на 0.04 – 7.5% по метрике NMAE, в 2 других случаях – ухудшение на 2.84 – 7.3%. Обучение на основе набора, построенного с использованием усечённого сингулярного разложения, привело к ухудшению качества прогноза на 0.05 – 7.4% в 5 из 6 случаев и улучшение на 7.5% в одном случае.

3.2 Нейронные сети

В данном разделе представлены результаты обучения модели MLP и CNN сетей в комбинации с методом главных компонент. Значения параметра метода PCA было принято принять за 1000, 500, 100 и 50.

3.2.1 Многослойный персептрон.

На Рис.10, 11 приведены графики изменения точности для прогнозирования метрик задач VoD и RV БД на проверочной выборке различных комбинаций методов в обычных и логарифмических шкалах. В таблицах 3, 4 приведены сводные значения точности для комбинации методов.

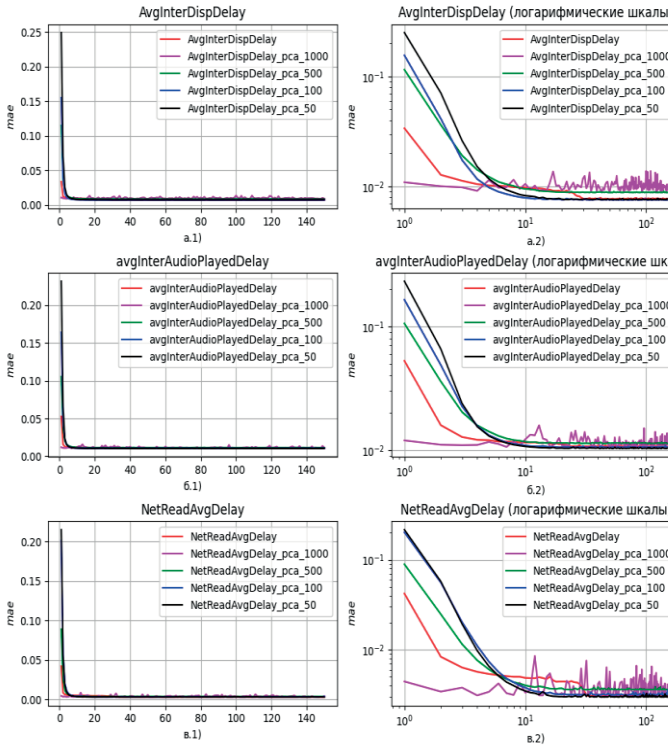

Рисунок 10. Графики точности при обучении MLP модели на метриках задачи VoD.

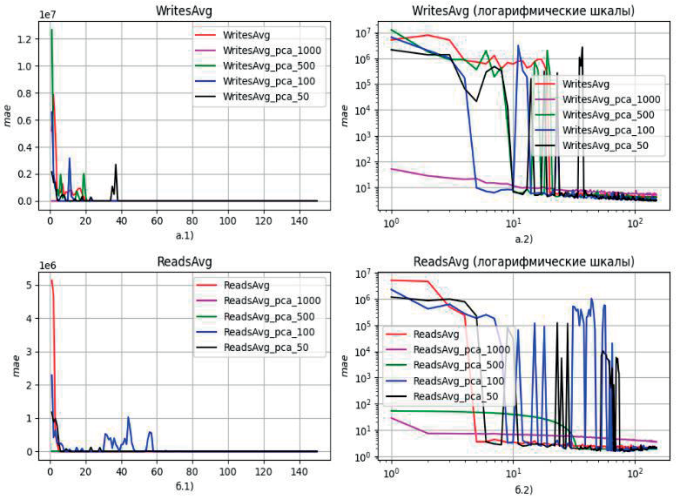

Рисунок 11. Графики точности при обучении MLP модели на метриках задачи KV.

	NetReadAvgDelay		avgInterAudioPlayedDelay		AvgInterDispDelay	
	min MAE	epoch	min MAE	epoch	min MAE	epoch
no PCA	0.0031	106	0.011	120	0.0077	48
PCA 1000	0.0031	3	0.011	53	0.009	42
PCA 500	0.0036	33	0.011	39	0.0088	134
PCA 100	0.0031	116	0.01	89	0.0076	109
PCA 50	0.01	120	0.01	92	0.0076	95

Таблица 3. Сводные значения точности для MLP на метриках задачи VoD

	WritesAvg		ReadsAvg	
	min MAE	epoch	min MAE	epoch
no PCA	4.25	147	2.07	147
PCA 1000	5.1	149	3.48	149
PCA 500	3.82	128	1.87	35
PCA 100	3.16	148	1.57	71
PCA 50	2.97	147	1.5	81

Таблица 4. Сводные значения точности для MLP на метриках задачи KV

Лучшие результаты показала модель с размерностью признакового пространства 50, добившись точности 2,97 и 1,5 на метриках WritesAvg и ReadsAvg соответственно. Модель с размерностью пространства признаков 100 показала результат хуже, чем предыдущая на 6% и 4,5% соответственно, что численно составляет 0,19 и 0,07.

На основании полученных данных, было принято решение о выборе моделью для дальнейшего сравнительного анализа модель реализующую комбинацию MLP и PCA(100).

На Рис. 12 изображен график зависимости точность на проверочной выборке для задачи прогнозирования времени выполнения задачи на супер вычислителе. При этом лучшее значение точности, которого добилась сеть, составляет 14,5 секнд.

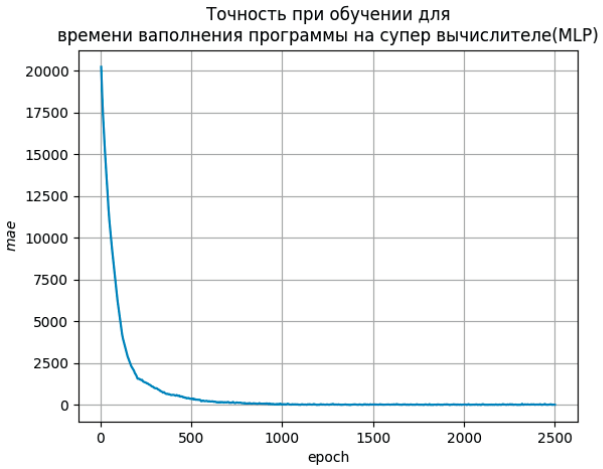

Рисунок 12. Графики точности при обучении MLP модели для задачи прогнозирования времени выполнения программы на супер вычислителе.

3.2.2 Сверточная нейронная сеть

На Рис. 13, 14 приведены графики изменения точности на проверочной выборке комбинаций методов CNN и PCA в обычных и логарифмических шкалах. В таблицах 5, 6 приведены сводные значения точности для комбинации методов.

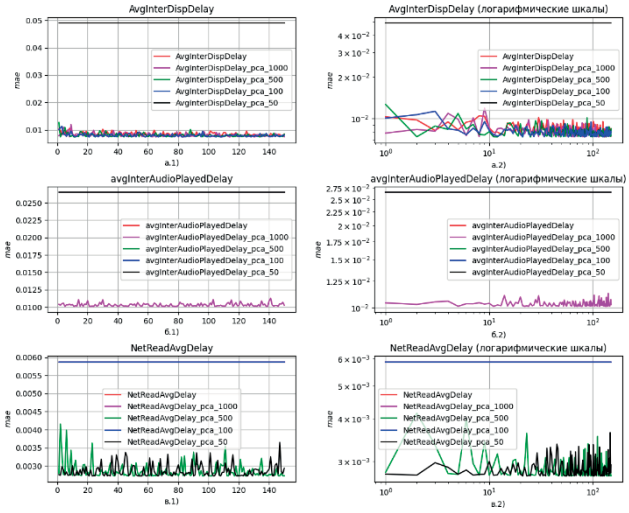

Рисунок 13. Графики точности при обучении CNN модели на метриках задачи VoD.

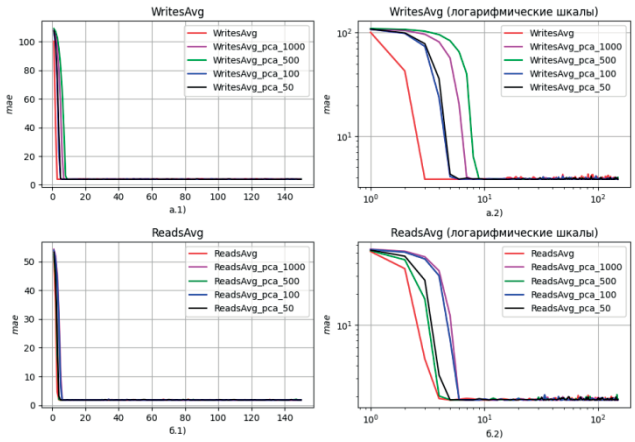

Рисунок 14. Графики точности при обучении MLP модели на метриках задачи KV.

	NetReadAvgDelay		avgInterAudioPlayedDelay		AvgInterDispDelay	
	min MAE	epoch	min MAE	epoch	min MAE	epoch
no PCA	0.0058	1	0.026	1	0.0076	86
PCA 1000	0.0058	1	0.01	77	0.0074	113
PCA 500	0.0027	101	0.026	1	0.0074	24
PCA 100	0.0058	1	0.026	1	0.0074	82
PCA 50	0.027	1	0.026	1	0.049	0

Таблица 5. Сводные значения точности для CNN на метриках задачи VoD

	WritesAvg		ReadsAvg	
	min MAE	epoch	min MAE	epoch
no PCA	3.84	45	1.85	69
PCA 1000	3.84	19	1.85	43
PCA 500	3.84	93	1.85	104
PCA 100	3.84	127	1.85	17
PCA 50	3.83	98	1.85	131

Таблица 6. Сводные значения точности для CNN на метриках задачи KV

На графиках 13.а.1, 13.б.1 и 13.в.1 видно, что некоторые модели не улучшают показатели точности с течением времени обучения. Это связано с тем, что само значение является достаточно малым – порядка 10^{-2}, 10^{-3}, и его улучшение невозможно при помощи данных моделей. Так же можно сделать вывод, что модели склонны к переобучению при решении задачи регрессии для метрик задачи VoD, это хорошо заметно на графиках 13.а.2 и 13.в.2.

В то же время на графиках 14.а.2 и 14.б.2 видно, что модель достаточно стабильна и не склонна к переобучению для задачи KV БД.

На основании полученных данных, было принято решение о выборе моделью для дальнейшего сравнительного анализа модель реализующую комбинацию CNN и PCA(500).

На Рис. 15 изображен график зависимости точность на проверочной выборке для задачи прогнозирования времени выполнения задачи на супер вычислителе. При этом лучшее значение точности, которого добилась сеть, составляет 14309 секунд. Заметим, что полученная

точность хуже, чем у MLP. Это может быть связано с тем, что при обучении был недостаточно большой набор обучающих данных.

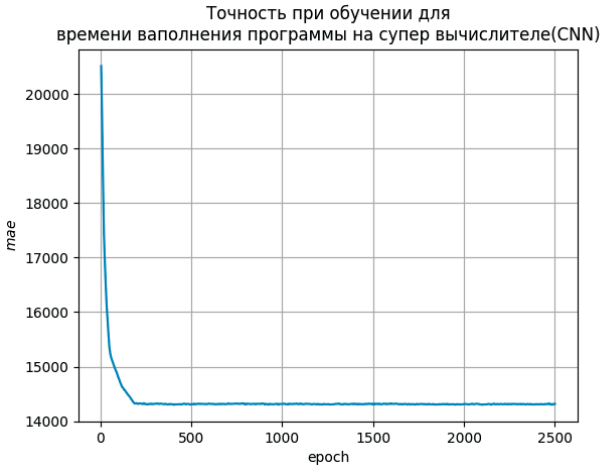

Рисунок 15. Графики точности при обучении CNN модели для задачи прогнозирования времени выполнения программы на супер вычислителе.

3.2.3 Сравнительный анализ

В данном разделе представлено сравнение двух методов.

Точность и время на обучение

Как было продемонстрированно на графиках и в таблицах обе модели показывают схожие результаты, но CNN модель достигает своих наилучших показателей точности быстрее чем MLP.

Склонность к переобучению

Модель MLP в данной задаче показала себя как модель не склонную к переобучению, в то время как при обучении CNN модели видны сильные флуктуации при обучении на метриках VoD, что свидетельствует о склонности модели к переобучению.

Стабильность

В таблицах 7, 8 отображены показатели метрики стабильности выбранных комбинаций методов. Из них можно сделать вывод, что MLP сеть более стабильная чем CNN.

	NetReadAvgDelay	avgInterAudioPlayedDelay	AvgInterDisp Delay
MLP PCA(100)	0.54	0.54	0.55
CNN PCA(500)	0.51	1.0	0.52

Таблица 7. Значения стабильности на задаче VoD для выбранных методов

	WritesAvg	ReadsAvg
MLP PCA(100)	0.49	0.54
CNN PCA(500)	0.5	0.51

Таблица 8. Значения стабильности на задаче KV для выбранных методов

ЗАКЛЮЧЕНИЕ

Для прогнозирования характеристик модельной задачи:

- наилучшим образом подходит модель случайного леса в сочетании с методами понижения размерности пространства признаков, а именно: анализ главных компонент
- при использовании нейронных сетей лучшие показатели, на 1-2 порядка, дали свёрточные нейронные сети (CNN) в комбинации с методом анализа главных компонент (PCA) при с количеством компонент равням 500.

При применении методов для прогнозирования временных характеристик прикладных сетевых сервисов на высокопроизводительном вычислителе точность предсказания - 4 часа, что в значительной мере зависит от предоставленного набора данных, а также требует дальнейшего расширения применяемых методов за счет техник матричного разложения для восстановления пропущенных данных и понижения размерности задачи:

- TT-toolbox на языке MATLAB [9]
- неотрицательное матричное разложение и другие…

ЛИТЕРАТУРА

[1] Slurm Support and Development|SHEDMD.The SLURM company [Электронный ресурс] - URL: https://www.schedmd.com (дата обращения: 11.04.2024)

[2] Data traces from a data center testbed – Kaggle. [Электронный ресурс] - URL: www.kaggle.com/datasets/jaliltaghia/data-traces-from-a-data-center-testbed (дата обращения: 11.09.2023)

[3] Federated learning for performance prediction in multi operator environments." Lan X., Taghia J., Moradi F., Khoshkholghi M.A., Listo Zec E., Mogren O., Mahmoodi T., Johnsson A. [Электронный ресурс] URL: doi.org/10.1016/j.future.2017.09.049 (дата обращения: 08.11.2023)

[4] A service-agnostic method for predicting service metrics in real time. Yanggratoke R., Ahmed J., Ardelius J., Flinta C., Johnsson A., Gillblad D., Stadler R. – International Journal of Network Management / September 2017

[5] Predicting service metrics for cluster-based services using real-time analytics. R. Yanggratoke, J. Ahmed, J. Ardelius, Ch. Flinta, A. Johnsson, D. Gillblad, R. Stadler - 11th International Conference on Network and Service Management / 2015

[6] Bishop C.M. Pattern Recognition and Machine Learning - Springer Science+Business Media, LLC / 2006 (дата обращения: 13.02.2024)

[7] Сайфутдинов Р. А. Исследование алгоритмов уменьшения размерности данных для задачи классификации» - СПбГУ / 2014

[8] https://github.com/Lycheva02/bachelory (дата обращения: 20.05.2024)

[9] https://github.com/oseledets/TT-toolbox (дата обращения: 20.05.2024)

Задача распределения программы обработки сетевых пакетов на стадии конвейера сетевого процессорного устройства

Никифоров Никита Игоревич
МГУ имени М.В. Ломоносова, факультет ВМК
Москва, Россия
rav263@asvk.cs.msu.ru

Волканов Дмитрий Юрьевич
МГУ имени М.В. Ломоносова, факультет ВМК
Москва, Россия
volkanov@asvk.cs.msu.ru

***Аннотация*—В работе представлена постановка задачи распределения программы обработки сетевых конвейеров на стадии СПУ с соблюдением ограничений по времени обработки и объёма передаваемого контекста и минимизацией объёма памяти занимаемого на устройстве.**

Для проверки оптимальности решения был разработана имитационная модель конвейера СПУ, в том числе вычислительных блоков, с использованием эмулятора SPIKE. Разработанная имитационная модель позволяет получать время обработки сетевых пакетов для заданного разбиения программы, а также статистику по использованной памяти вычислительных блоков и вызовам различных инструкций.

***Ключевые слова*—Сетевое процессорное устройство, классификация пакетов, архитектура RISC-V, организация конвейера.**

Введение

Сетевое процессорное устройство (СПУ) — это программируемый процессор, архитектура которого специально разработана для сетевых устройств и обеспечивает надёжную обработку пакетов.

К высокопроизводительным сетевым устройствам предъявляются разнообразные требования [1], среди которых можно выделить два ключевых: поддержка множества протоколов и высокая пропускная способность. Современные СПУ способны передавать данные со скоростью до 52 Тбит/с. Процесс обработки сетевых пакетов на СПУ включает несколько основных этапов:

1) Получение пакета с входного порта.
2) Анализ заголовка пакета и сохранение тела пакета в памяти.
3) Поиск правил обработки сетевого пакета в таблицах классификации.
4) Изменение заголовка сетевого пакета в соответствии с найденными правилами.
5) Объединение изменённого заголовка и тела сетевого пакета.
6) Отправка пакета на блок управления очередями.
7) Отправка пакета на выходной порт в соответствии с найденным правилом.

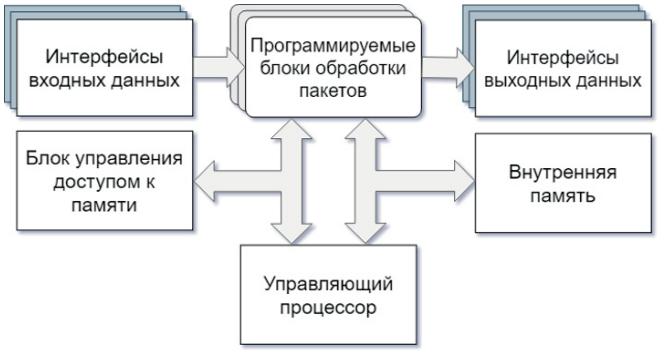

Рис. 1: Обобщённая архитектура СПУ.

В работе рассматривается конвейерная архитектура СПУ, представляющая собой архитектуру, в которой для обработки сетевых пакетов используется один или несколько конвейеров. Каждый конвейер состоит из последовательно расположенных N вычислительных блоков. Вычислительный блок может включать несколько вычислительных ядер RISC–V и память для хранения программы. Каждый вычислительный блок обрабатывает сетевой пакет в соответствии с программой обработки сетевых пакетов, загруженной на него. Обработка занимает ограниченное количество тактов.

В этой работе мы рассматриваем задачу распределения программы обработки сетевых пакетов по вычислительным блокам конвейера СПУ. Текущее решение, которое предполагает полное дублирование программы обработки на каждом вычислительном блоке СПУ, неэффективно с точки зрения использования памяти. Структура работы выглядит следующим образом:

В первом разделе приведено общее описание архитектуры СПУ. Во втором разделе описана неформальная постановка задачу оптимизации распределения программы обработки заголовков сетевых пакетов. В третьем разделе формализуется постановка задачи оптимизации распределения программы как задачу оптимизации перекрытий. В четвёртом разделе рас-

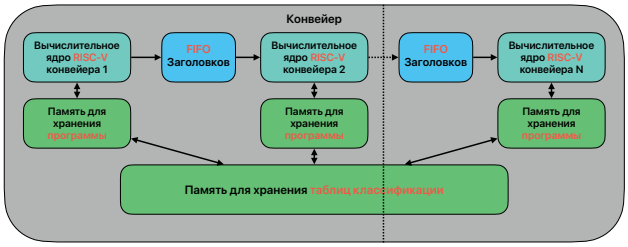

Рис. 2: Архитектура конвейера СПУ.

сматривается процесс имитационного моделирования работы СПУ, включая использование эмулятора Spike для вычислительных блоков конвейера СПУ.

I. Архитектура СПУ

A. Общее описание архитектуры СПУ

В общем случае устройство обработки сетевых пакетов [1] [2] состоит из 5 основных блоков:

1) интерфейсные модули приёма входных и передачи выходных данных с опциональной буферизацией;
2) программируемые блоки обработки пакетов, которые могут быть реализованы в виде конвейера;
3) внутренняя память;
4) блок управления доступом к памяти;
5) управляющий процессор или интерфейс взаимодействия с ним.

Обобщённая архитектура СПУ представлена на рисунке 1.

B. Архитектура конвейера СПУ

В рассматриваемой архитектуре СПУ используется конвейер для обработки заголовков сетевых пакетов. Конвейер состоит из N вычислительных блоков, памяти для хранения таблиц классификации, а также очередей FIFO между вычислительными блоками. Вычислительный блок —это совокупность вычислительных ядер RISC-V и памяти заданного размера M для хранения программы. Число заголовков сетевых пакетов в каждой очереди FIFO не должно превышать $FS_{\max}$. Архитектура конвейера СПУ представлена на рисунке 2.

Обработка заголовка сетевого пакета проходит последовательно, на каждом вычислительном блоке не меньше чем $T_{\min}$ тактов и не более чем $T_{\max}$, тактов. Порядок заголовков в процессе обработки меняться не может.

C. Архитектура RISC-V

Архитектура RISC-V [3] представляет собой открытую и свободную систему команд и процессорную архитектуру, основанную на концепции RISC (Reduced Instruction Set Computing). Архитектура RISC-V была разработана в 2010 году в Калифорнийском университете Беркли и предназначена для создания процессоров и микроконтроллеров.

Архитектура RISC-V обладает следующими преимуществами:

1) открытая и свободная архитектура;
2) гибкая и масштабируемая;
3) расширяемая;
4) более простая архитектура по сравнению с традиционными x86, arm и т.п. [4];
5) низкое энергопотребление.

В архитектуре RISC-V можно привести классификацию инструкций по двум признакам. Первые по типу использования:

- Арифметические инструкции: add, sub, mul, div;
- Логические инструкции: and, or, xor, sll;
- Инструкции работе с данными: lw, sw, lb, sb;
- Инструкции по передаче управления: jal, jr, beq, bne;
- Инструкции сравнения: $slt, sltu$.

И по типу доступа к данным, с которыми взаимодействует инструкция:

- Register (R-type) инструкции для работы с регистрами,
- Immediate (I-type) инструкции для работы с константами,
- Store (S-type) инструкции для работы с памятью,
- Branch (B-type) инструкции условного перехода,
- Upper immediate (U-type), инструкции для работы с длинными константами,
- Jump (J-type) инструкции безусловного перехода.

Для каждого типа инструкции первые 6 бит занимает код инструкции.

Определим инструкцию $inst_i$ — 32-битное число и программу на языке ассемблера RISC-V, как ограниченное упорядоченное множество инструкций: $P = \{inst_1, inst_2, \ldots, inst_n\}$. Отдельно рассмотрим инструкции B-type и J-type, которые осуществляют условную и безусловную передачу управления. Инструкции по передаче управления позволяют перейти по другому адресу программы:

$$P = \{inst_1, \ldots, inst_{i-1}, inst_i, inst_{i+1}, \ldots, inst_n\}. \quad (1)$$

Пусть $inst_j \to inst_i, \forall j \in [1, n]$, инструкция по передаче управления. Тогда при выполнении $inst_j$ управление переходит на инструкцию $inst_i$.

D. Сценарии работы СПУ

При использовании СПУ в сети для маршрутизации трафика необходимо, чтобы СПУ работал со следующим набором сценариев [2] обработки пакетов: L2 уровня (L2 switch, L2 switch + VLAN, L2 switch + VLAN + L3-unicast, L2 switch + VLAN + L3-multicast), IP/MPLS (LAG (Link Aggregation Group), LLDP/LACP/QoS, Mirror, MPLS-ingress, MPLS-VPN-ingress, MPLS-P,

MPLS-egress, MPLS-VPN-egress), PTP, ACL, GRE-ingress, GRE-egress.

E. Программа обработки заголовков сетевых пакетов

Для обработки заголовков сетевых пакетов используется программа обработки сетевых пакетов на языке ассемблера RISC-V. Программа обработки заголовков сетевых пакетов реализует различные сценарии работы СПУ и от реализации и корректности программы зависит скорость работы СПУ.

Также распределение программы на вычислительные блоки конвейера СПУ влияет на загрузку (количество заголовков) в очередях FIFO между вычислительными блоками и общее время обработки потока сетевых пакетов. Также уменьшение объёма памяти, занимаемого программой обработки, позволяет в одной программе реализовать большее количество сценариев работы СПУ.

При передаче управления с одного вычислительного блока на следующий по конвейеру СПУ. Вместе с заголовком сетевого пакета передаётся контекст выполнения программы обработки заголовков сетевых пакетов. В данном контексте хранится номер инструкции, с которой необходимо продолжить выполнение программы. Также в контексте передаются значения некоторых регистров, необходимые для продолжения обработки заголовка.

II. Неформальная постановка задачи

В данной работе рассматривается задача распределения программы обработки сетевых пакетов на вычислительные блоки.

Дано: Конвейер, который состоит из входной очереди, N вычислительных устройств, FIFO очереди заголовков сетевых пакетов между каждыми двумя последовательными вычислительными устройствами и выходной очереди. Также дана программа обработки заголовков сетевых пакетов на языке ассемблера RISC-V, представляющая собой упорядоченный список инструкций ассемблера.

Рассматриваемые в задаче ограничения:

1) размер каждой очереди FIFO заголовков пакетов между вычислительными блоками не должен превышать $FS_{\max}$,
2) программа обработки заголовков сетевых пакетов не имеет циклов,
3) память, используемая для программы на каждом вычислительном блоке, не должна превышать $M_{\max}$,
4) общее время выполнения программы не должно превышать
   $N * T_{\max}$ и не должно быть меньше чем $N * T_{\min}$.

Требования: Решение поставленной задачи должно позволить снизить общий объём памяти для хранения программ обработки заголовков сетевых пакетов в два раза, чтобы сократить объём памяти на кристалле,

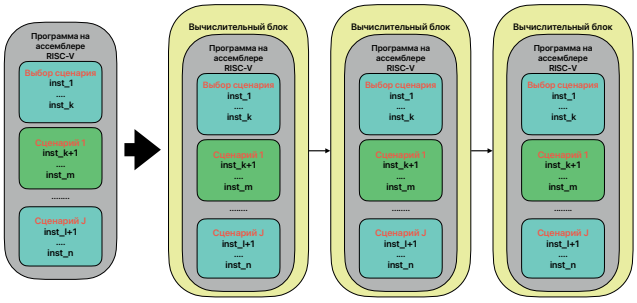

Рис. 3: Распределение программы на конвейере СПУ.

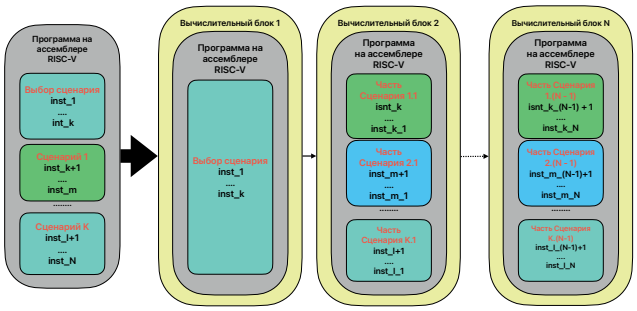

Рис. 4: Распределение программы на конвейере СПУ.

для удешевления производства. Т.е. $M_{\max} = \frac{M_{cur}}{2}$, где M_{cur} — текущий объём доступной памяти для хранения программы на каждом вычислительном блоке.

Необходимо: Разработать метод разбиения программы на языке ассемблера RISC-V на подпрограммы, каждая из которых выполняется своим вычислительным блоком с соблюдением всех ограничений, и с минимизацией суммарного объёма памяти для хранения программы, занимаемого на вычислительных блоках.

Текущая реализация в рассматриваемой архитектуре СПУ подразумевает полное дублирование программы обработки заголовков сетевых пакетов на вычислительные блоки конвейера [2]. Такое решение не позволяет соблюдать заданные ограничения. Текущий вариант распределения программы схематично изображён на рисунке 3.

Схожие задачи рассматривались в литературе, например в работах [5] [6] [7]. Но в них не решалась задача распределения программы на конвейер вычислительных блоков, где каждый вычислительный блок представляет собой независимое вычислительное устройство, на котором может выполняется не одна инструкция ассемблера (или часть инструкции), а подпрограмма обработки заголовков сетевых пакетов.

III. Формальная постановка задачи

Одним из методов оптимизации суммарного объёма памяти, занимаемого программой обработки сетевых

пакетов на вычислительных блоках конвейера СПУ, является минимизация перекрытия. Рассмотрим формальную постановку задачи распределения программы обработки сетевых пакетов на вычислительные блоки конвейера СПУ, как задачу минимизации перекрытий. Схожие задачи рассматривались в работах [8] [9].

Определим перекрытие: пусть имеется программа обработки сетевых пакетов $P = \{inst_1, inst_2, \ldots, inst_n\}$, пусть есть некоторое разбиение $D = \{P_1, P_2, \ldots, P_N\}$, где $P_i, i \in (1, N), N \in \mathbb{N}$ — некоторая подпрограмма исходной программы P. Подпрограмма $P_i = \{inst_1^i, inst_2^i, \ldots, inst_{n_i}^i\}$ — некоторая упорядоченная последовательность инструкций $inst_j^i, j \in (0, n_i)$. При этом подпрограмма $P_i \subseteq P$. Перекрытием O_i подпрограмм P_i, P_{i+1}, будем называть совпадающее множество инструкций $O_i = \{inst_{n_i-k-1}^i, inst_{n_i-k-2}^i, \ldots, inst_{n_i}^i\} = \{inst_1^{i+1}, inst_2^{i+1}, \ldots, inst_k^{i+1}\}$, при этом: $inst_{n_i-j}^i = inst_{k-j}^{i+1}, \forall j \in (1, k)$.

Тогда задача состоит в построении такого разбиения D, $\min_D(\sum_{i=1}^{N-1} size(O_i))$. С учётом ограничений, рассматриваемых в задаче: при работе программы размер каждой из очередей FIFO между вычислительными блоками F заголовков пакетов не должен быть превышен, программа обработки заголовков сетевых пакетов не имеет циклов, память для хранения программы, используемая на каждом вычислительном блоке не должна превышать $M_{\max}$, а также общее время выполнения программы не должно превышать $T_{\max}$.

В текущей реализации, программа P полностью дублируется на все вычислительные блоки, то есть разбиение $D_0 = \{P_1, P_2, \ldots P_N\}, \forall i \in (1, N), P_i \equiv P$. Посчитаем минимизируемый критерий для такого разбиения:

$$\sum_{i=1}^{N-1} size(O_i) = \sum_{i=1}^{N-1} size(P_i) = size(P) * (N-1) \quad (2)$$

Чтобы снизить значение критерия, можно предложить варианты распределения программы, в которых отсутствует полное дублирование, но при этом увеличивается среднее время обработки сетевого пакета на конвейере СПУ из-за неравномерности распределения программы. Также размер передаваемого контекста с заголовками сетевых пакетов (рис. 4).

IV. Имитационная модель СПУ

Для проверки соблюдения ограничений заданных в задаче разрабатывается имитационная модель СПУ на основе эмулятора RISC-V Spike [10], который позволяет эмулировать работу ядер с архитектурой RISC-V RV32IMA. Так как при работе СПУ важно учитывать не отдельные сетевые пакеты, а поток сетевых пакетов — с помощью работы имитационной модели СПУ будет проверятся влияние времени обработки одних сетевых пакетов на время обработки других сетевых

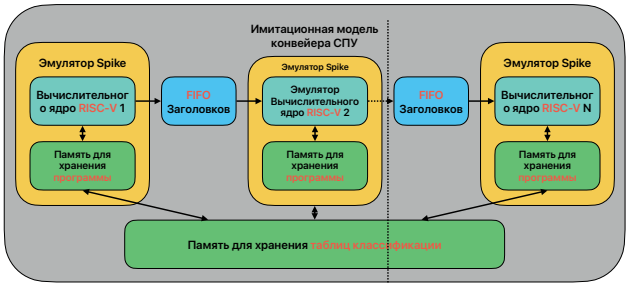

Рис. 5: Архитектура имитационной модели СПУ

пакетов, которое возникает из-за использования конвейерной архитектуры СПУ.

Основная задача при создании имитационной модели СПУ состоит в передаче данных между вычислительными блоками конвейера. Само же моделирование обработки сетевых пакетов будет проводится на эмуляторе Spike. Эмулятор Spike будет выступать модулем ядра вычислительного блока, который будет частью имитационной модели других систем СПУ на SystemC [11].

Предлагаемая архитектура имитационной модели представлена на рисунке 5.

A. SystemC

SystemC является библиотекой, созданной на языке C++ и для использования с языком программирования C++. SystemC позволяет моделировать как аппаратные системы, так и модели программных компонентов. Основным достоинством библиотеки является огромный инструментарий для моделирования интегральных микросхем и систем на кристалле.

Среди особенностей SystemC можно отметить:

- Возможность моделирования систем на различных уровнях абстракции, что позволяет моделировать систему начиная от детального описания каждого компонента и заканчивая высокоуровневым описанием процессов.
- Поддержка параллельного моделирования компонентов системы.
- SystemC является частью стандартов IEEE (IEEE 1666), что обеспечивает его широкую поддержку и совместимость в индустрии.

B. Spike

Эмулятор Spike — это функциональный симулятор архитектуры RISC-V, который реализует модель одного или нескольких процессоров (hart). Он служит золотым стандартом для функционального моделирования и предоставляет возможность полной системной эмуляции или проксированной эмуляции с использованием HT IF/FESVR.

Среди особенностей эмулятора Spike можно отметить:

- Поддерживается различные расширения RISC-V, такие как RV32IMAFDQCV.
- Поддерживается несколько моделей памяти, включая слабую и полную упорядоченность записи (WMO и TSO).
- Эмулятор Spike позволяет добавлять и тестировать новые инструкции.
- Также поддерживается добавление новых модулей эмулятора.

V. Заключение

В данной статье приведена постановка задачи распределения программы по вычислительным блокам в конвейере СПУ и описан подход к построению имитационной модели СПУ. Существующие методы распределения программы имеют ряд существенных недостатков: Программа полностью копируется на каждый вычислительный блок, что требует значительных ресурсов. Между вычислительными блоками передается большой контекст сетевого пакета, что увеличивает объём FIFO и накладные расходы.

В качестве будущих задач можно выделить: обзор схожих подходов к оптимизации, разработка метода распределения программы по вычислительным блокам конвейера СПУ, создание имитационной модели СПУ с использованием эмулятора RISC-V Spike для тестирования и оценки эффективности нашего метода.

Список литературы

[1] *Orphanoudakis T.*, *Perissakis S.* Embedded multi-core processing for networking // Multi-Core Embedded Systems. — CRC Press, 2018. — P. 429–494.

[2] Об одном подходе к построению сетевого процессорного устройства / С. О. Беззубцев, В. В. Васин, Д. Ю. Волканов, Ш. Р. Жайлауова, В. А. Мирошник, Ю. А. Скобцова, Р. Л. Смелянский // Моделирование и анализ информационных систем. — 2019. — Т. 26, № 1. — С. 39—62.

[3] RISC-V International; RISC-V: The Open Standard RISC Instruction Set Architecture. — [Accessed 01-10-2024]. https://riscv.org.

[4] *Frolov V. A.*, *Galaktionov V. A.*, *Sanzharov V. V.* Investigation of RISC-V // Programming and Computer Software. — 2021. — Vol. 47. — P. 493–504.

[5] *Beaty S. J.* Instruction scheduling using genetic algorithms : PhD thesis / Beaty Steven John. — Colorado State University, 1991.

[6] *Loehr D.*, *Walker D.* Safe, modular packet pipeline programming // Proceedings of the ACM on Programming Languages. — 2022. — Vol. 6, POPL. — P. 1–28.

[7] *Ertl M. A.*, *Krall A.* Instruction scheduling for complex pipelines // Compiler Construction: 4th International Conference, CC'92 Paderborn, FRG, October 5–7, 1992 Proceedings 4. — Springer. 1992. — P. 207–218.

[8] Optimizing overlapped memory accesses in user-directed vectorization / D. Caballero, S. Royuela, R. Ferrer, A. Duran, X. Martorell // Proceedings of the 29th ACM on International Conference on Supercomputing. — 2015. — P. 393–404.

[9] *Dehghan R.*, *Hazini K.*, *Ruwanpura J.* Optimization of overlapping activities in the design phase of construction projects // Automation in Construction. — 2015. — Vol. 59. — P. 81–95.

[10] GitHub - riscv-software-src/riscv-isa-sim: Spike, a RISC-V ISA Simulator — github.com. — [Последнее обращение 2024-10-01]. https://github.com/riscv-software-src/riscv-isa-sim.

[11] systemc.org — systemc.org. — [Последнее обращение 2024-10-01]. https://systemc.org.

Прогнозирование числовых значений показателей качества обслуживания гетерогенного канала при помощи авторегрессионных моделей

Пуртов Б.С
Факультет вычислительной математики и кибернетики Московский государственный университет имени М.В.Ломоносова
Москва, Россия
bog_29@mail.ru

Волканов Д.Ю.
Факультет вычислительной математики и кибернетики Московский государственный университет имени М.В.Ломоносова
Москва, Россия
volkanov@asvk.cs.msu.ru

Аннотация—Данная работа посвящена разработке метода прогнозирования числовых значений показателей качества обслуживания гетерогенных каналов связи, используя авторегрессионные модели. Исследование направлено на анализ временных рядов значений показателей качества обслуживания (КО). В работе рассматривается задача прогнозирования значений показателей КО на основе имеющихся временных рядов значений пропускной способности, джиттера и задержки.

Ключевые слова—прогнозирование, авторегрессия, показатели качества обслуживания, модель ARIMAX, компьютерные сети, сетевой трафик.

I. Введение

В эпоху цифровизации и широкого распространения информационно-коммуникационных технологий качество обслуживания (КО) становится критически важным аспектом для надёжной и эффективной работы сетевых инфраструктур. КО – это технология, которая присваивает разным классам трафика разные приоритеты, оптимизируя производительность сети и удовлетворяя требования различных приложений и услуг [1].

В работе рассматриваются три показателя КО, определяющие эффективность гетерогенных каналов передачи данных:

- пропускная способность (throughput);
- задержка при передаче пакета (delay);
- джиттер (jitter).

Пропускная способность (throughput) – это объём данных, который может быть передан через сеть за определённое время. Задержка при передаче пакета (delay) – это время, необходимое для прохождения пакета от источника к получателю, что критично для приложений, чувствительных к времени отклика, таких как видеоконференции и онлайн-игры. Джиттер (jitter) – это вариация задержек пакетов в одном потоке, что может ухудшить качество обслуживания, особенно для голосовых и видеосервисов.

Исследования [2], [3] показывают, что трафик в компьютерных сетях обладает свойством самоподобия, то есть сохраняет свою структуру при масштабировании, что позволяет точно прогнозировать его будущее поведение на основе анализа прошлых данных. Цель данного исследования – создать метод краткосрочного прогнозирования показателей КО с использованием авторегрессионных моделей [4], которые предсказывают будущие значения сетевых характеристик, опираясь на исторические данные о сетевом трафике.

Статья имеет следующую структуру: в главе II формулируется постановка задачи. В главе III представлен обзор различных авторегрессионных моделей с выбором наиболее подходящей для решения поставленной задачи. В главе IV описан алгоритм формирования авторегрессионной модели и предложенный метод решения. В главе V приводится описание и результаты экспериментального исследования.

II. Постановка задачи

Дано:

Совокупность временных рядов значений показателей КО, полученных с трафика внутренней сети за ограниченный период времени $[0; T]$:

- пропускная способность
- джиттер
- задержка

Необходимо:

Выполнить краткосрочное прогнозирование значений данных показателей на моменты времени $(T; T*]$.

III. Обзор авторегрессионных моделей

A. Критерии обзора

Для выбора наиболее подходящей модели были выбраны следующие критерии обзора:

- Учет нестационарности
- Возможность использования дополнительных данных
- Вычислительная сложность
- Предсказание данных с краткосрочной зависимостью (SRD)

В работе рассматриваются данные, собранные в сети кафедры АСВК за ограниченный период времени в 250 секунд, соответственно, временные ряды показателей КО, используемые для обучения модели, ограничены 250 записями. Это ограничение затрудняет выявление долгосрочных и сезонных зависимостей. В то же время, учет нестационарности и интеграция дополнительных данных становятся ключевыми факторами при выборе авторегрессионной модели. Это объясняется тем, что не все предоставленные временные ряды стационарны, а так же некоторые из них могут быть использованы для повышения точности прогнозов других рядов.

B. Обзор моделей

В данной работе рассматриваются следующие модели авторегрессии:

- VAR - Vector Autoregression [5],
- ARMA - Autoregressive Moving Average [6],
- ARIMA - Autoregressive Integrated Moving Average [7],
- SARIMA - Seasonal Autoregressive Integrated Moving Average [8],
- ARIMAX - Autoregressive Integrated Moving Average extended [9],
- SARIMAX - Seasonal Autoregressive Integrated Moving Average extended [9],
- FARIMA - Fractional Autoregressive Integrated Moving Average [10].

Эти модели были выбраны для рассмотрения в данной работе из-за их широкого распространения и проверенной эффективности в анализе и прогнозировании временных рядов в разнообразных областях.

1) VAR (Векторная авторегрессионная модель): Модель VAR используется для прогнозирования взаимосвязанных временных рядов, учитывая взаимное влияние временных рядов друг на друга [11]. VAR предполагает стационарность временных рядов, и если ряд нестационарен, требуется его предварительная обработка. Модель VAR не включает компоненты скользящего среднего, что делает её менее гибкой по сравнению с другими моделями.

2) ARMA (Авторегрессионная модель со скользящим средним): ARMA объединяет авторегрессионную компоненту (AR) и компоненту скользящего среднего (MA), но не поддерживает нестационарные ряды, сезонность и экзогенные данные. На практике ARMA часто модифицируется и улучшается для повышения точности прогноза. Например, внедрение метода наименьших квадратов для динамического вычисления параметров модели показывает улучшенные результаты на синтетических данных [5].

3) ARIMA (Авторегрессионная интегрированная модель со скользящим средним): Модель ARIMA способна дифференцировать данные для достижения стационарности [7]. Она проста в использовании и широко применяется для прогнозирования сетевого трафика и других областей, таких как гидрологические процессы и управление водными ресурсами [12], а также в медицинской сфере для предсказания распространения инфекционных заболеваний [13].

4) SARIMA (Сезонная авторегрессионная интегрированная модель со скользящим средним): SARIMA расширяет ARIMA, добавляя сезонные компоненты, что делает её подходящей для данных с сезонными колебаниями [8]. Однако для эффективного использования модели SARIMA требуется большее количество данных и правильное определение сезонных периодов.

5) ARIMAX и SARIMAX: Данные модели являются улучшениями моделей ARIMA и SARIMA соответственно, включающие экзогенные переменные. Это позволяет учитывать внешние факторы, влияющие на временной ряд. Исследования показывают, что ARIMAX и SARIMAX могут демонстрировать улучшенную производительность по сравнению со своими базовыми моделями [9], [14].

6) FARIMA (Фракционная авторегрессионная интегрированная модель со скользящим средним): Модель FARIMA отличается тем, что степень интегрирования может быть дробной, что позволяет лучше моделировать процессы с долговременной зависимостью. Исследования сетевого трафика показывают лучшие результаты при прогнозировании с помощью FARIMA по сравнению с ARIMA [10], особенно при наличии долгосрочных корреляций. Хотя также данная модель смогла показать себя лучше своей базовой версии и при прогнозировании сетевого трафика [18], где не было долгосрочной корреляции между наблюдениями

C. Методы оценки предсказанных значений

Наиболее используемыми метриками оценки прогнозов являются MAE, MSE, MARE, MAPE, NMSE и RMSE. Для сетевых характеристик с широким диапазоном значений формула оценки должна обладать свойством инвариантности к масштабу для обеспечения лучшей интерпретации результатов. MAPE и NMSE обладают этим свойством, однако NMSE усложняет интерпретацию из-за усиления влияния выбросов. Поэтому для оценки спрогнозированных значений была выбрана метрика MAPE, представляемая формулой (1).

$$MAPE = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{y_i - \hat{y}_i}{y_i} \right|. \quad (1)$$

где:

- y_i — истинные значения,
- $\hat{y}_i$ — предсказанные значения,
- $\bar{y}$ — среднее значение истинных значений y,
- n — количество наблюдений.

D. Выводы

Результаты обзора приведены в таблице I.

По итогам обзора была выбрана метрика MAPE для оценки прогнозов и авторегрессионная модель ARIMAX, так как она позволяет работать с нестационарными временными рядами, использовать экзогенные переменные для прогнозирования и имеет меньшую вычислительную сложность, по сравнению с другими небазовыми моделями.

IV. Построение решения задачи

A. Формирование модели

Решение задачи прогнозирования значений показателей КО включает создание трех авторегрессионных моделей для предсказания значений каждой характеристики (пропускная способность, джиттер, задержка). При формировании моделей используется методология Бокса-Дженкинса для подбора ARIMA-модели, состоящая из следующих этапов: идентификация модели, оценка параметров модели и тестирование модели [15].

1) Идентификация модели: На данном этапе определяется стационарность временного ряда с помощью теста Дикки-Фуллера и определяется значения параметра дифференцирования d. Далее определяются порядки для компонентов авторегрессии p и скользящего среднего q с помощью автокорреляционной (ACF) и частично автокорреляционной (PACF) функций [16].

2) Оценка коэффициентов модели: Применение Метода максимального правдоподобия или Метода наименьших квадратов для определения значений коэффициентов модели.

3) Тестирование модели: Проверка модели включает сравнение предсказанных значений с тестовой выборкой и анализ остатков модели. Используются тесты на нормальность распределения остатков (тест Харке-Бера), случайность остатков (тест Льюинга-Бокса) и отсутствие автокорреляции [17].

B. Решение задачи прогнозирования

Предлагаемый метод решения задачи предсказания значений показателей КО заключается в создании авторегрессионных моделей ARIMAX для каждого показателя (пропускная способность, джиттер, задержка), используя в качестве экзогенных данных значения других показателей.

Алгоритм решения задачи:

1) Выбор порядка предсказываемых показателей:
   - Пропускная способность выбирается для первичного прогнозирования из-за её большей однородности и меньшей зависимости от выборочных сетевых пакетов.
   - Джиттер выбирается вторым показателем из-за его большей однородности по сравнению с задержками сетевого канала.
   - Задержка выбирается в последнюю очередь. Для повышения точности прогнозирования необходимо больше экзогенных данных.
2) Создание модели ARIMAX для пропускной способности:
   - Определение параметров модели (p, d, q) для пропускной способности.
   - Экзогенные данные не используются.
3) Создание модели ARIMAX для джиттера:
   - Определение параметров модели (p, d, q) для джиттера.
   - В качестве экзогенных данных используется временной ряд пропускной способности.
4) Создание модели ARIMAX для задержки:
   - Определение параметров модели (p, d, q) для задержки.
   - В качестве экзогенных данных используются временные ряды пропускной способности и джиттера.

В результате были получены 3 обученные авторегрессионные модели - ARIMAX(9,1,11), ARIMAX(21,0,24) и ARIMAX(22,0,22) для прогнозирования значений пропускной способности, джиттера и задержки соответственно.

V. Экспериментальное исследование

A. Цель исследования

Целью экспериментального исследования является сравнение авторегрессионных моделей, реализованных по предложенному методу, с моделями ARIMA, которые предсказывают значения показателей КО без использования дополнительных данных. В исследовании проводится сравнение времени, необходимого для обучения моделей, а также точности прогнозирования, измеряемой по ошибкам прогнозов.

B. Методика исследования

Модели ARIMA были сформированы аналогично моделям ARIMAX в соответствии с описанным выше методом формирования моделей. Так как идентификация моделей ARIMAX не зависит от переданных экзогенных данных, параметры p, d и q для модели ARIMA получились равными соответствующим параметрам модели ARIMAX для каждого показателя КО. Модель ARIMAX для прогнозирования пропускной способности не использует экзогенных данных, поэтому ее результаты будут совпадать с результатами работы модели ARIMA для данного показателя и учитываться при сравнении методов не будут.

Критерий	ARMA	ARIMA	SARIMA	ARIMAX	SARIMAX	VAR	FARIMA
Учет нестационарности	-	+	+	+	+	+	+
Дополнительные данные	-	-	-	+	+	-	-
Вычислительная сложность	+	+	-/+	+/-	-	-	-/+
SRD	+	+	+	+	+	+	+

Таблица I: Обзор авторегрессионных моделей

1) Сравнение производительности моделей: Временные ряды, используемые в исследовании, состоят из 250 значений. Замер времени обучения для каждой модели производится по 10 раз, после чего результаты усредняются для получения более стабильной оценки производительности.

2) Сравнение результатов прогнозирования моделей: Сравнение обученных моделей производится на сетевом трафике, собранном внутри того же гетерогенного канала, на котором обучались модели, но на других временных промежутках.

Проведение экспериментального исследования включает следующие этапы:

- Разбиение временных рядов на тестовую часть, которая будет сравнивать с предсказанными значениями и часть, значения которой используются для прогнозирования (эндогенная). Количество эндогенных значений определяется максимальным значением параметров p и q для каждой модели.
- Предсказание значений показателя пропускной способности для его дальнейшего использования при прогнозировании других показателей моделью ARIMAX.
- Предсказание значений джиттера с помощью моделей ARIMAX и ARIMA и вычисление средней абсолютной процентной ошибки и сохранение спрогнозированных данных для использования их как экзогенных при предсказании значений задержки.
- Предсказание значений задержки с помощью моделей ARIMAX и ARIMA и вычисление средней абсолютной процентной ошибки.

C. Результаты

В таблице II представлены данные о среднем времени обучения моделей ARIMA и ARIMAX для показателей джиттера и задержки.

На графиках 1 и 2 представлено сравнение значений средних абсолютных ошибок в процентах полученных предсказаний для 10 временных рядов.

По результатам экспериментального исследования можно сделать вывод, что предложенный метод позволяет более точно прогнозировать значения джиттера и задержки, обеспечивая в среднем на 4% более точные прогнозы по сравнению с моделями ARIMA, используя метрику MAPE. Также экспериментальное исследование показало, что предложенный метод требует

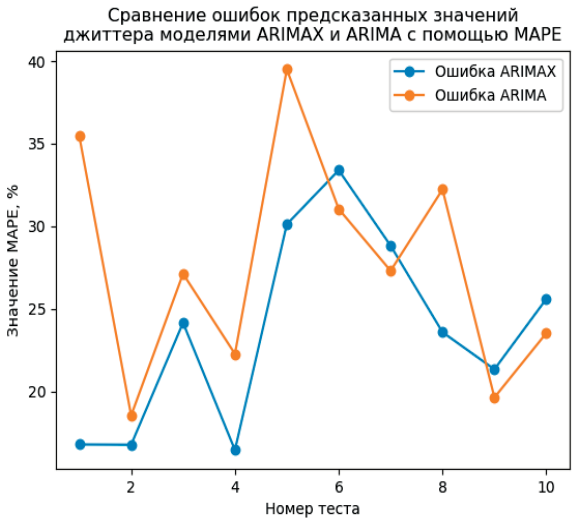

Рис. 1: Сравнение ошибок прогнозирования джиттера моделями ARIMAX и ARIMA

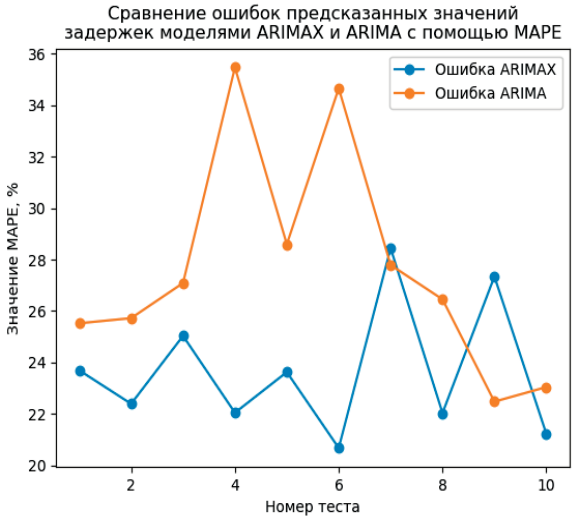

Рис. 2: Сравнение ошибок прогнозирования задержки моделями ARIMAX и ARIMA

	Пропускная способность	Джиттер ARIMA	Задержка ARIMA	Джиттер ARIMAX	Задержка ARIMAX
Время обучения, сек	3.7	10.1	10.7	13.5	12.5

Таблица II: Сравнение времени обучения моделей

большего времени для обучения модели из-за учета дополнительных временных рядов. Это не влияет на скорость прогнозирования после обучения, так как происходит вычисление новых значений на основе ограниченного множества определенных коэффициентов, то есть производительность обеих моделей остаётся сопоставимой. Но увеличенное время обучения может стать значительным недостатком при частом переобучении моделей.

VI. Заключение

В данной работе предложен метод прогнозирования значений показателей КО на основе выбранной авторегрессионной модели ARIMAX, заключающийся в последовательном предсказании значений временных рядов с использованием ранее предсказанных значений других показателей в качестве экзогенных данных. Также было проведено экпериментальное исследование которое показало, что предложенный метод с использованием экзогенных данных позволяет улучшить точность прогнозов по метрике MAPE для значений джиттера и задержки по сравнению с моделями ARIMA.

Список литературы

[1] Колотовкин И. С. Нейронные сети в задачах распределения трафика компьютерных сетей //Нейрокомпьютеры и их применение. – 2020. – С. 330-332.

[2] Киреева Н. В., Буранова М. А. Исследование самоподобного трафика с использованием пакета FRACTAN //T-Comm-Телекоммуникации и Транспорт. – 2012. – №. 5. – С. 50-52.

[3] Платов В. В., Петров В. В. Исследование самоподобной структуры телетрафика беспроводной сети //Радиотехнические тетради. – 2004. – №. 30. – С. 58-62.

[4] Jin X. et al. Kuiper test and autoregressive model-based approach for wireless sensor network fault diagnosis //Wireless Networks. – 2015. – Т. 21. – С. 829-839.

[5] Chandra S. R., Al-Deek H. Predictions of freeway traffic speeds and volumes using vector autoregressive models //Journal of Intelligent Transportation Systems. – 2009. – Т. 13. – №. 2. – С. 53-72.

[6] Lv J., Li T., Li X. Network traffic prediction algorithm and its practical application in real network //2007 IFIP International Conference on Network and Parallel Computing Workshops (NPC 2007). – IEEE, 2007. – С. 512-517.

[7] Yu Y. et al. Network traffic prediction and result analysis based on seasonal ARIMA and correlation coefficient //2010 International Conference on Intelligent System Design and Engineering Application. – IEEE, 2010. – Т. 1. – С. 980-983.

[8] Vujicic B., Chen H., Trajkovic L. Prediction of traffic in a public safety network //2006 IEEE International Symposium on Circuits and Systems (ISCAS). – IEEE, 2006. – С. 4 pp.

[9] Cools M., Moons E., Wets G. Investigating the variability in daily traffic counts through use of ARIMAX and SARIMAX models: assessing the effect of holidays on two site locations //Transportation research record. – 2009. – Т. 2136. – №. 1. – С. 57-66.

[10] Biernacki A. Improving quality of adaptive video by traffic prediction with (F) ARIMA models //Journal of communications and networks. – 2017. – Т. 19. – №. 5. – С. 521-530.

[11] Kim Y., Kim S. Electricity load and internet traffic forecasting using vector autoregressive models //Mathematics. – 2021. – Т. 9. – №. 18. – С. 2347.

[12] Valipour M., Banihabib M. E., Behbahani S. M. R. Comparison of the ARMA, ARIMA, and the autoregressive artificial neural network models in forecasting the monthly inflow of Dez dam reservoir //Journal of hydrology. – 2013. – Т. 476. – С. 433-441.

[13] Singh R. K. et al. Prediction of the COVID-19 pandemic for the top 15 affected countries: Advanced autoregressive integrated moving average (ARIMA) model //JMIR public health and surveillance. – 2020. – Т. 6. – №. 2. – С. e19115.

[14] Kongcharoen C., Kruangpradit T. Autoregressive integrated moving average with explanatory variable (ARIMAX) model for Thailand export //33rd International Symposium on Forecasting, South Korea. – 2013. – С. 1-8.

[15] Трегуб А. В., Трегуб И. В. Методика построения модели ARIMA для прогнозирования динамики временных рядов //Лесной вестник/Forestry bulletin. – 2011. – №. 5. – С. 179-183.

[16] Brockwell P. J., Davis R. A. (ed.). Introduction to time series and forecasting. – New York, NY : Springer New York, 2002.

[17] Артамонов Н. В. и др. Введение в анализ временных рядов.-Вологда: ВолНЦ РАН. – 2021.

[18] Колесников А. В. Моделирование сетевого трафика и алгоритмы борьбы с перегрузками на основе методов нелинейной динамики и краткосрочного прогнозирования временных рядов : дис. – Моск. гос. авиац. ин-т, 2015.

Время подготовки к работе сетевого сервиса при использовании разделяемого репозитария образов ВМ

В.О. Писковский
ВМК МГУ им. М.В. Ломоносова
Москва, Россия
vpiskovski@lvk.cs.msu.ru

В.Д Сагалевич
ВМК МГУ им. М.В. Ломоносова
Москва, Россия
vsagalev@gmail.com

Аннотация—Функционирование распределенных вычислительных систем в соответствии с предъявляемыми к ним техническими требованиями определяется эффективностью организации работы сетевого сервиса на оборудовании в сети, оборудования и платформы виртуализации. Пользователям сети необходимо предоставить набор информационных услуг с заданным уровнем качества и надёжности: - доступ к информации «в любое время в любом месте», то есть любому участнику процесса управления при наличии прав доступа при возникновении потребности и вне зависимости от места его нахождения; - информационное взаимодействие между автоматизированными системами; - своевременный мониторинг и анализ данных источников различного типа; - переход от централизованной схемы «загрузка с очисткой данных — анализ — распространение» к схеме «распределенные размещение и предобработка данных, и по необходимости — последующие загрузка, анализ и распространение». Одним из ключевых факторов обеспечения приведенных выше типов услуг с заданными вероятностно-временными характеристиками является время старта виртуальной машины на широком спектре оборудования и платформ виртуализации. Рабочая инфраструктура облачной вычислительной структуры может содержать репозитарий контейнеров с виртуальными машинами (ВМ), реализующими сетевые приложения и услуги, а равно и рабочие места сотрудников. Предполагается, что хранимые таким образом ВМ прошли принятую организаторами или владельцами вычислительной среды процедуру внедрения в постоянную эксплуатацию. Инфми словами, репозитарий эталонных ВМ содержит эталонные актуальные для работы типовые ВМ. Для реализации услуги можно копировать образы эталонных ВМ на используемые рабочие диски. Однако одним из существенных недостатков такого решения является то, что для размещения образа виртуальной машины с установленной на ней полноценной ОС необходимо передать по сети от одного до нескольких десятков гигабайт, что выливается в длительную процедуру копирования или обновления ВМ и может привести к «коллапсу» сетевой инфраструктуры. Чтобы избежать такой ситуации, предлагается

запускать ВМ в режиме расширенного фильтра записи. В этом режиме работа образа ВМ осуществляется непосредственно из репозитария, образ ВМ при этом остаётся доступным только в режиме чтения, все необходимые для работы ОС временные файлы, сохраняются локально и исключительно на время работы ВМ, после чего уничтожаются. Такое решение снижает время загрузки образа ВМ и нагрузку на сеть. В работе исследовалось время запуска ВМ в режиме расширенного фильтра записи на платформе виртуализации, а также скорость чтения данных в зависимости от способов хранения в репозитарии эталонных ВМ, протоколов доступа к ним, а также значений харатеристик сети.

Ключевые слова: KVM, бессерверные вычисления, прогнозирование временных характеристик, SMB, NFS, NBD, ZFS, Ceph, Lustre, Windows, Linux

I. Введение

Важным фактором для обеспечения этих услуг является время старта виртуальной машины на широком спектре оборудования и платформ виртуализации. Эффективность работы таких систем во многом зависит от возможности одновременной работы с внутренними и распределенными информационными ресурсами, а также доступа в сеть Интернет.

Для решения этих задач используется программное средство общего назначения с встроенными средствами защиты, таким как «Абонентский облачный терминал» (АОТ). Это средство позволяет совмещать на одном персональном компьютере произвольное количество изолированных рабочих мест, функционирующих под контролем разных операционных систем и в своих сетевых средах. Такая изоляция доменов обеспечивает высокий уровень безопасности и надежности работы, что критично для распределенных вычислительных систем.

Рабочая инфраструктура централизованной [облачной] вычислительной структуры
(ЦВС) предприятия может содержать репозитарий контейнеров с виртуальными машинами (ВМ), реализующими рабочие места сотрудников и прошедшими принятую на предприятии процедуру внедрения в постоянную эксплуатацию. Репозитарий эталонных ВМ и обновлений содержит эталонные ВМ, серверы с обновлениями ПО. Задачи, решаемые этим компонентом ЦВС: формирование проверенных, согласованных, безопасных эталонных рабочих мест в виде контейнеров с типовыми ВМ. Образы ВМ копируются на локальные диски АОТ. Однако одним из существенных недостатков такого решения является то, что для размещения на персональном компьютере образа вир-

туальной машины с установленной на ней полноценной ОС, например, MS Windows 10, необходимо скопировать и передать по сети около 40 Гб, что выливается в длительную процедуру обновления и приводит к «коллапсу» локальной сети. Для более эффективного управления виртуальными машинами (ВМ) и минимизации нагрузки на сеть предлагается использовать режим расширенного фильтра записи. Этот режим позволяет настроить образ ВМ таким образом, чтобы он оставался доступным только для чтения. Вся информация, создаваемая или изменяемая пользователем в процессе работы, сохраняется локально на специальном слое файловой системы, а не передается по сети. Это означает, что даже если пользователь внес изменения в систему или создал новые файлы, они сохранятся локально на его компьютере.

Этот подход имеет несколько преимуществ. Во-первых, он значительно упрощает работу системного администратора, поскольку ему не нужно обновлять каждый индивидуальный образ ВМ. Вместо этого достаточно обновить только эталонные образы, которые будут использоваться для создания новых ВМ или обновления существующих. Это существенно сокращает время и усилия, затрачиваемые на обслуживание системы.

Кроме того, данный подход поддерживает возможность работы в режиме 'толстого' клиента, когда весь образ ВМ загружается на локальные диски пользователя. Это позволяет продолжить работу с ВМ даже без подключения к сети, что сохраняет автономность и мобильность рабочего места. Таким образом, пользователь может продолжать свою работу в любое время и в любом месте, не завися от доступа к сети.

Этот подход также способствует снижению времени загрузки образа ВМ и нагрузки на сеть во время работы. Поскольку изменения сохраняются локально, нет необходимости передавать большие объемы данных по сети при каждом обновлении или изменении. Это позволяет значительно улучшить производительность системы и снизить риски связанные с возможной перегрузкой сети или длительными процессами обновления. Решение поставленной задачи должно позволить снизить время загрузки образа ВМ и нагрузку на сеть во время работы.

II. Постановка задачи

A. Цель работы

Оптимизировать работу абонентских облачных терминалов, загружаемых с общего разделяемого образа виртуальной машины.

B. Формальная постановка задачи

Введем обозначения:

- w – пропускная способность сети, Мбит/сек
- $\tau 0$ – задержка отправления пакетов в сети, мс
- Ncl – количество клиентов
- Mos – тип ОС (Windows, Linux)
- Cs – критерий старта
- P – протокол доступа к данным виртуального диска, P $\in \{SMB, NBD, NFS\}$
- Fs – файловая система, Fs $\in \{ZFS, Ceph, Lustre\}$

Требуется:

Экспериментальным методом найти зависимость времени запуска $T=f_t(w, \tau 0 , Ncl, Mos, Cs, P, Fs)$ и максимальную скорость чтения данных виртуального диска $Rx=f_{Rx}(w, \tau 0, Ncl, Mos, Cs, P, Fs)$

III. Протоколы и файловая система

Было решено исследовать работу трех протоколов – SMB [2], NBD [3], NFSv4 [?]rfc7530) [5], [6],. Все они позволяют возможность масштабирования. В протоколах SMB и NBD реализованы механизмы оптимизации. Механизмы кэширования позволяют протоколу NBD справляться с изменением характеристик сети, а протоколы SMB и NFSv4 позволяют продолжить существующие сессии в случае разрыва соединения.

Также была выбрана файловая система ZFS [7], [11], [12] версии OpenZFS. Она поддерживает необходимые протоколы. Также в ней реализовано множество механизмов кэширования, а благодаря концепции copy-on-write система позволяет восстанавливать данные после сбоя.

IV. Тесты производительности

- Iozone [9] – применяется для генерации и измерения различных файловых операций, таких как чтение и запись в различных режимах.
- Bonnie++ [10] – тест применяется для создания, чтения и удаления множества небольших файлов, и изменения их метаданных. В результате своей работы он даёт пользователю производительность его файловой системы в виде измерения характеристик чтения и записи.
- Diskspd [8] – приложение Microsoft. Позволяет провести тест производительности файловой системы при помощи генерации различных файловых операций, таких как чтение и запись в различных режимах.

В исследование предполагается использование виртуальной машины Windows 7, поэтому выбрано приложение Diskspd. Оно является самым современным и разработано специально для работы с данной операционной системой.

V. Описание практической части

A. Программа исследования

- Замерить время запуска ВМ и производительности виртуального диска при использовании протоколов:
  - SMB3
  - NFSv4
  - NBD

B. Методика исследования

Тестирование проводилось с использование двух различных стендов. Первый стенд использовался для тестирования работы системы с одним клиентом.

Стенд состоит из сервера и АРМ сотрудника, которые соединены через роутер с помощью Gigabite Ethernet.

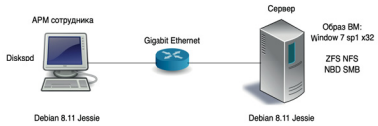

Рис. 1. Схема стенда для исследования

Сервер сконфигурирован на базе ОС Debian 8.11 Jessie, поверх которой установлен специальное ПО для администрирования АРМ сотрудника. Также на серверы установлены:

- SSD диск SanDisk SDSSDHP128G, скорость чтения 530 МБ/сек.
- Процессор Intel Core i5-4570 CPU 3.20GHz, L3: 6МБ кэша.
- Сетевая карта Intel Ethernet Connection I217-V, 256 КБ кэша.

На диск предварительно установлен образ ВМ Windows 7 sp1 x32, размером 15ГБ, в которой предустановлено приложение Diskspd и скрипты для его запуска. Также на сервере настроены протоколы и поставлены соответствующие настройки доступа к файлу с ВМ.

АРМ сотрудника сконфигурирован на базе ОС Debian 8.11 Jessie, поверх которой установлен специальное ПО для запуска ВМ. Также на нем настроена клиентская часть выбранных протоколов.

Второй стенд использовался для тестирования работы протоколов одновременно с несколькими клиентами.

Стенд состоит из сервера и четырех АРМ сотрудника, которые соединены сетью с пропускной способностью.

Сервер сконфигурирован на базе ОС Debian 8.11 Jessie, поверх которой установлен специальное ПО для администрирования АРМ сотрудника. Сервер сделан на основе: Intel Xeon Processor E5-2660 v4

Клиенты сконфигурированы аналогично клиенту на первом стенде.

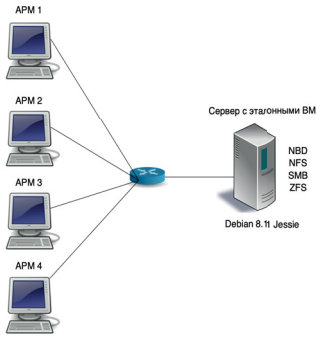

Рис. 2. Схема стенда для исследования

C. Инструменты для измерения производительности

- Ping: iputils-s20161105 – утилита используется для проверки соединения клиента с сервером, а также для фиксирования инициализации сетевой карты клиента
- Iperf3.1.3 – утилита используется для измерения скорости в сети между клиентом и сервером
- Linux Traffic Control – утилита используется для изменения скорости и задержки в сети между клиентом и сервером
- Diskspd 2.0.21a – бенчмарк используется для измерения производительности виртуального диска

D. Алгоритм тестирования

- Автоматизированное тестирование проводится с помощью скриптов на bash, находящихся на сервере.
  Константами задаются сетевые интерфейсы и адреса клиентов и сервера. Происходит запуск серверной и клиентской части протоколов (SMB3, NBD, NFSv4). Создается директория с общим доступом, в которой находится образ виртуальной машины, а также для хранения результатов тестирования.
  Проводятся тесты производительности сети (TCP) между сервером и удаленной машиной с помощью iperf. Затем происходит сбор информации о конфигурации удаленной машины и монтирование файловых систем (SMB3, NBD, NFSv4) на удаленной машине. Также перед каждым тестированием происходит очистка кэша. Происходит запуск виртуальной машины. С помощью утилиты ping и скриптов на виртуальный машине фиксируется время загрузки образа. Затем с помощью планировщика задач поочередно запускаются тесты производительности виртуального диска. Утилита для тестирования заранее установлена в образ вместе с тестовым файлом.
  Затем происходит отключение клиента от сервера и их перезапуск. С помощью утилиты Linux Traffic

Control происходит настройка сетевых интерфейсов сервера – изменяется скорость передачи данных и задержка в сети. После этого цикл начинается заново.

Результаты каждого теста сохраняются и записываются в csv-файл. В него входят: используемый протокол, задержка в сети (мс), скорость в сети (МБ/с), время загрузки ВМ (с), количество полученных сервером байт, количество отправленных сервером байт, количество пакетов, отправленных сервером, количество пакетов, полученных сервером. Тестирование позволяет получить данные о работе протоколов с различными параметрами сети, а также определить целесообразность их использования. Основными показателями, которые учитывались, были время загрузки ВМ и скорость чтения данных виртуального диска при различных значениях задержки и скорости в сети.

- Тесты проходят в режимах:
  - последовательного чтения данных блоками по 1МБ – при тестировании в таком режиме, каждый клиент скачивает с сервера блоки размером 1МБ, которые лежат последовательно
  - случайного чтения данных по случайно выбираемым смещениям блоками по 4КБ – при тестирование в таком режиме, каждый клиент запрашивает случайно выбранные блоков данных, которые могут находиться в разных местах диска
  - чтения данных по случайно выбираемым смещениям блоками по 4КБ с одновременным выполнением 32 операций чтения – при тестировании в таком режиме, каждый клиент запрашивает одновременно 32 случайно выбранных блока данных, которые могут находиться в разных местах диска. Обычно запросы помещаются в очередь и обрабатываются параллельно.

VI. Результаты экспериментов

В процессе экспериментов для каждого типа тестов получены данные, позволяющие определить целесообразность применения протокола в зависимости от параметров сети. Основными показателями, которые учитывались при анализе результатов эксперимента было время загрузки ВМ и скорость чтения данных с диска при разных значениях задержки и скорости в сети.

- Задержка – задержка в сети
- Сеть – скорость в сети
- Протокол – тестируемый протокол
- T – Время загрузки – время загрузки ВМ
- Rx – Скорость чтения – скорость чтения данных с диска

Графики показывают, что при отсутствии задержки наилучший результат демонстрирует протокол SMB.

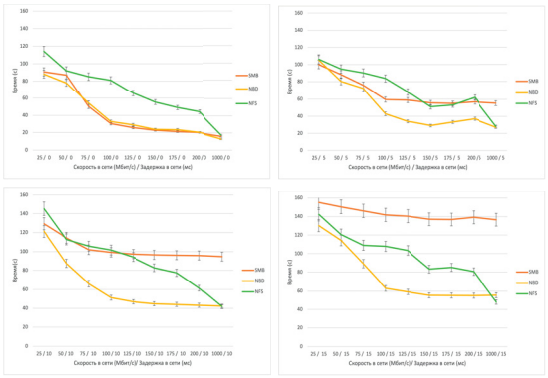

Рис. 3. Графики зависимости времени загрузки ВМ от скорости и задержки в сети с одним клиентом

Это объясняется тем, что SMB оптимизирован для быстрой передачи данных в локальных сетях и обычно обеспечивает высокую производительность при низких задержках. Однако с увеличением задержки, когда время на передачу данных становится значительным, оптимальным выбором становится протокол NBD. NBD позволяет справляться с высокими задержками более эффективно. Поэтому в среднем, при любых значениях задержки и скорости передачи данных, протокол NBD показывает наименьшее время загрузки. Протокол NFSv4 при низкой задержке показывает время больше, чем у SMB и NBD, однако с увеличением задержки его эффективность выше, чем у SMB.

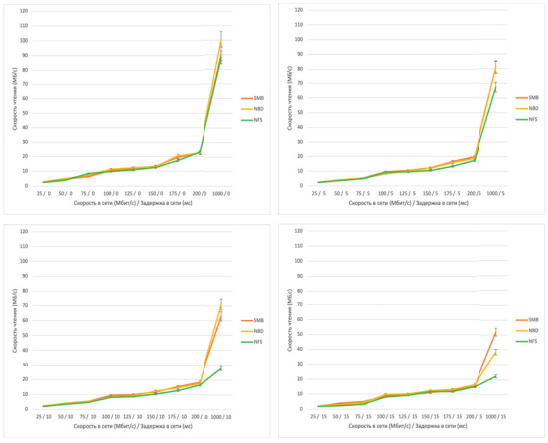

Рис. 4. Графики зависимости скорости чтения от скорости и задержки в сети при тестировании в режиме случайного чтения с одним клиентом

При низких значениях задержки от 0 до 10 мс наилучший результат показывает протокол NBD (Network Block Device). Это связано с его эффективным механизмом передачи данных через сеть, который позволяет обеспечить высокую производительность при низких задержках. Однако при увеличении задержки до 15 мс более предпочтительным становится протокол SMB. В то же время, протокол NFS во всех случаях демонстрирует наихудшие результаты, что, возможно, связано с особенностями его реализации или настройками сети.

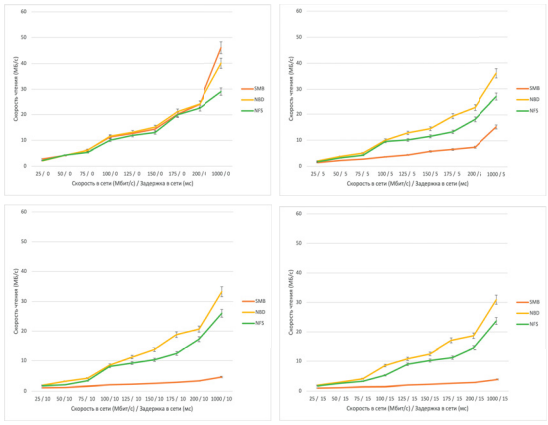

Рис. 5. Графики зависимости скорости чтения от скорости и задержки в сети при тестировании в режиме последовательного чтения с одним клиентом

При тестировании в режиме последовательного чтения, измеряется скорость чтения блоков данных, которые находятся на диске последовательного. Протоколы NFS и NBD показывают схожие результаты. При нулевой задержке, протокол SMB показывает лучший результат при скорости в 1000 Мбит/С. Вероятно, это связано с внутренним устройством протокола и механизмами кэширования.

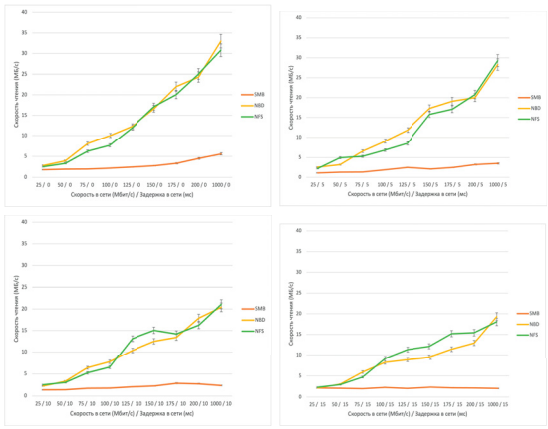

Рис. 6. Графики зависимости скорости чтения от скорости и задержки в сети при тестировании в режиме случайного чтения с одновременным выполнением 32 операций чтения с одним клиентом

При тестировании в режиме случайного чтения с одновременным выполнением 32 операций, каждый из которых генерирует одновременно 32 запроса, происходит параллельное выполнение запросов, помещенных в очередь. В условиях задержки от 0 до 10 мс наилучший результат демонстрирует протокол NFS (Network File System). Вероятно, это обусловлено его механизмами обработки запросов и эффективностью передачи данных в условиях низких задержек. Однако при увеличении задержки до 15 мс более предпочтительным становится протокол NBD (Network Block Device), вероятно, благодаря его механизмам обработки данных и более эффективному управлению задержками.

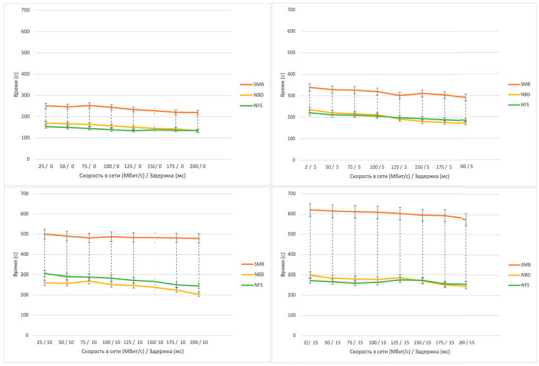

Рис. 7. Графики зависимости времени загрузки ВМ от скорости и задержки в сети с четырьмя клиентами

На графике показан результат работы системы с 4 клиентами. Они схожи с результатами работы одного клиента. Протокол SMB показывает худший результат даже с минимальной задержкой. При его использовании время загрузки может достигать 600 секунд. Протоколы NFSv4, в среднем, работает лучше, чем NBD при высокой задержке в сети. Однако в остальных случаях результаты почти одинаковые.

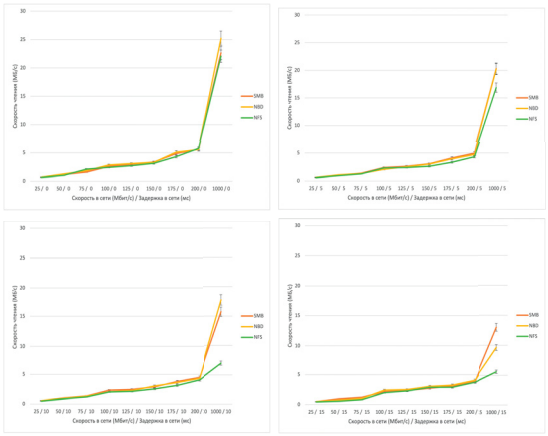

Рис. 8. Графики зависимости скорости чтения от скорости и задержки в сети при тестировании в режиме случайного чтения с четырьмя клиентом

В этом тесте, в отличие от случая с четырьмя клиентами, при низких значениях задержки от 0 до 10 мс протоколы SMB и NBD показывают схожие результаты. Однако при увеличении задержки до 15 мс более предпочтительным становится протокол SMB. Это объясняется тем, что протокол SMB лучше справляется с передачей данных в условиях более высоких задержек. В то же время, протокол NFS во всех случаях демонстрирует результаты хуже. Заметно, что пропускная способность делится между клиентами равномерно.

При тестировании в режиме последовательного чтения, измеряется скорость чтения блоков данных, которые находятся на диске последовательного. Протоколы NFS и NBD, как и в случае с одним клиентом, показывают схожие результаты. При нулевой задержке,

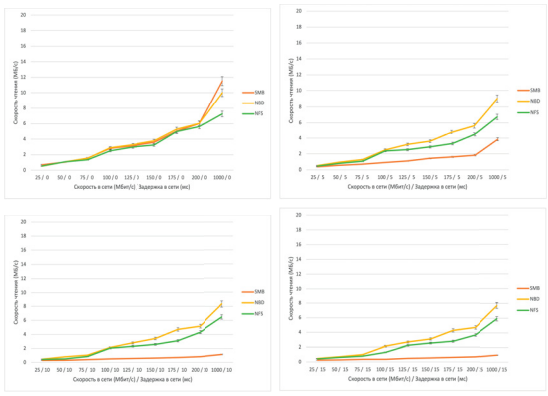

Рис. 9. Графики зависимости скорости чтения от скорости и задержки в сети при тестировании в режиме последовательного чтения с четырьмя клиентом

все протоколы показывают схожие результаты, однако при увеличении задержки, протокол SMB начинает работать сильно хуже. При высокой скорости в сети, протокол NBD показывает лучший результат.

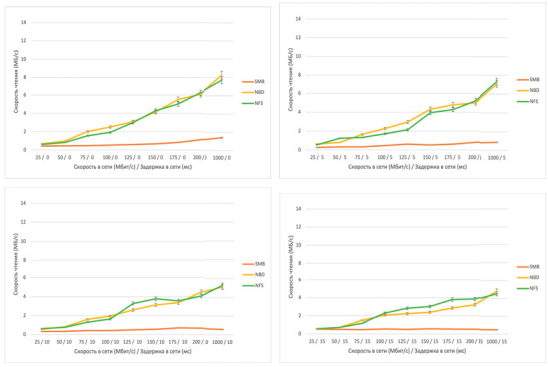

Рис. 10. Графики зависимости скорости чтения от скорости и задержки в сети при тестировании в режиме случайного чтения с одновременным выполнением 32 операций чтения с четырьмя клиентом

При тестировании в режиме случайного чтения с одновременным выполнением 32 операций, каждый из которых генерирует одновременно 32 запроса, происходит параллельное выполнение запросов, помещенных в очередь. При низкой скорости в сети и минимальной задержке, протоколы показывают схожие результаты. В условиях задержки от 5 до 15 мс наилучший результат демонстрирует протокол NBD.

По результатам тестов с протокол NBD в среднем показал наилучшие результаты. Время загрузки BM при его использовании в большинстве случаев оказывается наименьшим. Это связано с тем, что в протоколе NBD реализованы эффективные механизмы кэширования данных, которые позволяют уменьшить количество запросов и увеличить скорость доступа к данным.

Протокол SMB показал хорошие результаты во время измерения времени загрузки BM при условии маленькой задержки. В других случаях время загрузки больше, чем у NFS и NBD, так как протокол чувствителен к изменению задержки в сети. Также его производительность в тестах скорости чтения оказалась очень низкой. NFS в среднем показал результаты лучше, чем протокол SMB. Время загрузки BM при низкой задержке гораздо больше, чем у NBD и SMB. При увеличении задержки время загрузки BM с использованием NBD становится меньше, чем при использовании SMB.

VII. Заключение

Исследование показало, что оптимальным выбором для работы с BM в таком режиме является протокол NBD. В качестве файловой системы использовать ZFS. Благодаря оптимизационным механизмам, реализованным в нем, он эффективен при разных значениях скорости и задержки в сети. Однако в случае низкой задержки в сети лучше выбирать протокол SMB3, так как он позволит сократить время загрузки BM.

Список литературы

[1] Грушо, А.А., Николаев, А.В., Писковский, В.О., Сенчило, В.В., Тимонина, Е.Е., Подход к обеспечению информационной безопасности при использовании частных облачных вычислительных сред. Абонентский облачный терминал, MoNeTec-2020

[2] [MS-SMB2]: Server Message Block (SMB) Protocol Versions 2 and 3, Microsoft, 2020, Дата доступа: 02-11-2022, https://learn.microsoft.com/en-us/openspecs/windows_protocols/ms-smb2/5606ad47-5ee0-437a-817e-70c366052962

[3] The NBD protocol description, Дата доступа: 25-02-2023, https://github.com/NetworkBlockDevice/nbd

[4] RFC 7530, Дата доступа: 02-10-2022, Internet Engineering Task Force (IETF), https://www.rfc-editor.org/rfc/rfc7530.html

[5] Ming Chen, Dean Hildebrand, Geoff Kuenning, Soujanya Shankaranarayana, Vasily Tarasov, Arun O. Vasudevan, Erez Zadok, Ksenia Zakirova, Linux NFSv4.1 Performance Under a Microscope, Stony Brook University, IBM Research—Almaden

[6] An Overview of NFSv4, Storage Networking Industry Association, Дата доступа: 26-10-2022, https://www.snia.org/sites/default/files/SNIA_An_Overview_of_NFSv4-30.pdf

[7] ZFS Documentation, OpenZFS Community, https://openzfs.org/wiki/System_Administration, Дата доступа: 2024-01-20

[8] Diskspd, Microsoft, Дата доступа: 25-02-2023, https://github.com/microsoft/diskspd

[9] Iozone Documentation, Дата доступа: 25-02-2022, https://www.iozone.org/docs/IOzone_msword_98.pdf

[10] Bonnie++ documentation, Дата доступа: 25-02-2022, https://www.coker.com.au/bonnie++/readme.html

[11] Gurjar, Devyani and Kumbhar, Satish S., File I/O Performance Analysis of ZFS & BTRFS over iSCSI on a Storage Pool of Flash Drives, IEEE, 2019

[12] Phrornchana, Veerapat, Nupairoj, Natawut, Pirornsopa, Krerk, Performance Evaluation of ZFS and LVM (with ext4) for Scalable Storage System, IEEE, 2011

Задача синтеза управления транспортными потоками и ее решение методом машинного обучения

Софронова Е.А.
Федеральный исследовательский центр «Информатика и управление» РАН
Москва, Россия
sofronova_ea@mail.ru

Дивеев А.И.
Федеральный исследовательский центр «Информатика и управление» РАН
Москва, Россия
aidiveev@mail.ru

Белотелов В.Н.
Федеральный исследовательский центр «Информатика и управление» РАН
Москва, Россия
vbelotelov@gmail.com

Бобков А.А.
Федеральный исследовательский центр «Информатика и управление» РАН
Москва, Россия
bobkov_aa@mail.ru

***Аннотация* — Рассматривается математическая модель управления транспортными потоками на сети городских дорог. Модель представляет собой ориентированный граф, в котором узлам графа соответствуют участки дорог, а ребрам — маневры на перекрестках. В модель включено управление транспортным потоком, которое осуществляется путем переключения фаз светофоров на перекрестках. Переключение фаз вызывает разрешение или запрет определенных маневров, что соответствует удалению или созданию определенных ребер графа. Модель транспортного потока с известными параметрами — пропускной способностью маневров и распределением потоков по альтернативным направлениям — позволяет рассчитать количественные характеристики изменения транспортных потоков на всех участках дорог на каждом временном шаге. Для расчета используется система конечно-разностных уравнений. Для рассматриваемой модели сформулирована задача оптимального управления. Результатом решения задачи оптимального управления является оптимальный план координации работы светофоров. Планы координации обычно рассчитываются для конкретного времени суток, дня недели и даже сезона. В оптимальном плане координации длительность рабочих фаз светофоров рассчитывается в зависимости от параметров перекрестка и объема транспортного потока с учетом ограничений на длительность фаз светофора, порядок следования фаз и длительность цикла фаз. Параметры перекрестка включают в себя график разрешенных на перекрестке маневров, набор фаз, разрешающих маневры, а также пропускную способность каждого маневра и долю распределения потока по альтернативным направлениям. Транспортный поток измеряется в средних единицах транспортных средств. Параметры определяются на основе проектной документации на перекрестки рассматриваемой сети и данных, полученных с датчиков и видеокамер дорожной инфраструктуры. Оптимальные планы координации рассчитываются заранее с использованием статистических данных. В статье рассматривается ситуация, когда происходит резкое изменение транспортного потока. Тогда рассчитанный оптимальный план координации перестает быть оптимальным. Резкое изменение транспортного потока может быть вызвано дорожно-транспортными происшествиями, ремонтом дорог и т. д. В таких случаях предлагается синтезировать функцию управления, корректирующую длительность фаз светофора на основе разницы между фактическим потоком и потоком, параметры которого были использованы для расчетного плана координации. Синтез функции управления осуществляется заранее вместе с планами координации на основе данных, полученных из модели. В статье формулируется задача синтеза управления транспортным потоком и предложены два подхода к ее решению. Первый подход основан на использовании машинного обучения, в частности – символьной регрессии. Методы символьной регрессии позволяют найти математическое выражение функции, записанное в виде специального кода, с использованием эволюционного алгоритма. Результатом решения задачи синтеза является функция управления, изменяющая длительность фаз светофора в зависимости от потока транспорта на участках сети. Второй подход основан на решении задачи классификации, когда текущая ситуация оценивается некоторым набором параметров, а далее происходит ее отнесение к одному из заранее описанных классов ситуаций, для которых есть расчетные программы управления.**

Keywords* — *поток транспортных средств, транспортный поток, управление транспортными потоками, моделирование транспортных потоков, оптимизация транспортных потоков, дорожная сеть, управляемые сети.

I. Введение

Основным методом управления транспортными потоками в больших городах на сегодняшний день

является светофорное управление. Управление осуществляется с помощью переключения фаз светофоров, где каждая фаза характеризуется своей длительностью. В таком способе длительность каждой фазы может быть фиксированной, рассчитанной заранее, а также может использоваться адаптивное переключение, которое вызывается данными о наличии транспортных средств (ТС) в непосредственной близости от перекрестка, получаемыми с датчиков дорожной инфраструктуры [1-3]. В работе решаются задачи управления для метода светофорного управления с рассчитываемой длительностью фаз.

Набор фаз, порядок их переключения и их длительности входят в план координации [4]. Планы координации зависят от времени суток, дня недели и времени года. Обычно планы координации строятся на основе опыта специалистов, отвечающих за организацию дорожного движения. В последнее время при составлении планов координации используются пакеты имитационного моделирования [5], которые позволяют оценить изменение транспортного потока при разных планах координации.

В 2008 г. была сформулирована задача оптимального управления транспортными потоками [6]. Задача оптимального управления заключается в поиске такого плана координации работы светофоров (управления), который обеспечивает минимум заданного критерия качества для заданной модели перекрестка или сети с учетом ограничений на длительности фаз светофора, порядок следования фаз, длительность цикла фаз и др. Для решения задачи оптимального управления была построена математическая модель в виде системы конечно-разностных рекуррентных уравнений [6-7]. Модель включает описание графа сети дорог, параметров транспортной сети и транспортного потока. Для определенных фаз светофоров модель позволяет рассчитать изменение транспортного потока на всех участках дорог сети. Сеть дорог разделена на участки, каждый из которых характеризуется максимальной вместимостью ТС и числовой оценкой текущего состояния, выраженной в количестве ТС усредненного размера. Трудность применения оптимальных планов координаций состоит в том, что в реальных условиях возможны не предусмотренные при расчетах изменения транспортных потоков.

В работе предлагается решать задачу управления транспортными потоками как задачу синтеза, в которой необходимо найти управление как функцию вектора пространства состояний. Решение задачи синтеза обеспечит возможность учитывать непредвиденные изменения транспортного потока, например, ДТП, изменение погодных условий, проведение массовых мероприятий, которые провоцируют резкое увеличение транспортного потока и пр. Все это приводит к резкому изменению состояния объекта. При этом синтезированная функция управления должна обеспечить получение оптимального решения без дополнительных вычислений, в то время как программное управление, получаемое при решении задачи оптимального управления, может быть рассчитано только для ограниченных и заранее предусмотренных изменений потока. В качестве альтернативного подхода исследуется разработка системы выбора подходящего плана координаций среди базы заранее рассчитанных оптимальных планов. После формирования базы оптимальных планов координации необходимо разработать систему классификации или выбора оптимального плана из базы по соответствующему состоянию потока.

II. Постановка задачи синтеза

Рассмотрим задачу синтеза управления транспортными потоками в сети городских дорог. Задана рекуррентная конечно-разностная математическая модель управления транспортными потоками в сети городских дорог, построенная на основе теории управляемых сетей [7, 8]

$$\mathbf{x}(k+1) = \mathbf{x}(k) + \mathbf{f}(\mathbf{x}(k), \mathbf{A}(\mathbf{u}(k))) + \boldsymbol{\delta}(k), \qquad (1)$$

$$\mathbf{x} \in \mathbb{R}^n,\ \mathbf{u} \in \mathrm{U} \subseteq \mathbb{R}^m.$$

Значениями компонент вектора состояния $\mathbf{x}(k) = [x_1(k) \ldots x_L(k)]^T$ являются числовые оценки транспортного потока в единицах ТС усредненного размера на множестве всех участков дорог L рассматриваемой сети. Время дискретизировано на такты управления k, $k = \overline{1, K}$, K - заданное количество тактов управления. Изменение транспортного потока за один такт управления определяется функцией $\mathbf{f}(\mathbf{x}(k), \mathbf{A}(\mathbf{u}(k)))$, зависящей от вектора состояния и матрицы конфигураций $\mathbf{A}(\mathbf{u}(k))$. Матрица конфигураций представляет собой матрицу смежности графа сети дорог с маневрами, которые разрешены управлением на текущем такте. Кроме этого, на каждом такте управления происходит приращение входного потока $\boldsymbol{\delta}(k) = [\delta_1(k) \ldots \delta_L(k)]^T$.

Управляющими параметрами являются длительности рабочих фаз светофоров на всех регулируемых перекрестках M в сети, $\mathbf{u}(k) = [u_1(k) \ldots u_M(k)]^T$. Длительности рабочих фаз ограничены. Последовательность переключения фаз определена заранее и не изменяется. Задана область начальных состояний $\mathrm{X}^0 \subseteq \mathbb{R}^n$. На каждом участке дороги установлены минимальное и максимальное допустимые значения количества ТС.

Задан критерий качества управления:

$$J = \sum_{k=1}^{K} f_0(\mathbf{x}(k), \mathbf{A}(\mathbf{h}(\mathbf{x}(k)))) \to \min. \qquad (2)$$

Необходимо найти функцию управления как функцию от вектора начального состояния

$$\mathbf{u} = \mathbf{h}(\mathbf{x}(0), \delta(0)) \in \mathrm{U}, \qquad (3)$$

которая минимизирует критерий качества (2).

III. Методы решения

В настоящей работе предлагается решать задачу синтеза управления потоками транспорта методом символьной регрессии [9-12]. Этот метод позволяет получить нелинейную функцию управления, аргументом которой является вектор состояния. Для повышения эффективности выбранного метода используется подход, основанный на аппроксимации оптимальных планов координации. При таком подходе первоначально определяется множество начальных состояний

$$\mathrm{X} = \{\mathbf{x}^{0,1},\ldots,\mathbf{x}^{0,S}\} \in \mathrm{X}^0 . \quad (4)$$

Далее для каждого начального состояния решается задача оптимального управления [13, 14] по критерию (2) и находятся оптимальные планы координации

$$\mathrm{U}^* = \{\mathbf{u}^{*,1}(k),\ldots,\mathbf{u}^{*,S}(k)\} \in \mathrm{U} . \quad (5)$$

Моделирование объекта управления (1) с начальным условием из (4) и соответствующим управлением из (5) приводит к получению программного изменения транспортного потока. Множество программ изменения транспортного потока, образованных путем такого моделирования с различными начальными условиями из (4) имеет вид

$$\mathrm{X}^* = \{\mathbf{x}^{*,1}(k),\ldots,\mathbf{x}^{*,S}(k)\} . \quad (6)$$

Решение задачи синтеза осуществляется на основе аппроксимации множества программ изменения транспортного потока (6), то есть функция управления (3), находится по критерию

$$J = \sum_{j=1}^{S}\sum_{k=1}^{K}\left\|\mathbf{x}(k) - \mathbf{x}^{*,j}(k)\right\| \to \min . \quad (7)$$

где $\mathbf{x}(k)$ - состояние транспортного потока в момент k, вычисленное по модели (1) для плана координаций, сгенерированного функцией управления (3).

В результате находим математическое выражение для функции управления (3), согласно которой в зависимости начального состояния $\mathbf{x}(0)$ получаем план координаций, наиболее близкий к оптимальному из множества (5). Функция управления (3) обеспечивает возможность получения оптимального плана координаций для начального состояния, не входящего во множество (4). Задача синтеза управления является вычислительно сложной и для моделей больших размерностей требует существенных вычислительных ресурсов.

Для больших сетей дорог предлагается использовать альтернативный подход. Подход основан на оценке текущего состояния и его классификации для применения определенного плана координации из базы оптимальных планов (5). Для этой цели предлагается использовать искусственную нейронную сеть, которая обучена решать задачу классификации и определять план координации из (5), соответствующий данному текущему состоянию $\mathbf{x}(k)$

$$f_N(\mathbf{x}(k)) = l \ , \ l \in \{1,\ldots,S\} , \quad (7)$$

где $f_N(\mathbf{x}(k))$ - функция искусственной нейронной сети, $\mathbf{u}^{*,l}$ - план координации, соответствующий текущему состояний $\mathbf{x}(k)$.

Обучение искусственной нейронной сети производится на этапе проектирования после решения задачи оптимального управления. Обученная нейронная сеть функционирует в системе управления транспортным потоком и определяет соответствие текущего состояния текущему плана координации. Изменение планов координации необходимо выполнять дискретно, через заданные интервалы времени.

IV. Практическая реализация

Схема решения и функционирования системы на основе решения задачи синтеза управления методом символьной регрессии представлена на рис. 1. Под сценарием понимается состояние объекта управления в некоторый момент времени. В результате решения задачи синтеза получается математическое выражение функции управления. Текущее состояние объекта передается в найденное управление и формируется план координации.

На рис. 2 приведена схема реализации подхода, основанного на классификации дорожной обстановки и выборе оптимального плана координации. Первоначально решается задача обучения искусственной нейронной сети для классификации планов координации, а далее по данным о текущем состоянии объекта и текущем плане координации обученная нейронная сеть осуществляет выбор плана координации из существующей базы планов координации.

Одну из важнейших ролей при решении задачи синтеза играет модель объекта управления. Вопрос валидации использованной модели на предмет ее соответствия реальному процессу движения транспортных потоков в статье не представлен, хотя он подразумевается.

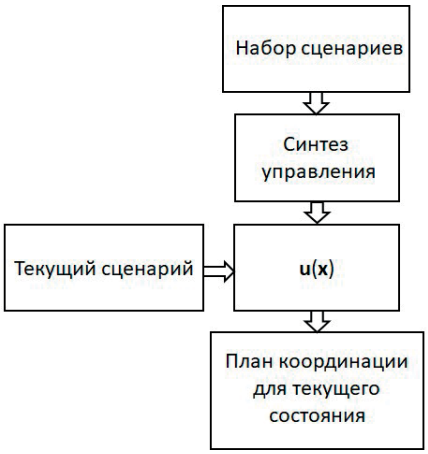

Рис.1. Реализация подхода на основе синтеза управления.

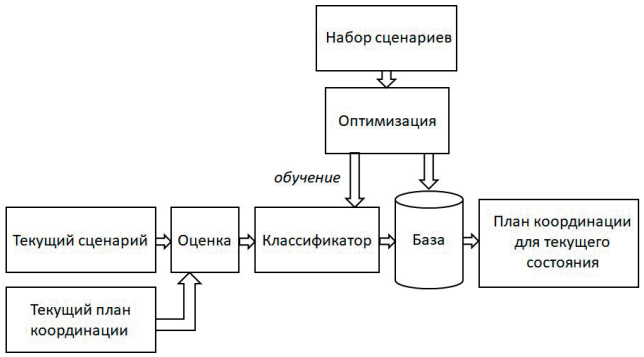

Рис.2. Реализация подхода на основе выбора оптимального плана координации.

V. Вывод

Предложено решать задачу управления транспортными потоками как задачу синтеза управления, в результате решения которой получается план координаций, соответствующий текущему состоянию потока. Сформулирована задача синтеза управления транспортными потоками и представлены два подхода к ее решению.

Первый подход основан на решении задачи синтеза методом символьной регрессии и получении математического выражения функции управления как функции текущего состояния. Для решения задачи синтеза предложено использовать метод аппроксимации оптимальных программ управления или оптимальных планов координации. Применение данного подхода позволяет получить оптимальный план координации в реальном времени. При этом полученный план координации может не совпадать ни с одним планом координации из множества предварительно рассчитанных оптимальных планов координации. Сложность данного подхода заключается в необходимости использования больших вычислительных ресурсов.

Второй подход основан на выборе оптимального плана координации из множества предварительно рассчитанных оптимальных планов координации. Для решения задачи классификации планов координации предложено использовать предварительно обученную искусственную нейронную сеть.

Оба предложенных подхода для решения задачи синтеза управления планируется использовать в разрабатываемой интеллектуальной транспортной системе (ИТС CTraf).

Список литературы

[1] Eom M., Kim B.I. The traffic signal control problem for intersections: a review // Eur. Transp. Res. Rev. — 2020. — Vol. 12, no. 50.

[2] Wei H., Zheng G., Gayah V., Li Z. A survey on traffic signal control methods // arXiv:1904.08117 [cs.LG]. — 2020.

[3] Global practices on road traffic signal control: Fixed-time control at isolated intersections / Keshuang Tang, Manfred Boltze, Hideki Nakamura, Zong Tian. — Elsevier, 2019.

[4] Кременец Ю.А., Печерский П.М., Афанасьев М.Б. Технические средства организации дорожного движения. М.: ИКЦ «Академкнига», 2005. – 279 с.

[5] Chao Q., Bi H., Li W., Mao T., Wang Z., Lin M.C., Deng Z. A Survey on Visual Traffic Simulation: Models, Evaluations, and Applications in Autonomous Driving. Computer Graphics Forum. 2019. doi:10.1111/cgf.13803.

[6] Дивеев А.И. Управляемые сети и их приложения// Журнал вычислительной математики и математической физики. 2008. 48(8). С. 1510-1525.

[7] Софронова Е.А. Универсальная рекуррентная модель управления транспортными потоками в классе микроскопических моделей // Вестник ВГУ. Серия: Системный анализ и информационные технологии. – 2021. – № 4. – С. 3–29.

[8] Дивеев А.И. Теория управляемых сетей и ее приложения. М.: ВЦ РАН, 2007. – 160 с.

[9] Дивеев А.И. Численные методы решения задачи синтеза управления: монография. М.: РУДН, 2019. –192 с.

[10] Affenzeller, M., Burlacu, B., Dorfer, V., Dorl, S., Halmerbauer, G., Konigswieser, T., Kommenda, M., Vetter, J., Winkler, S.: White box vs. black box modeling: On the performance of deep learning, random forests, and symbolic regression in solving regression problems. In: Moreno-D´ıaz, R., Pichler, F., Quesada-Arencibia, A. (eds.) Computer Aided Systems Theory – EUROCAST 2019. pp. 288–295. Springer International Publishing, Cham (2020)

[11] Makke, N., Chawla, S.: Interpretable scientific discovery with symbolic regression: a review. Artificial Intelligence Review 57(1), 2 (Jan 2024).

[12] . La Cava, W., Orzechowski, P., Burlacu, B., de Fran¸ca, F.O., Virgolin, M., Jin, Y., Kommenda, M., Moore, J.H.: Contemporary symbolic regression methods and their relative performance. arXiv:2107.14351 (2021).

[13] Софронова Е.А. К задаче оптимального управления транспортными потоками // В сборнике: XVI Всероссийская мультиконференция по проблемам управления (МКПУ-2023). Материалы мультиконференции. В 4 т. Волгоград, 2023. с. 185-187.

[14] Софронова Е.А., Дивеев А.И., Использование реальных данных из нескольких источников для оптимизации транспортных потоков в пакете CTraf // Компьютерные исследования и моделирование, Т.10, №1, 2023. С. 1-16.

Методы организации повторной обработки пакетов в конвейере сетевого процессорного устройства

Дмитрий Стамплевский[1], Дмитрий Волканов[2]

Московский Государственный Университет имени М. В. Ломоносова

Факультет Вычислительной Математики и Кибернетики

Кафедра автоматизации систем вычислительных комплексов[1,2]

[1]stamplevskiyd@gmail.com

[2]volkanov@asvk.cs.msu.ru

***Аннотация* — Сетевое процессорное устройство (СПУ) — это программируемый процессор, архитектура которого оптимизирована для работы в сетевых устройствах и обеспечения устойчивого режима обработки пакетов. Его основные задачи — выделение заголовков пакетов, осуществление классификации и обработка заголовков. Классификация происходит на конвейерах, состоящих из специальных вычислительных ячеек. Существуют заголовки, обработка которых занимает длительное время. В это время конвейер не может обрабатывать новые пакеты, в результате чего они теряют актуальность. Одно из решений данной проблемы — технология возвратной очереди. Она подразумевает возврат пакета в начало одного из конвейеров для дальнейшей обработки. В работе предложен подход к организации возвратной очереди, а также проведен обзор существующих алгоритмов выбора конвейера, на который отправится повторно обрабатываемый пакет. Механизм возвратной очереди был реализован в имитационной модели СПУ RuNPU. В имитационную модель добавлены реализации выбранных на основании обзора алгоритмов с возможностью переключения используемого алгоритма. Были проведены эксперименты с изменением доли повторно обрабатываемых пакетов, а также с неравномерным распределением пакетов между конвейерами. Был выбран лучший среди алгоритмов выбора конвейера — Central Manager. Он подразумевает отслеживание уровня загрузки всех конвейеров и поиск среди них наименее загруженного.**

***Ключевые слова* — Сетевое процессорное устройство, NPU, Конвейер, Повторная обработка пакетов, Распределение задач**

I. Введение

В настоящее время активно развиваются программно-конфигурируемые сети (ПКС) [1]. В ПКС уровни управления сетью и передачи данных разделяются между двумя классами устройств: контроллерами и коммутаторами. Контроллеры отвечают за управление сетью, коммутаторы – за обработку и передачу пакетов.

Основным элементом ПКС коммутатора является сетевое процессорное устройство (СПУ) – программируемый процессор, архитектура которого оптимизирована для использования в сетевых устройствах и обеспечения устойчивого режима обработки пакетов [2].

Его основные задачи – классификация полученных пакетов по их заголовкам и их обработка. Этим занимаются конвейеры СПУ.

Некоторые пакеты могут обрабатываться достаточно долго. Это приведет к накоплению пакетов во входных очередях коммутатора (ведь пакеты не могут начать обработку из-за занятости конвейера СПУ) и потере актуальности этих пакетов.

Для решения данной проблемы было предложено ввести лимит числа тактов, за которое пакет должен обработаться на конвейере. При достижении этого лимита нужно остановить обработку, сохранить текущие результаты работы и отправить заголовок в начало другого конвейера, где обработка продолжится с учетом сохраненных результатов. Текст состоит из введения, постановки задачи, описания предметной области, в том числе описания сетевого процессора и самой идеи повторной обработки пакетов, обзора методов выбора конвейера для повторной обработки, описания имитационной модели СПУ, на которой проводились эксперименты, экспериментального исследования и его итогов, и заключения.

II. Постановка задачи

Требуется разработать и реализовать метод выбора конвейера СПУ для повторной обработки заголовка, соблюдая следующие требования:

- Все конвейеры в архитектуре должны быть одинаковыми и работать синхронно.
- Выбор конвейера для повторной обработки должен происходить быстро, чтобы СПУ не простаивал, ожидая решения.
- Повторная обработка пакета должна завершаться корректно при любом наборе правил таблиц классификации.
- Пакет должен покидать конвейер только после прохождения всех ячеек (при достижении лимита

числа тактов пакет передается между ячейками без модификаций).

III. Описания предметной области

Основная идея программно-конфигурируемых сетей – разделение уровней передачи данных и управления сетью. Это помогает более гибко настраивать топологию сети и увеличивает универсальность отдельных устройств [1].

Коммутатор в ПКС занимается обработкой и передачей пакетов (уровень передачи данных).

У ПКС коммутатора есть СПУ (сетевое процессорное устройство), ЦПУ (центральное процессорное устройство), входные и выходные порты, а также система.

СПУ отвечает непосредственно за обработку пакетов. ЦПУ управляет коммутатором (собирает данные о работе, управляет питанием и прочими модулями), а также отвечает за взаимодействие с контроллером (устройством уровня управления сетью).

Помимо этого, у ПКС коммутатора есть оперативная и постоянная память, блок питания и система охлаждения, но они не относятся к теме данного исследования.

Основное устройство в коммутаторе – СПУ. Он занимается обработкой и классификацией заголовков пакетов. СПУ оптимизирован именно для работы с пакетами, что позволяет ускорить их обработку и снизить задержки в сети. Они могут выполнять операции с пакетами, такие как классификация, маршрутизация, фильтрация и так далее.

В данной работе рассматривается СПУ с конвейерной архитектурой, использующей для обработки пакетов несколько одинаковых конвейеров [3].

Обработка пакетов происходит следующим образом. Коммутатор получает пакет с одного из входных портов и передает его СПУ. На СПУ пакет попадает в один из буферов входной очереди, где находится до начала обработки. Затем он попадает в разделитель памяти. Его задача – отделить заголовок пакета от его тела, которое отправляется в основную память. Оно не нужно при обработке.

В исследовании рассматривается конвейерная архитектура СПУ. Она подразумевает несколько одинаковых конвейеров, состоящих из вычислительных ячеек.

Заголовок обрабатывается на одном из конвейеров, на каком именно – зависит от реализации логики выбора конвейера в СПУ.

Конвейер обычно состоит из нескольких ячеек. У каждой ячейки есть входной порт, вычислительное ядро, память и выходной порт. Память хранит внутреннюю программу ячейки и таблицы классификации. Вычислительное ядро отвечает за обработку заголовка.

В процессе обработки ячейка последовательно просматривает правила в своей памяти и после того, как найдет подходящее, модифицирует заголовок и передает его на выходной порт. Или же выполняет более сложные действия в зависимости от программы.

После завершения обработки на конвейере заголовок пакета попадает в буфер выходной очереди конвейера. Затем – вновь соединяется с телом и отправляется в буфер выходной очереди коммутатора, а оттуда на выходной порт.

Для решения проблемы долгой обработки пакетов вводится лимит числа тактов работы конвейера на обработку одного заголовка.

В случае, если во время обработки заголовок достигает лимита времени обработки, в предложенной архитектуре происходит следующее:

Обработка останавливается, заголовок передается в выходную очередь без обработки в следующих ячейках, если такие есть.

Заголовок пакета содержит набор метаданных – некой дополнительной информации о ходе обработки пакета (номер в последовательности или число тактов). Метаданные добавляются заголовку на этапе отделения от тела, а также в ходе обработки на конвейере.

В случае повторной обработки в метаданных делается пометка о том, что обработка не завершена.

Также сохраняется контекст обработки – набор данных о ходе обработки, необходимый для корректного завершения обработки на новом конвейере.

Поэтому нужна специальная программы, работающая на ЦПУ, которую будем называть менеджером повторной обработки.

Менеджер будет просматривать все выходящие с конвейеров пакеты и благодаря отметкам в метаданных определять необработанные пакеты, для которых он выберет конвейеры и передаст заголовки в их входные очереди.

Как следствие, менеджер также должен отслеживать загруженность конвейеров и осуществлять сохранение контекста обработки.

На рис. 1 представлена схема СПУ с поддержкой повторной обработки.

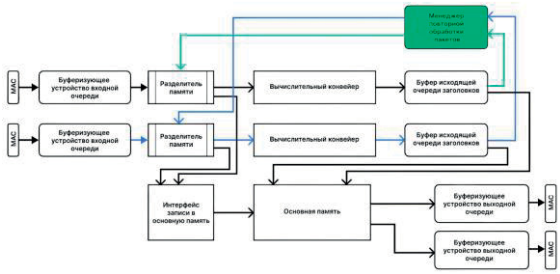

Рис. 1: Схема СПУ с поддержкой повторной обработки.

IV. Обзор методов выбора конвейера для повторной обработки

Были выбраны следующие критерии оценки алгоритмов выбора конвейера:

- Расход памяти СПУ
- Скорость принятия решения
- Сложность реализации

- Стоимость реализации
- Качество работы (алгоритм должен обеспечивать максимально равномерную загрузку конвейеров).

Рассматривались несколько алгоритмов и на основе критериев были выбраны те, которые будут применяться в исследовании.

Задача выбора конвейера для повторной обработки пакета во многом похожа на задачу балансировки нагрузки.

Алгоритмы распределения нагрузки бывают:

Динамические [4]. Они используют информацию о состоянии системы для принятия решений на протяжении всего времени работы системы и способны менять решение о распределении задач в процессе работы (то есть в контексте СПУ они могли бы переносить заголовок с одного конвейера на другой прямо в процессе его обработки, что противоречит заданным ранее требованиям).

Статические [4]. В статических алгоритмах решение принимается до начала выполнения работы и не изменяется. Статические алгоритмы все-таки способны учитывать загрузку системы (загрузку конвейеров в случае СПУ), но только до момента принятия решения.

Соответственно, для рассматриваемой архитектуры применимы только статические алгоритмы.

Среди них для исследования были выбраны следующие алгоритмы.

Round Robin: Конвейеры для повторной обработки выбираются по очереди [5]. Например, если алгоритм отправил пакет на дообработку на 3-й конвейер, то в следующий раз он отправит пакет на 4-й. После последнего будет выбран первый и так далее.

Round Robin прост в реализации и не подразумевает серьезной доработки СПУ.

Randomized algorithm: Конвейеры выбираются случайным образом. Такой алгоритм прост и дешев в реализации и требует минимальный объем памяти. Однако он достаточно сложен в исследовании из-за непредсказуемого поведения, из-за чего в дальнейшем рассматриваться не будет.

Central Manager: Central Manager [5] выбирает наименее загруженный конвейер для каждой повторной обработки и отправляет заголовок на него. Такой подход учитывает реальную загрузку системы и не добавляет работы уже загруженным конвейерам, что дает лучшие результаты. При этом алгоритм требует наличия средств мониторинга загрузки конвейеров. В качестве критерия загруженности конвейера можно выбрать число пакетов в его входной очереди.

Dedicated Pipeline: В архитектуру СПУ добавляется еще один конвейер. Он занимается исключительно повторной обработкой пакетов.

Этот подход прост в реализации, работает мгновенно и не требует дополнительной памяти. Но при большом числе повторно обрабатываемых пакетов есть риск перегрузки выделенного конвейера. К тому же добавление еще одного конвейера увеличит итоговую стоимость СПУ.

Send Same: В случае повторной обработки пакет отправляется на тот же конвейер, где обрабатывался до этого.

Send Same не требует много памяти и прост в реализации. Если повторная обработка происходит одинаково часто на всех конвейерах, все они загружаются равномерно. Если же нет – те конвейеры, где она происходит чаще, могут перегружаться, а остальные будет простаивать.

Throttled algorithm: Для каждого конвейера вычисляется порог загрузки, при достижении которого конвейер считается перегруженным. Заголовок отправляется на любой из не перегруженных конвейеров. Алгоритм может обеспечить достаточно хороший уровень распределения нагрузки, однако, вычисление порога проблематично. Нужно выяснить, какой порог задать, как часто его перевычислять. И это очень сильно усложняет реализацию алгоритма.

Для исследования были выбраны Round Robin, Central Manager, Dedicated Pipeline, Send Same.

V. Имитационная модель СПУ

Эксперименты данного исследования проводились на эмуляторе ПКС коммутатора – имитационной модели (ИМ) СПУ RuNPU. Она написана на языке программирования Python 3.

Модель состоит из нескольких модулей: управляющий модуль, а также модули, отвечающие за функции входной очереди, конвейера (в том числе модули, отвечающие за работу ячеек конвейера), упрощенный модуль ЦПУ. Время работы измеряется в тактах работы конвейера. Считается, что ЦПУ работает очень быстро и время его работы не учитывается моделью.

В рамках исследования в имитационную модель был добавлен модуль, отвечающий за работу менеджера повторной обработки с несколькими алгоритмами выбора конвейера. Также была реализована возможность отправки пакета на новый конвейер, сохранение контекста обработки и ряд менее значительных нововведений.

Схема модифицированной имитационной модели представлена на рис. 2.

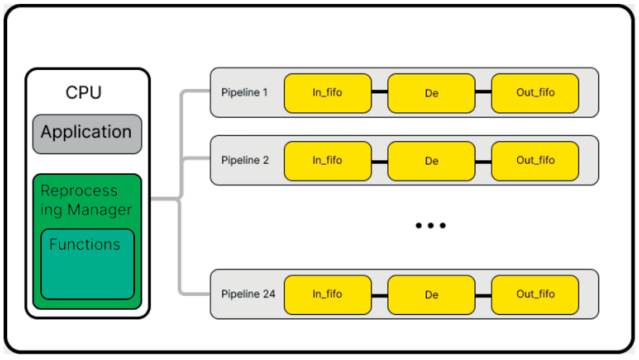

Рис. 2: Схема модифицированной имитационной модели.

Стандартные характеристики СПУ, которые также использовались и в исследовании:

- Число конвейеров: 24
- Число входных и выходных портов: по 24 (каждому конвейеру соответствует входной и выходной порты).
- Число ячеек конвейера: 1
- Размер заголовка пакета: 128 байт. То есть первые 128 байт пакета считаются его заголовком, а оставшаяся часть – его телом.

VI. Экспериментальное исследование

Эксперименты проводились следующим образом.

На конвейеры подавался набор пакетов, сохраненных в PCAP-файлах.

Модель останавливается после завершения работы последнего из конвейеров. Чем меньше тактов будет потрачено на обработку всех пакетов всеми конвейерами, тем лучше.

Также варьировался входной трафик: менялась доля пакетов, требующих повторной обработки и распределение пакетов между конвейерами.

Чем эффективнее распределение задач между конвейерами, тем быстрее модель закончит работу.

В исследовании присутствовало 2 типа пакетов. Первый тип обрабатывается за 7 тактов, второй – за 9. Лимит числа тактов был задан как 8. Соответственно, под повторную обработку будет попадать только второй тип.

Для каждой группы экспериментов сперва приводится описание методики проведения эксперимента, после чего идут таблицы с числом тактов работы сценария, а после – выводы из полученных результатов.

Первая группа экспериментов.

В первой группе экспериментов пакеты будут приходить на конвейеры равномерно, при этом будет меняться доля пакетов, требующих повторной обработки.

Всего пакетов 5000 штук, процент повторно обрабатываемых пакетов (то есть пакетов второго типа) составлял 10, 25, 50 % в разных тестах. В сценарии с выделенным конвейером было 25 конвейеров, 24 обычных и один для повторной обработки. В остальных – 24 обычных конвейера.

Результаты первой группы экспериментов (эксперименты проводились несколько раз, значения усреднялись) приведены в таблице 1.

ТАБЛИЦА I
СРЕДНЕЕ ЧИСЛО ТАКТОВ РАБОТЫ СЦЕНАРИЯ ПРИ 10% ПОВТОРНО ОБРАБАТЫВАЕМЫХ ПАКЕТОВ.

Round Robin	Send Same	Central Manager	Dedicated Pipeline	Без повторной обработки
2219	2222	2159	2158	2178

На 10% пакетов второго типа все алгоритмы показывают схожие результаты. При этом Central Manager и Dedicated Pipeline отработали даже лучше случая без повторной обработки. Это объясняется тем, что изначально пакеты распределены случайно, но равномерность их распределения по конвейерам все равно неидеальная. На каких-то конвейерах было больше пакетов, и они заканчивали позже. Перераспределение пакетов убрало эту неравномерность.

Увеличение доли сложных для обработки пакетов до 25% дало следующие результаты.

ТАБЛИЦА II
СРЕДНЕЕ ЧИСЛО ТАКТОВ РАБОТЫ СЦЕНАРИЯ ПРИ 25% ПОВТОРНО ОБРАБАТЫВАЕМЫХ ПАКЕТОВ.

Round Robin	Send Same	Central Manager	Dedicated Pipeline	Без повторной обработки
2392	2407	2293	3757	2293

Все сценарии работают чуть дольше, так как число пакетов то же, но доля тех из них, которые занимают 9, а не 7 тактов, выше.

Dedicated Pipeline показывает себя заметно хуже остальных. Выделенный конвейер перегружается. При этом в первом эксперименте этот подход давал лучшие результаты. Это говорит о том, что схема с выделенным конвейером хороша только если повторная обработка происходит относительно редко.

Лучше всех себя снова показывает Central Manager.

Увеличение доли “сложных” пакетов до 50% дало ожидаемые результаты.

ТАБЛИЦА III
СРЕДНЕЕ ЧИСЛО ТАКТОВ РАБОТЫ СЦЕНАРИЯ ПРИ 50% ПОВТОРНО ОБРАБАТЫВАЕМЫХ ПАКЕТОВ.

Round Robin	Send Same	Central Manager	Dedicated Pipeline	Без повторной обработки
2579	2588	2486	7579	2370

Вторая группа экспериментов.

Повторно обрабатываются 10% пакетов, но они приходят только на первые 2 порта.

ТАБЛИЦА IV
СРЕДНЕЕ ЧИСЛО ТАКТОВ РАБОТЫ СЦЕНАРИЯ ПРИ 10% ПОВТОРНО ОБРАБАТЫВАЕМЫХ ПАКЕТОВ ВО ВТОРОЙ ГРУППЕ ТЕСТОВ.

Round Robin	**Send Same**	**Central Manager**	**Dedicated Pipeline**	**Без повторной обработки**
4451	5183	4338	4338	4653

Разница небольшая, но заметна уже на 10% “сложных” пакетов.

VII. ИТОГИ ЭКСПЕРИМЕНТАЛЬНОГО ИССЛЕДОВАНИЯ

На основании проведенных экспериментов были сделаны следующие выводы об исследуемых алгоритмах:

- Round Robin хорошо показывает себя во всех рассмотренных ситуациях, но при этом нигде не является лучшим. Хорошо подойдет в случаях, когда оценка загруженность конвейеров не может быть использована.
- Send Same хорошо показывает себя в условиях равномерного распределения пакетов. Если же это не так, то он сильно перегружает порты, на которых повторно обрабатываемых пакетов больше.
- Central Manager. Всегда показывал себя одним из лучших среди исследованных алгоритмов. Точность работы не зависит ни от загруженности конвейеров, ни от того, насколько часто происходит повторная обработка. Единственный недостаток – более высокие требования для реализации. Помимо самого факта возможности анализа загрузки конвейеров, он еще и должен происходить достаточно быстро.
- Dedicated Pipeline. Выделенный конвейер очень хорошо показывает себя, если повторная обработка происходит редко. В противном случае конвейер может перегружаться и тормозить СПУ. Кроме того, это достаточно дорогостоящий, хотя и простой в реализации подход.

VIII. ЗАКЛЮЧЕНИЕ

В данной работе была рассмотрена задача повторной обработки пакетов на конвейерах сетевого процессорного устройства и получены следующие результаты:

- Предложен подход к организации повторной обработки пакетов, подход реализован в имитационной модели СПУ RuNPU.
- На основании обзора были отобраны алгоритмы выбора конвейера для повторной обработки заголовка пакета.
- Выбранные алгоритмы были реализованы в ИМ СПУ RuNPU.
- Проведено экспериментальное исследование выбранных алгоритмов, в результате которого выбран алгоритм, показывающий себя лучшим в различных условиях: Central Manager.

Поскольку схема Central Manager показала свою эффективность, в качестве будущих направлений работ можно выделить оптимизацию собираемых параметров для работы данной схемы.

СПИСОК ЛИТЕРАТУРЫ

[1] Смелянский Р.Л. Программно-конфигурируемые сети // Открытые системы. - 2012. - №9. - С. 15-26.

[2] Kornaros G. Multi-Core Embedded Systems // Embedded Computing Systems: Applications, Optimization, and Advanced Design. - 2010. - С. 399-459.

[3] Беззубцев С.О., Васин В.В., Волканов Д.Ю., Жайлауова Ш.Р., Мирошник В.А., Скобцова Ю.А., Смелянский Р.Л. Об одном подходе к построению сетевого процессорного устройства // Моделирование и анализ информационных систем. - 2019. - №26. - С. 39-62.

[4] Хританков А.С. Модели и алгоритмы распределения нагрузки. Алгоритмы на основе сетей СМО // Информационные технологии и вычислительные системы. - 2009. - №3. - С. 33-48.

[5] Sandeep S., Sarabjit S., Meenakshi S. Performance Analysis of Load Balancing Algorithms // International Journal of Civil and Environmental Engineering. - 2008. - №2. - С. 367-370.

Сравнение динамики кольчуги Буслаева и цепочки контуров с симметричным или несимметричным расположением узлов

Яшина М.В.
Кафедра высшей математики Московского автомобильно-дорожного государственного технического университета (МАДИ) и ФГУП «НАМИ» Государственного научного центра Российской Федерации
Москва, Россия
mv.yashina@madi.ru

Таташев А.Г.
Кафедра высшей математики Московского автомобильно-дорожного государственного технического университета (МАДИ) Кафедра математической кибернетики и информационных технологий Московский технический университет связи и информатики (МТУСИ)
Москва, Россия
a-tatashev@yandex.ru

Кутейников И.А.
Кафедра высшей математики Московский автомобильно-дорожный государственный технический университет (МАДИ); Кафедра математической кибернетики и информационных технологий Московский технический университет связи и информатики (МТУСИ)
Москва, Россия
ivankuteynikov09@gmail.com

***Abstract*—Рассматриваются динамические системы, принадлежащие классу сетей Буслаева. Сравнивается поведение вариантов двумерной сети - кольчуги и одномерной сети - цепочки контуров. Показаны, что кольчуга характеризуется более разнообразным поведением и процесс функционирования кольчуги может быть эквивалентен функционированию фрагментов кольчуги.**

Keywords—модели трафика, клеточные автоматы, случайные процессы с запретами, сети Буслаева

I. Введение

В модели Нагеля-Шрекенберга [1] частицы, соответствующие автотранспортным средствам, движутся по определенным правилам на решетке, в каждой ячейке которой не может одновременно находиться более одной частицы. Такие модели можно интерпретировать, как клеточные автоматы [2] или случайные процессы с запретами [3]. В простом варианте частицы движутся по правилу элементарного клеточного автомата 184 (Elementary Cellular Automaton, ECA 184) в нумерации С. Вольфрама. В соответствии с этим правилом частица перемещается на каждом шаге (в дискретный момент времени) на одну ячейку вперед при условии, что ячейка впереди свободна. Для такой модели получены аналитические результаты, например, в [4].

В двумерной модели BML [5] частицы движутся по правилу, аналогичному правилу ECA 184, в перпендикулярных направлениях на тороидальной решетке. Аналитические результаты для такой модели или ее обобщений и разновидностей получены, например, в [6]-[8].

А.П. Буслаев разработал класс динамических систем с целью создания сетевых моделей транспорта [9]. Этот класс назван контурными сетями или сетями Буслаева. В дискретном варианте такой сети частицы движутся по замкнутым последовательностям ячеек - контурам. Движение частиц осуществляется по правилу ECA 184 или в соответствии с кластерным движении [10], при котором находящиеся в соседних ячейках частицы образуют кластеры и перемещаются одновременно, рис.1. Кластерное движение соответствует правилу элементарного клеточного автомата 240. В непрерывном варианте контурной сети находящиеся в соседних ячейках частицы образуют кластеры и перемещаются одновременно. В непрерывном варианте по окружностям движутся отрезки постоянной длины по аналогии с дискретным вариантом называются кластерами. Контуры имеют общие точки, называемые узлами, при прохождении которых возникают задержки, обусловленные тем, что более одной частицы (одного кластера) не может одновременно проходить через узел. Если на текущем такте ячейка впереди частицы свободна, то перемещение реализуется. Если ячейка впереди в текущий момент занята, то частица остается на месте.

Если более одной частицы стремятся переместиться в одну и ту же ячейку, то перемещается только частица, выбираемая в соответствии заданным правилом разрешения конкуренции, которое может быть детерминированным и вероятностным.

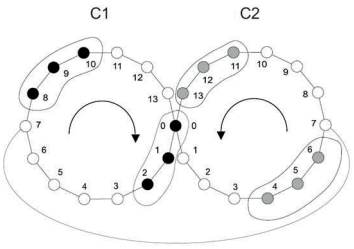

Рис. 1. Сеть с кластерным движением

Среднее расстояние, проходимое частицей в единицу времени, равное для сети с дискретным временем стационарной вероятности перемещения частицы на текущем такте, называется средней скоростью частицы. Если средняя скорость частиц одинакова для всех частиц, то она равна отношению среднего числа частиц, перемещающихся на текущем такте, к числу всех частиц. Множество возможных значений средних скоростей частиц при различных начальных состояний называется спектром средних скоростей.

При детерминированном правиле разрешения конкуренции после первого повторения одного состояния контурной сети периодически повторяется замкнутая последовательность состояний системы (реализуется

цикл), рис 2. Для исследовавшийся видов контурных сетей с вероятностным правилом конкуренции система спустя время с конечным математическим ожиданием движение становится детерминированным и реализуется цикл, на котором не возникает конкуренций одновременно подошедших к узлу частиц.

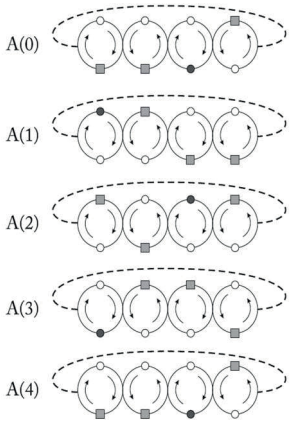

Рис. 2. Цикл

В [9] были введены одномерная контурная сеть, которая впоследствии стала называться бинарной замкнутой цепочкой контуров, и двумерная сеть с тороидальной системой контуров - кольчуга.

Аналитические результаты получены [11]-[15] в предположении, что на каждом контуре кольчуги имеются 4 ячейки и одна частица, Каждая ячейка является общей для двух соседних контуров - ячейка совмещена с соответствующим узлом, рис. 3. В [11], [12] предполагалось, что для любой пары соседних контуров тороидальной системы на одном из этих контуров частица движется в направлении против часовой стрелки, а на другом частица движется по часовой стрелке (сонаправленное движение). При размере кольчуги $M \times N$ сонаправленное движение реализуемо лишь в случае, если числа M и N - четные.

В [13] наряду с тороидальной кольчугой с сонаправленным движением и другими видами контурных сетей рассматривалась кольчуга с однонаправленным движением, при котором на всех контурах движение частиц направлено против часовой стрелки. В [11]-[13] предполагалось, конкуренции разрешаются по вероятностному правилу, например, по справедливому правилу, при котором участвующие в конкуренции частицы выигрывают равновероятно. В [14], [15] наряду с тороидальными кольчугами с сонаправленным или однонаправленным движением (замкнутыми кольчугами) рассматривались соответствующие кольчуги на прямоугольной системе контуров (открытые кольчуги).

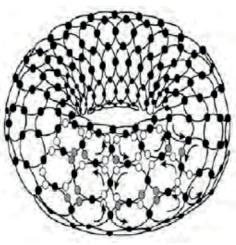

Рис. 3. Кольчуга

Замкнутая цепочка контуров представляет собой контурную сеть, в которой каждый контур имеет по одному узлу с одним из двух соседних контуров.

На каждом контуре бинарной цепочки контуров находятся две ячейки и одна частица, рис. 4. Узлы расположены между ячейками. Аналитические результаты для бинарных цепочек контуров с различными правилами разрешения конкуренции получены в [16].

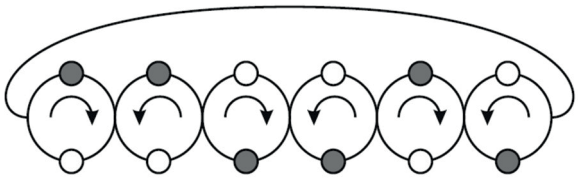

Рис. 4. Бинарная цепочка

В [17] исследовался вариант замкнутой цепочки контуров, который наиболее сходен с рассматривавшимися вариантами кольчуги. В этом варианте на контуре замкнутой цепочки находится четыре ячейки и одна частица. Каждый контур имеет общую ячейку - узел - с двумя соседними.

В [18] рассматривалась замкнутая цепочка, на каждом контуре которой находится четное число ячеек и одна частица, рис. 5. Узлы располагаются между ячейками и делят контуры пополам.

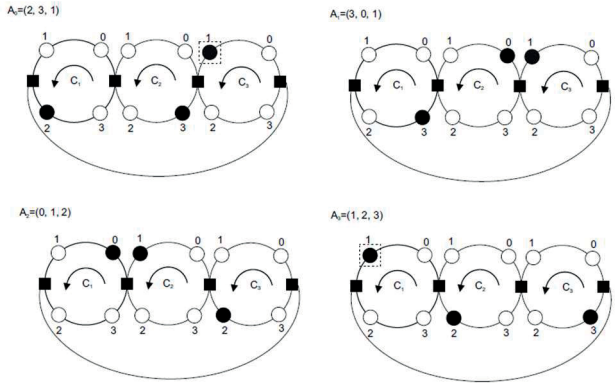

Рис. 5. Одночастичная замкнутая цепочка

В [19] рассматривалась дискретная замкнутая цепочка с кластерным движением на контуре.

В [20] рассматривалась дискретная открытая цепочка с кластерным движением.

II. Замкнутая цепочка с симметричным расположением узлов на контуре и замкнутая кольчуга с однонаправленным движением

A. Вариант замкнутой цепочки контуров

В [17] исследовалась замкнутая цепочка, содержащая N контуров $0,1,\ldots,N$. На каждом контуре имеются 4 ячейки 0, 1, 2 и 3, рис. 6. Контуры цепочки нумеруются слева направо по модулю N. Ячейки на контуре нумеруются по модулю 4 против часовой стрелки (в направлении движения частиц). Ячейка 0 контура i и ячейка 3 контура $i+1$ (сложение по модулю N) представляют собой одну и ту же ячейку - общую для

этих контуров. На каждом контуре имеется одна частица, занимающая одну из ячеек и перемещающаяся в моменты времени $t = 0,1,2,\ldots,$ если нет задержек. Каждая частица перемещается в текущий момент времени в следующую ячейку по направлению движения, если эта ячейка свободна и не возникает конкуренция с частицей соседнего контура, стремящейся перейти в ту же ячейку.

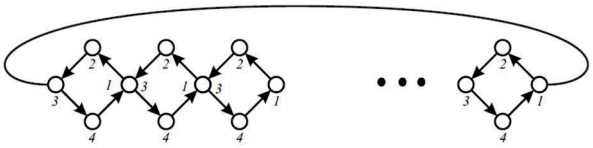

Рис. 6. Одночастичная замкнутая цепочка

При левоприоритетном правиле разрешения конкуренцию выигрывает частица, расположенная слева от общего для конкурирующих частиц узла, т.е. частица контура i (частица i) выигрывает конкуренцию у частицы $i+1,$ сложение по модулю N).

При справедливом правиле разрешения конкуренции каждая из конкурирующих частиц выигрывает конкуренцию равновероятно.

Состояние системы представляет собой вектор $x_0, x_1, \ldots, x_{N-1},$ где x_i - номер ячейки, в которой находится частица контура i, $i = 0,1,\ldots,N-1.$

Среднюю скорость частиц обозначаем через v.

Для рассматриваемой системы в [17] доказано следующая теорема для спектра средних скоростей при левоприоритетном правиле.

Для любого $k = 0,1,\ldots,\left[\frac{N}{3}\right]$ *(с помощью квадратных скобок обозначаем целую часть числа) существуют циклы со средней скоростью* $v = 1 - \frac{k}{N}$.

Пример 1. При $N = 3$ имеется цикл со скоростью 1 (свободное движение), которому соответствует последовательность переходов между состояниями $(0,0,0) \to (1,1,1) \to (2,2,2) \to (3,3,3) \to (0,0,0) \to \ldots,$ и цикл со средней скоростью $2/3$ $(0,0,0) \to (1,1,1) \to (2,2,2) \to (3,3,3) \to (0,0,0) \to \ldots$

При справедливым правиле разрешения конкуренции система за время с конечным ожиданием система попадает в состояние свободного движения.

B. Замкнутая кольчуга с однонаправленным движением

Кольчуга представляет собой сеть со структурой $M \times N$. Узлы совмещены с ячейками. Имеются mn контуров на торе. Элемент матрицы $A = \left(a_{ij}\right)_{M \times N},$ соответствует контуру $a_{ij},$ $i = 0,1,\ldots,M-1,$ $j = 0,1,\ldots,N-1.$ На каждом контуре располагаются 4 ячейки. Для контура a_{ij} ячейка с номером 0 совмещена с ячейкой 2 контура $a_{i,j-1}$ (вычитание по модулю N), ячейка с номером 1 совмещена с ячейкой 3 контура $a_{i-1,j}$ (вычитание по модулю M) и т.д. Правило движения на контуре аналогично правилу движения на контуре замкнутой цепочки, рассматривавшейся в п. *A*.

Рассмотрим два варианта системы.

Для рассматриваемой системы при справедливом правилом разрешения конкуренции верно следующее [11]. За время с конечным математическим ожиданием система попадает в состояние, принадлежащее следующему множеству состояний. Пусть числа $k_1,$ $i_1,\ldots,i_k,$ $j_1,\ldots,j_k,$ *удовлетворяют следующим условиям. Множество всех контуров разбивается на* k *непересекающихся множеств* $G_{i_1 j_1},\ldots,G_{i_k j_k}$ *образует блок контуров размера* 2×2. *Пусть* G *- дополнение к объединению этих множеств. Существует такое начальное состояние, что множества* $G_{i_1 j_1},\ldots,G_{i_k j_k}$ *находятся в состоянии коллапса (скорости частиц равны 0), а каждая частица контура из множества* G *находится в состоянии свободного движения.*

Рассмотрим следующий аналог левоприритетного правила разрешения конкуренции. При конкуренции частиц контуров $a_{i,j}$ и $a_{i+1,j}$ (сложение по модулю N) конкуренцию выигрывает частица контура a_{ij}; при конкуренции частиц контуров a_{ij} и $a_{i,j+1}$ (сложение по модулю N) конкуренцию выигрывает частица контура $a_{ij},$ $i = 0,1,\ldots,M-1,$ $j = 0,1,\ldots,N-1.$

Для рассматриваемой системы [14] с правилом разрешения конкуренции, аналогичным левоприритетному правилу, помимо циклов, реализующихся для системы со справедливым правилом, при любом $s = 0,1,\ldots,[N/3]$ *существуют такие циклы, что для каждой частицы средняя скорость равна* $v = (M-s)/M,$ *и для любого* $s = 0,1,\ldots,[M/3]$ *существуют такие циклы, что для каждой частицы равна* $v = (M-s)/M$. *На этих циклах на множествах контурах, соответствующих строкам или столбцам кольчуги, последовательности конфигураций частиц соответствуют последовательностям конфигураций частиц на замкнутой цепочке контуров с левоприритетным правилом.*

C. Вариант замкнутой цепочки контуров

Как следует из изложенного в пп. *A*, *B*, при аналоге левоприоритетного правила разрешения конкуренции существуют циклы, на которых частицы движутся так, что на контурах, соответствующих строках (столбцам) матрицы, движение частиц аналогично движению частиц на замкнутых цепочках с левоприоритетным правилом разрешения конкуренции.

При справедливом правиле разрешения конкуренции для замкнутой цепочки контуров не существует циклов со скоростью меньше 1. В соответствии с этим при справедливом правиле разрешении конкуренции для кольчуги с однонаправленным движением не существует циклов со средней скоростью меньше 1.

Для обоих правил разрешения конкуренции существуют циклы, на которых есть блоки размера $2\times 2,$

находящиеся в состоянии коллапса. Наличие таких циклов требует двумерной структуры цепи и подобные циклы не реализуемы для цепочки контуров.

III. Открытая цепочка контуров с симметричным расположением узлов и открытая кольчуга с однонаправленным движением

Рассмотрим вариант открытой цепочки контуров, отличающейся от рассмотренного в п. *A* варианта замкнутой цепочки тем, что крайний левый и крайний правый контур имеют только по одному соседнему. как следует из результатов [20], *для открытой цепочки с левоприоритетным правилом (как и для цепочки со справедливым правилом) не существует циклов со скоростью меньше 1.*

Соответственно, для открытой кольчуги на прямоугольной структуре возможны только циклы, на которых могут существовать блоки размера 2×2, *находящиеся в состоянии коллапса, а частицы, не принадлежащие этим блокам, находятся в состоянии свободного движения.*

IV. Замкнутая цепочка с несимметричным расположением узлов на контуре и замкнутая кольчуга с сонаправленным движением

Рассмотрим кольчугу с сонаправленным движением, для которых на контурах, соответствующих элементам матрица a_{ij}, частица движется против часовой стрелки, если число $i+j$ нечетное, и - по часовой стрелке, если это число четное, рис. 7.

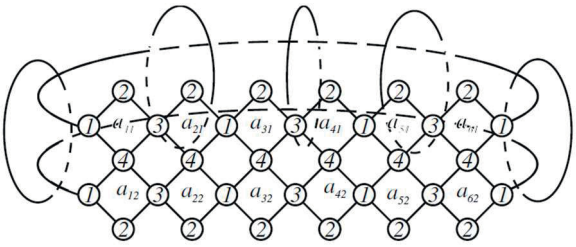

Рис. 7. Кольчуга на торе с сонаправленным движением

Пусть $M=2m,\ N=2n.$ Введем понятие "диагонали". Пусть d - наибольший общий делитель чисел m и $n.$ Каждая диагональ содержит $4mn/d$ контуров: $D_{sij}(0), D_{sij}(1),\ldots, D_{sij}((4mn/k)-1)$ контуров. Элементом $D_{sij}(0)$ является контур (i,j). Пусть $i+j$ - четное число. Тогда элементы $D_{sij}(1)\ldots, D_{sij}((4mn/d)-1)$ задаются так, чтобы выполнялось следующее. Если на контуре $D_{sij}(0)=(i,j)$ частица находится в вершине с номером s, т.е. $x_{ij}=s,$ а на контуре D_{sik} частица находится в ячейке $s-1$ (вычитание по модулю 4), если число k четное, и частица находится в ячейке s, если число k нечетное, $0\le k\le 4mn/d,$ то все частицы, находящиеся на контурах диагонали, не будут перемещаться в текущий момент и в будущем (система находится в состоянии *коллапса*). Пусть $i+j$ - нечетное число. Тогда элементы $D_{sij}(1)\ldots, D_{sij}(4mn/d)-1)$ задаются так, чтобы выполнялось следующее. Если на контуре $D_{sij}(0)=a_{ij}$ частица находится в вершине с номером s, т.е. $D_{sij}(0)=a_{ij}=s,$ и на контуре D_{sik} частица находится в ячейке $s+1$ (сложение по модулю 4), если число k четное, и частица находится в ячейке s, если число k нечетное, $0\le k\le 4mn/d,$ то все частицы будут находиться в состоянии коллапса.

Пусть $i+j$ - четное число. Перечислим элементы, которые содержат диагонали D_{sij}, $s=0,1,2,3.$ Сложение и вычитание в первом индексе понимается по модулю M, а сложение и вычитание во втором индексе понимается по модулю $N.$

Диагональ D_{0ij} содержит элементы

$$D_{0ij}(2k)=(i-k,j-k), D_{0ij}(2k+1)=(i-k-1,j-k).$$

Диагональ D_{1ij} состоит из элементов

$$D_{1ij}(2k)=(i+k,j-k), D_{1ij}(2k+1)=(i+k,j-k-1).$$

В диагонали D_{2ij} содержатся элементы

$$D_{2ij}(2k)=(i+k,j+k), D_{2ij}(2k+1)=(i+k+1,j+k).$$

Диагональ D_{3ij} содержит элементы

$$D_{3ij}(2k)=(i-k,j+k), D_{3ij}(2k+1)=(i-k,j+k+1).$$

Перечислим элементы, которые содержат диагонали в предположении, что число $i+j$ нечетное.

Тогда диагональ D_{0ij} состоит из элементов

$$D_{0ij}(2k)=(i+k,j-k),\ D_{0ij}(2k+1)=(i+k+1,j-k),$$

$$k=0,1,\ldots,\frac{4mn}{d}-1,\qquad i=0,1,\ldots,M-1,\qquad j=0,1,\ldots,N-1$$

(вычитание в первой координате контура по модулю M, вычитание по второй координате контура по модулю N).

Диагональ D_{1ij} содержит элементы

$$D_{1ij}(2k)=(i+k,j+k), D_{1ij}(2k+1)=(i+k,j+k+1).$$

Диагональ D_{2ij} содержит элементы

$$D_{2ij}(2k)=(i-k,j+k), D_{2ij}(2k+1)=(i-k-1,j+k).$$

Диагональ D_{3ij} содержит элементы

$$D_{3ij}(2k)=(i-k,j-k), D_{3ij}(2k+1)=(i-k,j-k-1).$$

Для такой системы верно следующее [14], [15]. *Пусть множество контуров кольчуги размера* $(2m)\times(2n)$ *с сонаправленным движением разбито на* d *непересекающихся диагоналей* $D_{000}, D_{002},\ldots,\ D_{00(2d-2)}.$ *Тогда для любого набора целых чисел* $s_0, s_1,\ldots, s_{d-1},$ *удовлетворяющего условию* $0\le s_i\le mn/d,\quad i=1,\ldots,d,$ *существуют циклы такие, что на любом контуре*

диагонали $D_{00(2l)}$ *средняя скорость частицы равна* $s_l d/(mn),\ i=1,\ldots,d.$

Если диагональ рассматривать в предположении, что остальные контуры кольчуги отсутствуют, то ее можно считать замкнутой цепочкой контуров с несимметричным расположением узлов на контуре, рис. 8.

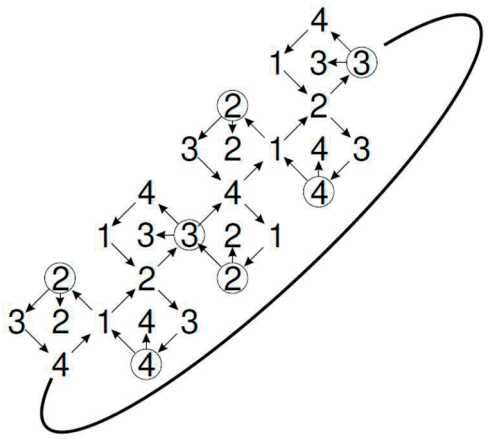

Рис. 8. Несимметричная замкнутая цепочка, соответствующая диагонали замкнутой кольчуги с сонаправленным движением

Таким образом, движение на циклах кольчуги можно представить как движение на параллельных замкнутых цепочках такого вида.

V. Открытая цепочка с несимметричным расположением узлов на контуре и открытая кольчуга с сонаправленным движением

Как и для соответствующей открытой цепочки с симметричным расположением узлом, *для открытой цепочки с несимметричным расположением узлов, аналогичной замкнутой цепочке, соответствующей диагонали кольчуги, устанавливается свободное движение.*

В соответствии с этим *для открытой кольчуга с сонаправленным движением все циклы соответствуют свободному движению.*

VI. Заключение

Рассмотрены различные варианты двумерной сети Буслаева - кольчуги и соответствующие варианты более простой сети - цепочки контуров. Отмечены аналогии между поведением одномерной и двумерной сети. На циклах двумерной сети поведение ее фрагментов может соответствовать поведению одномерных сетей. Однако поведение двумерной сети более разнообразно и для нее могут существовать циклы, для которых нет аналогий в одномерных сетях.

Список литературы

[1] Nagel K., Schreckenberg M. A cellular automation models for freeway traffic. J. Phys. I. France 2, 1992, pp. 2221-2229. DOI: 10.1051/jp1:1992277

[2] Wolfram S. Statistical mechanics of cellular automata. Reviews of Modern Physics, 55, 3, 601, 1983. DOI:10.1103/RevModPhys.55.601

[3] Spitzer F. Interaction of Markov Processes // In Random Walks. Brownian Motion, Interacting Particle Systems. Springer, 1991, pp.66-110.

[4] Бланк М.Л., “Точный анализ динамических систем, возникающих в модлях транспортных потоков”, УМН, 55:3(333) (2000), 167--168. DOI:10.4213/rm295

[5] Biham O., Middleton A.A., Levine D. (1992). Self-organization and a dynamical transition in traffic-flow models. Physical Review A, 46(10), R6124.\\ DOI:10.1103/PhysRevA.46.R6124

[6] Linesch N.J., D’Souza R.M. (2008). Periodic states, local effects and coexistence in the BML traffic jam model. Physica A: Statistical Mechanics and its Applications, 387(24), 6170--6176.\\ DOI:10.1016/j.physa.2008.06.052

[7] Austin T., Benjamini I. For what number must self organization occur in the Biham-Middleton-Levine traffic model from any possible starting configuration? arXiv preprint math/0607759, 2006.

[8] Yashina M.V., Tatashev A.G. Loading conditions for self-organization in the BML model with stochastic direction choice. Mathematical Methods in the Applied Sciences. First published: 26 August 2024 https://doi.org/10.1002/mma.10276

[9] Kozlov V.V., Buslaev A.P., Tatashev A.G. On synergy of totally connected flows on chainmails. Proceedings of the 13th International Conference on Computational and Mathematical Methods in Sciences and Engineering, CMMSE~2013, Almeria, Spain, 2013, vol.~3, pp.~861--874 (2014).

[10] Bugaev A.S., Buslaev A.P., Kozlov V.V., Yashina M.V. (2011, October). Distributed problems of monitoring and modern approaches to traffic modeling. In 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC) pp.~477-481). IEEE. DOI: 10.1109/ITSC.2011.6082805

[11] Buslaev A.P., Tatashev A.G., Yashina M.V. Qualitative properties of dynamical system on toroidal chainmail // AIP Conference Proceedings, vol. 1558, pp. 1144-1147. American Institute of Physics, 2013. DOI:10.1063/1.4825710

[12] Kozlov V.V., Buslaev A.P., Tatashev A.G., Yashina M.V. Monotonic walks of particles on a chainmail and coloured matrices // In Proceedings of the 14th International Conference on Computational and Mathematical Methods in Science and Engineering, 2014, CMSSE~2014, vol.~3, pp.~801--805.

[13] А.С. Бугаев, А.П. Буслаев, В.В. Козлов, А.Г. Таташев, М.В. Яшина, Обобщенная транспортно-логистическая модель как класс динамических систем, Матем. моделирование, 27:12 (2015), 65–87.

[14] Fomina M.J., Tolkachov A.G., Tatashev D.A., Yashina M.V. (2018, September). Cellular Automata as Traffic Models and Spectrum of Two-Dimensional Contour Networks—Open Chainmails. In 2018 IEEE International Conference" Quality Management, Transport and Information Security, Information Technologies"(IT&QM&IS) (pp. 435-440). IEEE. DOI: 10.1109/ITMQIS.2018.8525079

[15] Kuteynikov I.A., Tatashev A.G., Yashina M.V. (2019). On properties of closed/open two-dimensional network-chainmail with different rules of particle movement. Periodicals of Engineering and Natural Sciences, 7(1), 447-457. DOI:10.21533/pen.v7i1.340.

[16] Kozlov V.V., Buslaev A.P., Tatashev, A.G. (2015). Monotonic walks on a necklace and a coloured dynamic vector. International Journal of Computer Mathematics, 92(9), 1910--1920. DOI:10.1080/00207160.2014.915964

[17] Kozlov V.V., Buslaev A.P., Tatashev A.G. Behavior of pendulums on a regular polygon // Journal of Communication and Computer., 2014,. 11, vol. 1, pp. 30-38.

[18] Мышкис П.А., Таташев А.Г., Яшина М.В. Дискретная замкнутая одночастичная цепочка контуров // Прикладная дискретная математика. 2021, №.52. С.114-125.

[19] Бугаев А.С., Таташев А.Г., Яшина М.В. Спектр непрерывной замкнутой симметричной цепочки с произвольным числом контуров. Математическое моделирование, 33.4 (2021): 21-44.

[20] Yashina M., Tatashev A. (2019, November). Discrete open Buslaev chain with heterogenous loading. In 2019 7th International Conference on Control, Mechatronics and Automation (ICCMA) (pp.283-288). IEEE. DOI: 10.1109/ICCMA46720.2019.8988654

Защита контейнеризованных сред. Анализ специфики уязвимостей, направленных на нарушение границ контейнера (Container Escape Attacks) и разработка методов их определения с использованием собственных реализаций Container Runtime Interface (CRI) и с Extended Berkeley Packet filter (EBPF) для ядра Linux

Фирсов Сергей Владимирович
аспирант 2-го курса кафедры «БТК (факультет «Кибернетика и информационная безопасность»)
Московский Технический Университет Связи и Информатики
Москва, Россия
sergey@firsov.tech,

Аннотация—Контейнризированные среды и системы их управления заняли свою нишу в разворачивании, управлении, и разработке программного обеспечения. Угрозы и уязвимости, которым подвержены такие среды имеют свою специфику в силу архитектуры построения систем. В статье проанализированы типичные угрозы и факторы, способствующие появлению таких угроз для контейнириризированных сред, сделан обзор архитектуры компонентов выполнения контейнеров и их оркестраторов, способы построения систем предотвращения вторжений в статическом и динамическом исполнении.

Ключевые слова—контейнер, контейнеризированная среда, обнаружение вторжений, Intrusion Detection Systen (IDS), container escape attacks, Extended Berkley Packet Filter(eBPF), Docker, Kubernetes, Container Runtime Interface(CRI)

I. ВВЕДЕНИЕ

Поскольку контейнеризация становится все более популярной для развертывания приложений, важно обеспечить безопасность контейнеризированных сред от существующих и потенциально новых угроз. Контейнеризация предоставляет множество преимуществ для развертывания приложений, включая переносимость, масштабируемость и последовательность. Однако она также вносит новые вызовы в области безопасности, которые необходимо решить, чтобы защититься от потенциальных угроз.

Контейнеризация — это техника разработки программного обеспечения, которая позволяет разработчикам упаковывать приложение и его зависимости в единицу, называемую контейнером. Контейнеры изолированы друг от друга и от операционной системы хоста, что делает их более переносимыми и безопасными, чем традиционная виртуальная машина. При контейнеризации отличается подход к виртуализации: контейнеры работают поверх хост-машины и используют ее ресурсы, однако, вместо виртуализации базового оборудования, они виртуализируют операционную систему хоста.

Образы контейнеров могут быть общими и повторно используемыми, что облегчает быстрое и последовательное развертывание приложений [1].

Контейнеризация революционизирует разработку и развертывание программного обеспечения, инкапсулируя приложения в легких, изолированных средах. Обеспечивая эффективное использование ресурсов и масштабируемость, контейнеры приобрели значительную популярность. По данным отчетов отрасли на 2023 год, уровень принятия контейнеризации среди организаций по всему миру составляет впечатляющие 67% [1].

Docker, на сегодняшний день, является одной из лидирующих платформа для контейнеризации. Однако, несмотря на виртуализацию и изоляцию контейнеров, эта платформа (как частный случай) и подобные платформы подвержены определенным типам специфичных угроз.

II. КОМПОНЕНТЫ ИНФРАСТРУКТУРЫ DOCKER ПОДВЕРЖЕННЫЕ АТАКАМ ИЗ КОНТЕЙНЕРОВ

Рассмотрев высокоуровнево компоненты инфраструктуры Docker, можно выделить как минимум 3 фактора обуславливающих возможность атак на такую систему и способы использования таких факторов [2].

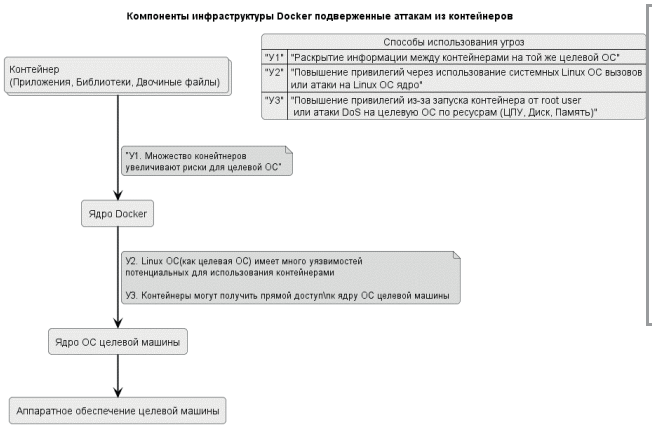

Рис. 1. – Компоненты инфраструктуры Docker подверженные атакам из контейнеров

ТАБЛИЦА I. ОПИСАНИЕ ФАКТОРОВ УГРОЗ ОБУСЛАВЛИВАЮЩИХ ВОЗМОЖНОСТЬ АТАК НА ИНФРАСТРУКТУРУ DOCKER

Фактор угрозы	**Описание**	**Способ использования**
У1	Эффективная архитектура с использованием нескольких контейнеров на хосте и совместное использование процессора, памяти, сети, идентификаторов пользователей и других ресурсов от одного ядра также является уязвимостью и риском для безопасности. Это связано с тем, что, если ядро атакуется, злоумышленники могут получить привилегии root на хосте и оттуда атаковать другие контейнеры и всю систему [3]. Некоторые уязвимости, сообщенные в CVE: CVE-2020-15257, CVE-2021-41103, CVE-2021-4154 и CVE-2022-31030	Раскрытие информации между контейнерами запущенными на той же целевой ОС.
У2	Контейнер полагается на ядро Linux, и существует множество уязвимостей, связанных с ядром Linux, которые могут повлиять на безопасность контейнера, например, уязвимость в модуле runc, позволяющая злонамеренному контейнеру получить привилегии root на хостовой машине [4]. На сегодняшний день в MITRE Details перечислено около 3000 уязвимостей в Linux [5]. Однако не было проведено достаточно глубоких исследований по количеству и типам уязвимостей в Linux, которые непосредственно влияют на контейнеры. Некотосрые уязвимости контейнеров на основе ядра были недавно обнаружены и сообщены CVE-2018-16884, CVE-2021-3669 и CVE-2021-44733.	Повышение привилегий через использование (недокументированное) системных вызовов целевой ОС или прямые атаки на нее.
У3	Одной из особенностей контейнера является его прямое подключение к ядру хоста, в отличие от виртуальной машины (ВМ), для которой требуется обход ядра ВМ и гипервизора приложением. В результате злоумышленнику легче получить доступ к ядру хоста, если он может проникнуть в приложение внутри контейнера на хосте [6]. Эта уязвимость была использована в CVE-2018-16884, CVE-2021-4154 и CVE-2022-0811.	Повышение привилегий через использование факта запуска контейнера от root пользователя или атаки направленные на отказ в обслуживании (DoS) по ресурсам (ЦПУ, память, диск и тд).

III. АТАКИ НА ПРЕОДОЛЕНИЕ ГРАНИЦ КОНТЕЙНЕРА (CONTAINER ESCAPE ATTACKS)

"Container escape attacks"—это тип атак на безопасность, при которых злоумышленник получает несанкционированный доступ из контейнера к базовой системе хоста или к другим контейнерам, работающим на том же хосте. Это позволяет злоумышленнику выйти за пределы контейнера и потенциально скомпрометировать всю систему хоста или другие контейнеры. Особенности таких атак в том, что они включают в себя использование уязвимостей в платформе контейнеризации времени выполнения, манипуляцию ресурсами ядра или использование неправильных настроек в среде контейнера.

Некоторые недавние и критические уязвимости Common Vulnerabilities and Exposures (CVE), связанные с атаками на "container escape", включают:

A. CVE-2020-15257, CVE-2021-41103. Фактор угрозы У1. Повышение привилегий и доступ к другим контейнерам.

CVE-2020-15257 [7]

Containerd—это стандартный компонент среды выполнения контейнеров, доступный в качестве сервиса для Linux и Windows, с акцентом на простоту, надежность и переносимость. В версиях containerd до 1.3.9 и 1.4.3 API контейнера containerd-shim публично доступен другим контейнерам в том же пространстве имен сети. Контроль доступа основан только на том, что подключающийся процесс имеет эффективный UID 0. Это позволяло вредоносным контейнерам, работающим в том же пространстве имен сети, что и shim, с эффективным UID 0, но в остальном с ограниченными привилегиями, запускать новые процессы с повышенными привилегиями.

CVE-2021-41103 [8]

Была обнаружена ошибка в containerd, когда корневые каталоги контейнера и некоторые плагины имели недостаточно ограниченные разрешения, позволяя в противном случае не привилегированным пользователям Linux обходить содержимое каталогов и выполнять программы. Когда в контейнерах были включены исполняемые программы с расширенными битами разрешения (например, setuid), не привилегированные пользователи Linux могли обнаруживать и выполнять эти программы. Когда UID не привилегированного пользователя Linux на хосте совпадал с владельцем файла или группой внутри контейнера, не привилегированный пользователь Linux на хосте мог обнаруживать, читать и изменять эти файлы.

B. CVE-2022-31030, CVE-2022-1708, CVE-2021-3669. Фактор угрозы У2. Отказ в обслуживании.

CVE-2022-31030 [9]

Была обнаружена ошибка в реализации Container Runtime Interface (CRI) в containerd, когда программы внутри контейнера могут приводить к неконтролируемому потреблению памяти containerd во время вызова API ExecSync. Это может привести к исчерпанию всей доступной памяти на компьютере containerd, что приведет к отказу в обслуживании других легитимных рабочих нагрузок. ExecSync может использоваться при выполнении проверок (probes) системой Kubernetes или при выполнении процессов с помощью средства "exec".

CVE-2022-1708 [9]

Запрос ExecSync выполняет команду в контейнере и возвращает вывод в Kubelet. Он используется для проверок готовности и жизнеспособности внутри пода. CRI-O запускает команды ExecSync через conmon. conmon - программа мониторинга и коммуникации между менеджером контейнеров (например, CRI-O) и OCI средой выполнения (например, runc) для отдельного контейнера.CRI-O запрашивает conmon запустить процесс, а conmon записывает вывод на диск. Затем CRI-O читает вывод и возвращает его в Kubelet.

Если вывод команды достаточно велик, есть возможность исчерпать память (или использование диска) узла. Ниже приведен пример yaml-файла развертывания, который будет выводить около 8 ГБ символов ‘A’, которые будут записаны на диск conmon и прочитаны CRI-O.

```
containers:
- name: nginx
  image: nginx:1.14.2
  lifecycle:
    postStart:
      exec:
        command: ["/bin/sh", "-c", "seq 1 50000000`; do echo -n 'aaaaaaaaaaaaaaaa'; done"]
```

CVE-2021-3669 [9]

Чтение из системного каталога /proc/sysvipc/shm отвечающего за разделяемую память неправильно масштабируется в соответствии с числом сегментов такой памяти, что приводит к исчерпанию ресурсов и отказу в обслуживании.

C. CVE-2022-0811 [10]. Фактор угрозы У3. Доступ к ядру хоста, повышение привилегий.

Kubernetes - это открытая платформа для оркестрации контейнеров, которая автоматизирует развертывание, масштабирование и управление контейнеризированными приложениями. Kubernetes использует менеджер контейнеров, такой как CRI-O или Docker, для безопасного разделения ядра и ресурсов каждого узла с различными контейнеризированными приложениями, работающими на нем. Ядро Linux принимает параметры времени выполнения, которые управляют его поведением. Некоторые параметры пространственно ограничены и могут быть установлены в одном контейнере, не затрагивая всю систему. Kubernetes и менеджеры контейнеров, которые он управляет, позволяют подам обновлять эти "безопасные" настройки ядра, блокируя доступ к другим [11].

Дефект, внедренный в версии CRI-O 1.19, использует утилиту pinns для установки опций ядра для пода. Pinns обычно вызывается так:

“pinns -s kernel_parameter1=value1+kernel_parameter2=value2”.

В связи с добавлением поддержки sysctl в версии 1.19, pinns теперь просто устанавливает любые переданные ему параметры ядра без проверки. В результате любой пользователь с правами на развертывание пода в кластере Kubernetes, который использует менеджер контейнеров CRI-O, может злоупотребить параметром ядра "kernel.core_pattern". В частности, этот параметр используется при задании формата имени файла дампа при падении. Использовав специфику команды pinns, можно указать командный файл (mailicious.sh) который выполнится при сохранении файла дампа ядра [12]. Далее несложной командой вызыв дамп: “kill -SIGSEGV <PID>” можно выполнить произвольный файл с root правами на целевой машине, вне границ контейнера [11].

```
> cat ./sysctl-set.yaml
apiVersion: v1
kind: Pod
metadata:
  name: sysctl-set
spec:
  securityContext:
   sysctls:
   - name: kernel.shm_rmid_forced
     value: "1+kernel.core_pattern=|/var/lib/containers/storage/overlay/3e6/diff/malicious.sh"
  containers:
  - name: alpine
    image: alpine:latest
    command: ["tail", "-f", "/dev/null"]
> kubectl create -f ./sysctl-set.yaml
pod/sysctl-set created
```

IV. Компоненты и интерфейсы систем контейнеризации. Механизм реализации системы обнаружения container escape атак основанный на анализе параметров.

Проанализировав распространённые угрозы и риски (CVE - Common Vulnerabilities and Exposures) для упомянутых выше компонентов containerd, cri-o, можно увидеть, что данные компоненты имеют списки

уязвимостей исчисляемые десятками, в зависимости от базы данных угроз и типа запроса.

Например, использовав базу данных угроз Vulners.com для containerd находится 36 известных угроз и рисков, для cri-o – 9. Меньшее число объясняется меньшей распространенностью компонента.

ТАБЛИЦА II. ЧИСЛО ИЗВЕСТНЫХ УГРОЗ ПО ТИПУ МЕНЕЛЖЕРА КОНТЕЙНЕРОВ

Компонент	Адрес запроса	Число угроз и рисков
containerd	https://vulners.com/search?query=affectedSoftware.name:%22github.com/containerd/containerd	36
cri-o	https://vulners.com/search?query=affectedSoftware.cpeName:%22kubernetes:cri-o%22	9

Для более детального рассмотрения обозначенных компонентов в инфраструктуре систем контейнеризации и оркестрации, представим их на диаграмме:

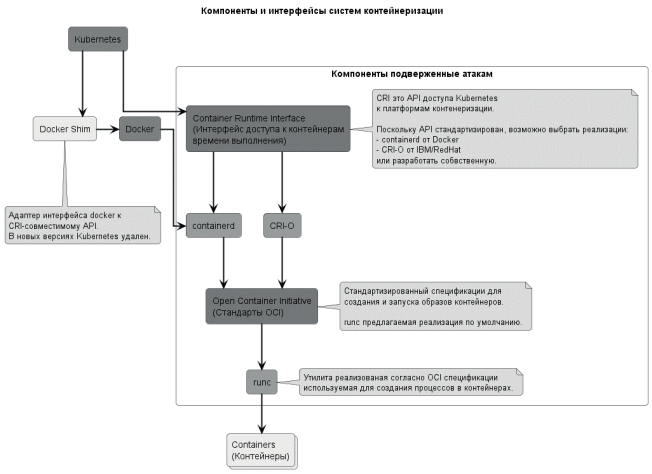

Рис. 2. Компоненты и интерфейсы систем контейнеризации [13]

Взаимодействие с платформами контейризации стандартизировано и специфировано согласно Common Runtime Interface (CRI) и Open Container Initiative (OCI). Проанализировав описанные интерфейсы, можно увидеть, что параметры запуска и инициализации контейнеров, которые часто являются целями атаки и компрометаций, доступны для программного анализа, не требуют reverse engineering стороннего кода или длительных исследований нахождения способов интеграции и перехвата вызовов [14],[15].

```
message ContainerConfig {
    //    Метаданные    контейнера.    Информация
однозначно определяет и идентифицирует контейнер.
ContainerMetadata metadata = 1 ;
    // Образ для использования.
    ImageSpec image = 2;
    //    Команда    запуска    (например    точка
входа(entrypoint) для docker)
    repeated string command = 3;
    // Аргументы команды (например команда(comand)
для docker)
    repeated string args = 4;
    // Текущая рабочая директория.
    string working_dir = 5;
    // Список переменных окружения.
    repeated KeyValue envs = 6;
    // Точки и тома для монтирования в контейнер.
    repeated Mount mounts = 7;
    ...
    // Конфигурации специфичные для Linux и
Windows контейнеров.
    LinuxContainerConfig linux = 15;
    WindowsContainerConfig windows = 16;
    ...
}

message CreateContainerRequest {
    string pod_sandbox_id = 1;
    // Конфигурация контейнера (описана выше)
    ContainerConfig config = 2;
    PodSandboxConfig sandbox_config = 3;
}


message ExecSyncRequest {
    // ID контейнера.
    string container_id = 1;
    // Команда запуска.
    repeated string cmd = 2;
    // Время в секундах отведенное для завершения
команды. По-умолчанию 0.
    int64 timeout = 3;
}
```

```
service RuntimeService {
  // Выполняет команду в контейнере синхронно.
  rpc ExecSync(ExecSyncRequest) returns
(ExecSyncResponse) {}
    ...
}
```

В параметрах сообщения ContainerConfig доступны команда запуска, аргументы, монтируемые диски, конфигурация контейнера для Windows/Linux.

В интерфейсе RuntimeService и его методе ExecSyncRequest - запрос на запускаемую команду. Доступ к этим данным, позволяет реализовать механизмы предотвращения атак, имеющих те же вектора что и CVE-2022-31030, CVE-2022-1708, в зависимости от использования уязвимости CVE-2021-3669, CVE-2022-0811.

Разработав собственные реализации, менеджеры контейнеров согласно спецификациям CRI/OCI, выступающие в роли декораторов (паттерн проектирования Decorator), позволит получить интегрированные точки для перехвата вызовов запуска контейнеров и команд в них независимо от того, будет ли контейнер использоваться оркестратором Kubernetes или системой контейнеризации Docker. Эти интегрированные точки перехвата параметров будут являться достаточными для разработки методов обнаружения и предотвращения типов атак, описанных выше.

V. МЕХАНИЗМЫ РЕАЛИЗАЦИИ СИСТЕМЫ ОБНАРУЖЕНИЯ CONTAINER ESCAPE АТАК ВО ВРЕМЯ ВЫПОЛНЕНИЯ ИСПОЛЬЗУЯ EXTENDED BERKELEY PACKET FILTER.

Тем не менее, для реализации методов обнаружения и предотвращения атак направленных на ядро целевой ОС и ее системные вызовы, методы предложенный выше не подходит и требуется более низкоуровневых подход который позволит анализировать весь поток системных вызовов и нахождения сигнатур в их параметрах.

Рассмотрим вариант реализации механизма определения уязвимости CVE-2024-23651 (очередная уязвимости при эксплуатации которой удается нарушить границы контейнера). Уязвимость возникает из-за гонки времени проверки/использования (TimeOfCheck/TimeOfUse-TOCTOU) при монтировании кэш-тома во время сборки контейнера. Buildkit предлагает использование кэш-томов с помощью директивы RUN --mount=type=cache в Dockerfile, что позволяет монтировать постоянный каталог в контролируемом месте во время создания образа Docker. Эта функциональность предназначена для улучшения производительности инструментов, таких как менеджеры пакетов, путем сохранения постоянного кэша между сборками.

Возможно указать путь источника внутри уже примонтированной кеш директории, и было выявлено, что проверка этого пути источника существование пути и является ли он директорией приводит к состоянию гонки. Параллельно выполняющийся этап сборки, который уже монтирует целевой каталог кэша, может заменить путь источника символической ссылкой между проверкой и использованием (т.е. источником системного вызова mount), что приводит к неправильному монтированию произвольного каталога целевой ОС в файловую систему контейнера [16].

Для выявления манипуляций на уровне файловой системы, необходимо уметь отслеживать системный вызов: syscalls/sys_mount [17]. Реализовывать такой функционал позволяет Extended Berkeley Packet Filter.

eBPF (extended Berkeley Packet Filter) - это мощный механизм в ядре Linux, который позволяет создавать и встраивать программы для анализа и фильтрации событий в ядре операционной системы. Он предоставляет возможность реализовывать широкий спектр функций, включая мониторинг системных вызовов, сетевую активность и поведение контейнеров, что делает его эффективным инструментом для обеспечения безопасности и мониторинга в контейнеризованных окружениях.

Программы eBPF запускаются по событиям и выполняются, когда ядро или приложение проходит определенную точку подключения. Предопределенные точки подключения включают системные вызовы, вход/выход из функций, трассировочные точки ядра, сетевые события и многие другие [18].

Описав на eBPF программу для мониторинга указанных системных функций:

```
struct trace_event_raw_sys_mount {
        unsigned short common_type;
        unsigned char common_flags;
        unsigned char common_preempt_count;
        int common_pid;
        int __syscall_nr;
        const char *dev_name;
        const char *dir_name;
        const char *type;
        unsigned long flags;
        const void *data;
};

struct event {
        __u32 pid;
        __u8 dev_name[255];
        __u8 dir_name[255];
        __u8 fs_type[16];
        __u64 timestamp;
};

const struct event *unusedevent __attribute__((unused));

struct {
        __uint(type, BPF_MAP_TYPE_RINGBUF);
        __uint(max_entries, 1 << 24);
} events SEC(".maps");

SEC("tracepoint/syscalls/sys_mount")
int trace_mount(struct trace_event_raw_sys_mount *ctx)
{
        if (!ctx) return 0;

        const char *dev_name = (const char *)ctx->dev_name;
        const char *dir_name = (const char *)ctx->dir_name;
        const char *fs_type = (const char *)ctx->type;

        struct event *event;
        event = bpf_ringbuf_reserve(&events, sizeof(struct event), 0);
        if (!event) return 0;

        u64 id = bpf_get_current_pid_tgid();
        u32 pid = id;
        event->pid = pid;
        u64 timestamp = bpf_ktime_get_ns();
        event->timestamp = timestamp;
        bpf_probe_read_user_str(&event->dev_name, sizeof(event->dev_name), dev_name);
        bpf_probe_read_user_str(&event->dir_name, sizeof(event->dir_name), dir_name);
```

```
    bpf_probe_read_user_str(&event->fs_type, sizeof(event->fs_type), fs_type);
    bpf_ringbuf_submit(event, 0);
    return 0;
}
```

и сгенерировав клиентскй код на необходимом языке, например для Go используя bpf2go, можно реализовать код чтения событий доступа к символическим ссылкам [19]:

```
type Event struct {
    Pid           uint32
    DevName       string
    DirName       string
    FsType        string
    TimestampMicros uint64
}

type ExploitDetectionHandler func(*Event)

func Listen(handler ExploitDetectionHandler) {
    objs := bpfObjects{}
    if err := loadBpfObjects(&objs, nil); err != nil {
        return
    }
    defer objs.Close()

    chdir, err := link.Tracepoint("syscalls", "sys_enter_mount", objs.TraceMount, nil)
    if err != nil {
        return
    } else {
        defer chdir.Close()
    }

    readEvents(objs.Events, handler)
}

func readEvents(events *ebpf.Map, handler ExploitDetectionHandler) {
    reader, err := ringbuf.NewReader(events)
    if err != nil {
        return
    }
    for {
        record, err := reader.Read()
        if err != nil {
            if errors.Is(err, perf.ErrClosed) {
                return
            }
            return
        }

        bpfEvent := (*bpfEvent)(unsafe.Pointer(&record.RawSample[0]))
        devName := utils.ConvertCString(bpfEvent.DevName[:])
        dirName := utils.ConvertCString(bpfEvent.DirName[:])
        fsType := utils.ConvertCString(bpfEvent.FsType[:])
        timestampMicros := bpfEvent.Timestamp / 1000 // eBPF timestamp is in nanos
        handler(&Event{bpfEvent.Pid, devName, dirName, fsType, timestampMicros})
    }
}
```

Реализовав проверку монтируемой директории (dirName) на маску "/var/lib/docker/buildkit/executor/([a-zA-Z0-9]{24,})/rootfs/" сможем обнаруживать изменение точки монтирования кеша и его преднамеренного изменения.

Используя механизмы описания поведения основанные на статичных правилах, на основе которых работают большинство систем обнаружения вторжений (Intrusion Detection Systems, IDS), возможно реализовать конфигурируемую динамическую систему обнаружения атак на уровне целевой операционной системы и может быть задействован не только для применения с менеджерами контейнеров. Процедура создания правил показывает, что для эффективного мониторинга безопасности необходимо иметь точные знания о запущенных в контейнерах сервисах. Это включает точки коммуникации, требуемые пути файлов и названия программ [20]. Такой подход означает что набор правил для анализа должен быть постоянно обновляем.

VI. Заключение

Контейнеризированные среды обладают своими уникальными факторами подверженности угрозам. Существующие угрозы и риски (CVE) направленные на такие среды отличаются своей разнонаправленностью в направлении использовании уязвимостей. Наиболее критическими, с моей точки зрения, являются уязвимости позволяющие нарушить периметр контейнера и повысить привилегии при реализации атаки. Предложенные механизмы в статье позволяют реализовать системы определения и предотвращения (на уровне CRI) подобных уязвимостей. Отдельным

аспектом является создание базы сигнатур для анализа в обоих подходах, особенно в части динамического анализа основанного на eBPF. Анализ на eBPF для описания сигнатуры требует задания паттерна поведения системы, и чем разнобразнее предполагаемый задаваемый паттерн, тем больше аспектов системы необходимо наблюдать с помощью фильтров. Задача описания паттернов уже решается некоторыми существующими продуктами на рынке, вопрос масштаба наблюдения и перехвата с помощью фильтров требует конфигуруемоего подхода к настройке правил реагирования.

СПИСОК ЛИТЕРАТУРЫ

[1] Vartura.com, “Conteinirzation Security: A Deep Dive into Isolation Techniques and Vulnerability Management,” 2023. https://www.varutra.com/containerization-security-a-deep-dive-into-isolation-techniques-and-vulnerability-management/

[2] A. Wonga, E. Chekola, M. Ochoab, “On the Security of Containers: Threat Modeling, Attack Analysis, and Mitigation,” Computers & Security, 2023

[3] Z. Jian and L. Chen, "A defense method against docker escape attack.," in Proceedings of the 2017 International Conference on Cryptography, Security and Privacy. Association for Computing Machinery, New York, 2017.

[4] RedHat, "runc - malicious container escape - CVE-2019-5736," 2020. https://access.redhat.com/security/vulnerabilities/runcescape..

[5] "Vulnerability details: : CVE-2017-18641," 2020. https://www.cvedetails.com/cve/CVE-2017-18641/.

[6] J. Chelladhurai, P. Chelliah and S. Kumar, "Securing docker containers from denial of service (dos) attacks.," in IEEE International Conference on Services Computing, 2016.

[7] "Vulnerability details: : CVE-2020-15257," 2020. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-15257

[8] "Vulnerability details: : CVE-2021-41103," 2021. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-41103

[9] "Vulnerability details: : CVE-2022-31030, CVE-2022-1708, CVE-2021-3669," 2021-2022. [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name= CVE-2022-31030, CVE-2022-1708, CVE-2021-3669

[10] "Vulnerability details: : CVE-2022-0811," 2022. https://cve.mitre.org/cgi-bin/cvename.cgi?name= CVE-2022-0811

[11] J. Walker, “cr8escape: New Vulnerability in CRI-O Container Engine Discovered by CrowdStrike (CVE-2022-0811),” 2022. https://www.crowdstrike.com/blog/cr8escape-new-vulnerability-discovered-in-cri-o-container-engine-cve-2022-0811/

[12] “cri-o: Arbitrary code execution in cri-o via abusing “kernel.core_pattern” kernel parameter,” 2022. https://github.com/cri-o/cri-o/security/advisories/GHSA-6x2m-w449-qwx7

[13] T. Donohue, “The differences between Docker, containerd, CRI-O and runc. Containers run using a complex stack of libraries, runtimes, APIs and standards,” 2024. https://www.tutorialworks.com/difference-docker-containerd-runc-crio-oci/

[14] CRI Protocol Specification, 2024. https://github.com/kubernetes/cri-api/blob/master/pkg/apis/runtime/v1/api.proto

[15] “Introducing Container Runtime Interface (CRI) in Kubernetes,” 2016. https://kubernetes.io/blog/2016/12/container-runtime-interface-cri-in-kubernetes/

[16] R. McNamara, “Buildkit mount cache race: Build-time race condition container breakout (CVE-2024-23651),” 2024. https://snyk.io/blog/cve-2024-23651-docker-buildkit-mount-cache-race/

[17] LINUX SYSTEM CALL TABLE FOR X86 64. 2012. https://blog.rchapman.org/posts/Linux_System_Call_Table_for_x86_64/

[18] “What is e-BPF?”. e-BPF Documentation. https://ebpf.io/what-is-ebpf/

[19] J. Smith, “Leaky Vessels: Docker and runc container breakout vulnerabilities,” 2024. https://snyk.io/blog/leaky-vessels-docker-runc-container-breakout-vulnerabilities/

[20] H. Gantikow, “Rule-based Security Montoring of Containerized Workloads,,” in Proceedings of the 9th International Conference on Cloud Computing and Services Science, DOI: https://doi.org/10.5220/0007770005430550, 2019, pp. 543-550

On modern approaches to the design network processor unit.

Nikita Nikiforov
Lomonosov Moscow State University
Moscow, Russia
rav263@asvk.cs.msu.ru

Dmitry Volkanov
Lomonosov Moscow State University
Moscow, Russia
volkanov@asvk.cs.msu.ru

Alexey Volchaninov
Lomonosov Moscow State University
Moscow, Russia
mr.volchaninov@yandex.ru

Alexey Alexandrov
Lomonosov Moscow State University
Moscow, Russia
Alexandrov.AlexeyV@yandex.ru

***Index Terms*—Network processor, software-defined networks, packet classification, search trees.**

***Abstract*—Network processing unit (NPU) is a programmable processor. Its architecture is optimized to work in network devices and to provide stable packet processing mode. Such processing units are made to deal with the extraction, classification and processing of network packets. High-performance network switches meet a raw of different requirements. Main of these requirements are: support for a large number of network protocols (usage scenarios) and high bandwidth. Devices that are based on modern NPUs are able to transmit data at high speeds: up to 52 terrabits per second. To develop new NPUs it is necessary to determine which parts of such kind of processing units are the most important to pay attention to. To design a competitive device the following aspect should be clear: what are the main trends and possible improvements of the technical solutions that can be found in modern NPUs of common network vendors.**

In this article authors make a review of some latest devices of major manufacturers. The purpose of this review is to emphasize the most effective architectural solutions to use them for a new device and to improve its characteristics. The criteria of provided review are as following: NPU programmability, NPU type, chip characteristics, performance, classification table sizes, network packets pipeline features, memory organization, software structure, used monitoring methods, supported queuing methods, supported policies of QoS. Authors focus on several devices designed by the following companies: Huawei, Cisco, Broadcom, Barefoot and Nvidia. As a result the most perspective solutions found in observed devices documentation are highlghted. Also a set of NPUs parts that are of the greatest interest for the following improvements has been found.

I. General view on NPUs architecture

In general, a network packet processing device [1], [2], consists of five main blocks:

1) interface modules for receiving input data and transmitting output data with optional buffering;
2) programmable blocks for packets processing that can be implemented using a pipeline;
3) internal memory;
4) external memory access control unit;
5) CPU or CPU interface.

Common NPU architecture is shown in Figure 1.

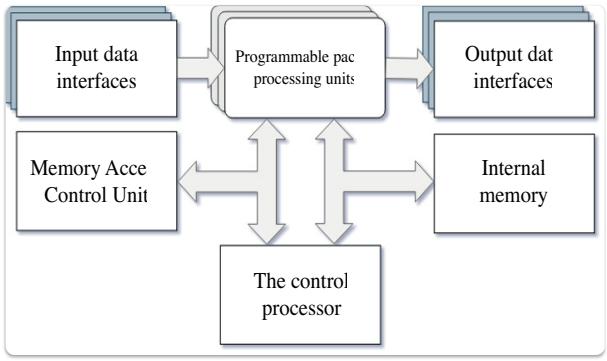

Fig. 1: Common NPU architecture

1) **Interface modules for receiving input data and transmitting output data with optional buffering.** Interface for receiving input data are PHY receiving data block interfaces that implements an access to the physical data transmition environment. The data received from these interfaces is interpreted as incoming packets and transmitted to the packet processing units. Interfaces oh the output data are the PHY physical environment transmitters interfaces.
2) **Programmable packet processing blocks.** Packet processing blocks implements core NPU features: packet fields emphasizing; classification; modification. These functions can be performed sequentially by one block or in parallel on many different blocks. The packet processing units can be fully or partially programmable, can be equipped with some specialized processors or with processors based on general-purpose processor cores. The programmability capabilities are defined by the type of NPU.
3) **Internal memory, memory on the chip.** The internal memory in located on the chip. Most of the implementations use this memory to store the clas-

TABLE I: Comparison of modern network processors according to the review criteria

Name	Release year	Performance	Process programmability	Type	Available switches interfaces configurations
Marvell Octeon 10	2024	800 Gb/s	Yes	SoC ARM processor	16x50 GbE
Broadcom Trident 5	2022	Up to 16.0 Tb/s	Yes	ASIC	24x400GbE 8x800GbE
Marvell Teralynx 10	2023	Up to 51.2 Tb/s	Yes	ASIC	128x400GbE, 256x200GbE, 512x25/50/100GbE
Broadcom Tomahawk 5	2022	Up to 51.2 Tb/s	Yes	ASIC	64x800GbE, 128x400GbE, 256x200GbE
Nvidia Spectrum 4	2022	Up to 51.2 Tb/s	Yes	ASIC	64x800GbE, 128x400GbE, 256x200/100 /50/25/10GbE
Huawei NetEngine 8000 F8	2022	Up to 6.4 Tb/s	Yes	ASIC	4x400GbE, 24x100GbE, 48x50GbE, 240x10GbE, 448xGbE
Cisco Nexus 9800 400G	2019	Up to 14.4 Tb/s	Yes	ASIC	36x400GbE, 48x100GbE, 10x25x50GbE

sification tables, some of them even use it to store packet bodies. SRAM and TCAM memory types are used for internal memory.

4) **External memory control unit.** Network switches can be equipped with blocks of external memory (DDRDRAM or RLDRAM arrays) for temporary placement of the processed packets bodies, . As usual, classification tables and other data that needs fast access are located in internal memory or in TCAM memory blocks. Metadata of the processed network packet such as: header, output port, packet body address (in memory) are usually located in register files. Access time for the external memory is access time to external memory is several times bigger than to on-chip memory. That is the reason why the packet bodies, binary trees and hash-tables for packet classification can be stored using this memory.
5) **CPU or CPU interface.** In network switches the processing unit can be implemented as a processor based on general-purpose processor cores that is presented by separate micro scheme element or by a certain part of the network processor VLSI. The processing unit is frequently used to implement the administrator interfaces and the management software. It is also responsible for some auxiliary problems: collecting statistics, providing an access to peripheral devices and etc. These problems are not directly related to the packet processing. Peripheral devices interfaces are made to connect the processing unit that is responsible for system management and external NPU blocks.

II. Stocking points of the modern NPUs development

Modern NPUs architectures have many growth points:

- network packet header processing;
- organization of the memory hierarchy;
- organization of classification tables;
- organization of the NPU pipeline;
- organization of re-processing network data units;
- organization of the processing program;
- organization of queues;
- quality assurance methods;
- methods of monitoring the state of the NPU.

III. Review of modern NPUs

A. Criteria of the review

The following criteria are used to make the review of network switches architecture:

- year of release;
- brief description;
- price (price of the network device that uses NPU).
- performance – maximum possible bandwidth of the switch;
- processor unit programmability – ability to develop new programs for network traffic processing, programming language and SDK (Software Development Kit) that are used for it;
- type – ASIC, NPU or integrated circuit based on general purpose processors;
- available configurations of the switch interfaces;
- key architecture features description (if available):
  - device / network packets pipeline structure;
  - network packets processing units characteristics;
  - traffic control unit structure;
  - classification tables memory type;
  - classification tables features (used data structures, size) ;
  - the location of memory blocks for package bodies and classification tables relative to the crystal;
- chip characteristics (if available):
  - energy consumption;
  - process technology;
  - micro-scheme square;

TABLE II: Comparison of modern network processors according to the review criteria

Name	Device / packet processing pipeline structure	packet processing processors characteristics	classification tables memory type	organization of the classification tables	energy consumption	technical process
Marvell Octeon 10	Programmable pipeline	vector network packets handlers	DDR5	On the chip	70 Вт	5 nm
Broadcom Trident 5	Programmable pipeline and NetGNT	No data	Shared access memory	On the chip	No data	5 nm
Marvell Teralynx 10	Programmable pipeline	No data	No data	On the chip	1.5 kWt per device	5 nm
Broadcom Tomahawk 5	Programmable pipeline	No data	No data	On the chip	4 kWt per device	5 nm
Nvidia Spectrum 4	Programmable pipeline	No data	No data	On the chip	500 Wt per chip, 2 kWt per device	5 nm
Huawei NetEngine 8000 F8	Programmable pipeline	No data	No data	On the linear cards	2 Тб/с: 1.6 kWt; 6.4 Tb/s: 2.3 kWt per device	No data
Cisco Nexus 9800 400G	Programmable pipeline	No data	TCAM	On the chip	No data	7 nm

- control processor on a chip (if provided);
- type of CPU interface;
- used methods of switch parameters monitoring;
- supported queuing methods, QoS policies;
- switch software architecture.

Review focuses on the following devices:

1) NVIDIA (Mellanox) SPECTRUM-4
2) Marvell Teralynx 10
3) Marvell Octeon 10
4) Broadcom Tomahawk 5
5) Broadcom Trident 5
6) Cisco Nexus 9800
7) Huawei NetEngine 8000 F8

B. NVIDIA (Mellanox) SPECTRUM-4

Nvidia Spectrum 4 Proccessor [3]in an programmable ASIC (Application-Specific Integrated Circuit). This allows users to set its work up according to some certain requirements of an application. Bandwidth of this processor 51.2 terrabits per second. Such a high perfomance gives this processing unit an opportunity to deal with large data volumes rather effectively. Spectrum 4 provides different configurations of the switch interfaces, for example: 64x800GbE, 128x400GbE, 256x200/100/50/25/10GbE. Different methods of switch parameters monitoring are available using this processor, for example: FabricView telemetry, open and scalable API. Processing unit is equipped with fully distributed packets buffer. This feature increases the effectiveness of network packets processing.

The manufacturer also provides an adaptive routing, load balancing and network congestion. Traffic management and congestion prevention become more effective due to this feature.

The technical process of the processing unit is 5 nanometers. It leads to high performance and low energy consumption. The processor consumes 500 watt per chip and 2 kilowatts per device. This is a quite low energy consumption for a processor with such a high performance.

Finally, processing unit can be used with different CPUs (Control Processing Units), such as: Intel x86, AMD x86, ARM. Interface to connect with the CPU is PCIe4 x4.

C. Marvell Teralynx 10

This device was developed in 2023 [4]. Its bandwith equals to 51.2 terrabits per second. Processing unit is programmable and compatible with SDK Mellanox Teralynx 7.

The device has a programmable network packet processing pipeline that uses special processor cores. Traffic management is organized using the control processing unit. Classification tables are stored with special memory that is located on the chip.

Technical process of this device equals to 5 nanometers. Its energy consumption equals to 1.5 kilowatts. The manufacturer keep the square of the microscheme in secret. Chip is equipped with a control processing unit of ARM arcitecture with PCIe5 x4 interface.

Special proprietary Teralynx Flashlight utilities are used to monitor the switch parameters.

QoS policies hardware support includes the following features: big packet buffers and scalable shared law latency bus.

D. Marvell Octeon 10

Marvell Octeon 10 processor was introduced in 2024 [5]. It is a SoC ARM with 3 ARM Neoverse N2 2,7 gigahertz cores with several hardware accelerators on the chip.

TABLE III: Comparison of modern network processors according to the review criteria

name	CPU on the chip	CPU interface type	used methods of monitoring the switch parameters	quality assurance methods	quality management methods	architecture of the device software
Marvell Octeon 10	Yes, ARM with 36 ARM Neoverse N2 2,7 GHz cores	PCIe 5	Network analytics: sFlow, IPFIX	hardware queue management	Hardware queuing	Proprietary Marvell
Broadcom Trident 5	Yes, 6-core ARM processor	No data	No data	Broadview Gen 4	No data	No data
Marvell Teralynx 10	ARM microprocessor	PCIe5 x4	propritary Teralynx Flashlight utilities, P4 bus telemetry	No data	Big buffers, scalable shared bus with low latency	Mellanox SDK
Broadcom Tomahawk 5	ARM microprocessor	No data	Competible with OpenBMC, Open Network Install Environment	No data	No data	Cometible with OpenBMC, Open Network Install Environment
Nvidia Spectrum 4	No	PCIe4 x4	FabricView telemetry, open and shared API WJH	Fulle distributed packet buffer	Adaptive routing, load balancing, congestion control	No data
Huawei NetEngine 8000 F8	No	No data	Monitoring of the key performance indicators (KPI)	Software and hardware queuing	No data	No data
Cisco Nexus 9800 400G	No	PCIe 4	Monitoring of the SPAN and RSPAN ports	Hardware queuing	Hardware	No data

Maximum bandwidth provided by this processing unit equals to 800 Gb/s. It supports a configuration with 16 ports with 50 GbE for each. Manufacturer provides a programmable pipeline with vector packets handlers and specialized cores. DDR5 memory is located on the chip. Energy consumption of the processor equals to 70 watts, its technical process equals to 5 nanometers.

Network analytics methods such as sFlow and IPFIX are used to monitor the switch parameters. Device supports such queuing methods as hardware queues management and hardware queuing.

Software architecture is Marvell proprietary.

E. Broadcom Tomahawk 5

Broadcom Tomahawk 5 [6] processing unit was revealed in 2022. Its price is not in public access. Bandwitch of this device equals to 51.2 terrabits per second. Processor is programmable, NPL language is used to make programs for it. This processor is an ASIC (Application-specific integrated circuit). The device supports different switch interfaces configurations: 64x800GbE, 128x400GbE, 256x200GbE. Network packets processing is performed on the chip using a programmable pipeline. The following classification tables are supported by this processing unit: 128 thousands of MAC table records, 4 thousands of VLAN table records, 850 thousands of ALPM(V4) records, 360 thousands of ALPM(V6_64) records and 240 thousands of ALPM(V6_128) records.

The device using this processor consumes 4 kilowatts of power. CPU is located on the chip, it is a 6-core ARM that wark with OpenBMC and Open Network Install Environment.

F. Broadcom Trident 5

Broadcom Trident 5 [7] processor is programmable and was presented in 2022. Its technical process equals to 5 nanometers. It leads to a high bandwidth of 51.2 terrabits per second. This processing unit is an ASIC. It supports different switch interfaces configurations: 64x800GbE, 128x400GbE and 256x200GbE.

Programmable pipline is responsible for packet processing. The following classification tables are used: 128 thousands of MAC records, 4 thousands of VLAN records, 850 thousands of ALPM(V4) records, 360 thousands of ALPM(V6_64) records and 240 thousands of ALPM(V6_128) records.

Energy cosumption of a device using this processor is up to 4 kilowatts. CPU is located on the chip, it‘s a 6-core ARM that works with OpenBMC and Open Network Install Environment. This feature provides an effective software architecture and quality management.

G. Cisco Nexus 9800

The device was introduced in 2022, its announced bandwidth equals to 6.4 Tb/s. This NPU is an ASIC with programmable pipeline. The device supports different configuration of such interfaces as Ethernet 100GbE,

50GbE, 40GbE, 25GbE, 10GbE and 1GbE. Classification tables are stored using linear cards with TCAM memory.

The device have two different performance modes: 2 Tb/s and 6.4 Tb/s. Energy consumption when using these modes equals to 1.6 KWt and 2.3 KWt correspondingly.

This device supports intelectual monitoring of key performance indicators(KPI). Both software and hardware instruments are used for queuing and traffic management.

H. Huawei NetEngine 8000 F8

This device was presented in 2019. Its bandwidth equals to 14.4 Tb/s. This NPU is an ASIC with programmable pipeline. Its technical process equals to 7 nm.

The device supports different of such interfaces configurations as: 36x400GbE, 48x100GbE and 10x25x50GbE. Classification tables are stored via TCAM that is located on the chip.

This device is equiped with Intel Xeon general purpose processor with PCI Express interfaces of the fourth generation. The SPAN and RSPAN ports monitoring is supported. Hardware instuments are used for queuing and traffic management.

I. Comparison of the reviewed architectures

All considered NPUs belong to the backbone class, with the exception of Marvel Octeon 10. It is used for endpoint maintenance of 5G network base stations. Also, almost all processors are made according to modern 5nm and 7nm process technologies, which allows the considered NPUs to have significantly reduced power consumption. A comparison of the considered SPAS according to the review criteria is presented in the tables: I, II, III.

Modern solutions in the architectures of the considered NPUs allow to achieve speeds up to 51.2 TBit/s. Some NPUs use GDDR5 memory to store classification tables, which makes the device cheaper, unlike using TCAM memory.

Almost all of the considered NPUs are of the ASIC type, which simplifies the production and design of the chip, but complicates the implementation of new protocols.

IV. Solutions to the problems of modern NPUs

A. Header processing

The header processing stage includes separating the packet body and extraction separate headers for each supported protocol. This process is crucial for understanding how to handle the packet correctly. The use of packet descriptors, which are special data structures that store the header fields required for processing, can improve the efficiency of further packet processing. These descriptors allow for quick access to the necessary header information, thereby reducing the time needed to send necessary information between conveyor stages.

B. Memory organization

In network processing devices, memory is organized hierarchically to ensure high performance. High-speed SRAM is used for temporary storage of packets and metadata, DRAM is used for larger data storage, and cache memory is used to speed up access to frequently used data. This organization allows for the efficient processing of large volumes of network traffic in real time. Classification tables are also stored in the memory units to facilitate quick lookup and efficient packet processing. This organization allows for the efficient handling of large volumes of network traffic in real time.

C. Classification tables organization

Modern classification tables use various variants of binary trees as methods of searching and storing data. The problem with this solution is a large number of pointers to the memory cells that store the rules. Using data hashing methods aside of Linked Lists, it is possible to achieve better overhead memory performance, and even improve search time. As downside, the time for inserting a new rule into the table worsens slightly.

D. Pipeline organization

Pipelining plays a vital role in achieving effective data processing in network devices. It enables faster communication speed by allowing for parallel processing while maintaining optimum utilization of network bandwidth resources. Data transfers over networks can be problematic when dealing with heavy volumes, packet loss during transmission or routing process can add additional delays and reduce network efficiency, which can lead to poor usability in communication. There is a raw of investigations that focus on different aspects of pipelines development, for example: [8], [9],The pipelined architecture is able to send numerous packets simultaneously serially without stopping until all packets have been sent without waiting for previous acknowledgments from receiver. Pipelining enables the overlapping processing of packets at different stages/stations of network communication lines, allowing more efficient utilization of available bandwidth and resources over the network. Pipelining techniques for packet switching reduces network latency and response times to a minimum while reducing congestion points since simultaneous processing reduces intermediate waiting time/queues formation leading to high bandwidth usage optimization rates that improve overall network performance.

E. Queueing methods

Queueing mechanisms are used in any network device where packet switching is used — a router, a network switch, or a destination node. The need for a queue arises during network congestions. If the cause of the congestion is the processor block of the network device, then the raw packets are temporarily placed in

the input queue. In the case when the cause of the congestion is the limited speed of the output interface (and it cannot exceed the speed of the supported protocol), the packets are temporarily stored in the output queue. Using intellectual queuing methods leads to increasing the effectiveness by decreasing buffer time of input or output ports. Also queuing methods often make queues based on some packet priorities. This leads to some of remarkable abilities for implementation of QoS policies. There is a raw of queuing policies [10], that are used in modern network devices, for example:

1) FIFO - First In First Out
2) Priority Queuing
3) Custom Queuing
4) WFQ - Waited Fair Queuing

Adding features related with configuration of queuing methods into network devices makes it universal and increases its performance.

F. QoS methods

Quality of Service (QoS) is an important aspect in modern network technologies. QoS techniques can be applied in various areas to ensure the reliability and availability of mission-critical applications. The following QoS methods allow you to manage the use of available network resources, optimize data transfer and ensure priority for mission-critical applications:

1) bandwidth limitation
2) priority management
3) delay control
4) jitter control
5) limiting the number of data packets

These methods can be applied in different areas, for example: mobile networks [11], high performance networking [12] and etc. However, there is some limitations: QoS methods may require additional resources and complex network configuration, which may increase the cost and complexity of its maintenance. In addition, the effectiveness of QoS methods may depend on the type of network applications and network characteristics that are used in a certain network device.

G. Monitoring network switches parameters methods

This point is very important, because monitoring network switches parameters gives automatic or manual configuration systems an opportunity to change the current device configuration according to real-time parameters values. Monitoring can be used for a raw of different problems, for example: filtering of network data [13], preventing DDoS attacks [14],and etc. All manufacturers of network devices also provide monitoring tools. Making data that is collected by this tools more accurate and full and making user interfaces of such tools more friendly for users increases QoE metrics of network devices.

V. Conclusion

In this article authors make a review of some latest devices of major manufacturers. The article considered the architectural problems of modern NPUs faced by major manufacturers: Huawei, Cisco, Broadcom, Barefoot and Nvidia.

The results of the review were possible architectural solutions that can be applied in the new NPU. Also a set of NPUs parts that are of the greatest interest for the following improvements has been found.

References

[1] Theofanis Orphanoudakis and Stylianos Perissakis. “Embedded multi-core processing for networking”. In: *Multi-Core Embedded Systems*. CRC Press, 2018, pp. 429–494.

[2] Mellanox. *Mellanox NP-5 Product Brief*. https://www.mellanox.com/related-docs/prod$_{n}pu/PB_{N}P-5.pdf$. [Accessed 06-11-2021]. 2021.

[3] *Spectrum-4 Datasheet — resources.nvidia.com*. https://resources.nvidia.com/en-us-accelerated-networking-resource-library/ethernet-switches-pr?xs=425229#page=1. [Accessed 02-06-2024].

[4] Marvell. *marvell teralynx 10 Product Brief*. https://www.marvell.com/content/dam/marvell/en/public-collateral/switching/marvell-teralynx-10-data-center-ethernet-switch-product-brief.pdf. [Accessed 01-06-2024]. 2023.

[5] Marvell. *marvell octeon 10 Product Brief*. https://www.marvell.com/content/dam/marvell/en/public-collateral/embedded-processors/marvell-octeon-10-dpu-platform-product-brief.pdf. [Accessed 01-06-2024]. 2023.

[6] *Tomahawk 5 — broadcom.com*. https://www.broadcom.com/products/ethernet-connectivity/switching/strataxgs/bcm78900-series. [Accessed 20-06-2024].

[7] *ToR Switches — broadcom.com*. https://www.broadcom.com/products/ethernet-connectivity/switching/strataxgs/bcm78800. [Accessed 20-06-2024].

[8] Jamil Al Azzeh, Mohammed Abdo, and Igor Zotov. “A PARALLEL PIPELINED PACKET SWITCH ARCHITECTURE FOR MESH-CONNECTED MULTIPROCESSORS WITH INDEPENDENTLY ROUTED FLITS”. In: *Jordanian Journal of Computers and Information Technology* 05 (Aug. 2019), p. 1. DOI: 10.5455/jjcit.71-1556375171.

[9] Kimon Karras, Thomas Wild, and Andreas Herkersdorf. “A folded pipeline network processor architecture for 100 Gbit/s networks”. In: Oct. 2010, p. 2. DOI: 10.1145/1872007.1872010.

[10] https://web.archive.org/web/20160404153707/http://www.openbsd.org/faq/pf/queueing.html. [Accessed 20-06-2024].

[11] A.G. Markoc and G. Sisul. “Quality of service in mobile networks”. In: Jan. 2013, pp. 263–267. ISBN: 978-953-7044-14-5.
[12] Abhijit Bora and Tulshi Bezboruah. “Some Aspects of QoS for High Performance of Service-Oriented Computing in Load Balancing Cluster-Based Web Server”. In: Jan. 2017, pp. 557–592. ISBN: 9781522517856. DOI: 10.4018/978-1-5225-1785-6.ch021.
[13] Sean Pennefather, Karen Bradshaw, and Barry Irwin. “Real-Time Geotagging and Filtering of Network Data using a Heterogeneous NPU-CPU Architecture”. In: Sept. 2018.
[14] Oğuz Yarımtepe, Gökhan Dalkılıç, and Mehmet Ozcanhan. “DDoS Prevention Techniques”. In: May 2015.
[15] Niraj Shah and Kurt Keutzer. “Network processors: Origin of species”. In: *International Symposium on Computer and Information Sciences*. CRC Press. 2022, pp. 41–45.

Methods for marking network traffic routes to solve the task of predicting the quality of heterogeneous channels

Alexander Ryazanov
Lomonosov Moscow State University
Moscow, Russia
alryaz@xavux.com

Dmitry Volkanov
Lomonosov Moscow State University
Moscow, Russia
volkanov@asvk.cs.msu.ru

I. Introduction

Quality of Service (QoS) is a technology that provides different classes of traffic with various priorities in service [1].

This work considers the following 4 QoS indicators:

- bandwidth;
- latency;
- jitter;
- loss ratio.

In modern conditions, it is critically important to create sovereign solutions in the field of algorithmic support for QoS management in heterogeneous channels in data transmission networks (DTNs). Known approaches and products (management complexes of cloud and/or edge computing, software-configurable overlay networks, etc.) can hardly be called entirely Russian. Firstly, they contain several modules that are either proprietary or freely distributed foreign software. Secondly, the mentioned intelligent network management tools are trained based on foreign repositories. Additionally, passive borrowing of applied task settings for centralized/decentralized resource management of DTNs and the widespread use of existing algorithmic support hinder the proactive development of the theoretical base and mathematical apparatus for effectively solving the growing number of applied tasks for managing and optimizing the use of DTN resources. All this does not allow creating trusted software complexes and, in general, does not lead to ensuring technological sovereignty of the Russian Federation in the field of information technology. The reasons for the current problematic situation in the field of mathematical and algorithmic support are:

- the lack of adequate mathematical models of modern DTNs and their components, due to the scale, complexity, and dynamic development of the systems themselves;
- the uncertainty and non-stationarity of the DTN structure, data transmission flows;
- incomplete available information about the internal algorithms of the functioning of DTN hardware and software, related to trade secrets;
- insufficient use of a large amount of observations that indirectly contain useful information about the current load level and state of various parts of DTNs;
- lack of trusted training sample resources for software configuration; "poverty" of existing resources in terms of the diversity of scenarios of heterogeneous data flow fluctuations;
- lack of theoretical results on estimation in some classes of stochastic systems.

This article considers a range of subtasks related to the formation of training sets that can be used to solve the tasks of analyzing and predicting the QoS of heterogeneous channels in DTNs.

Section 1 of this article presents the formulation of the problem of predicting the quality of heterogeneous channels in DTNs. Section 2 describes the QoS parameters for the heterogeneous channel, and section 3 provides the choice of methods for collecting and storing network traffic. Section 4 contains a description of the architecture of tools for collecting, storing, and analyzing network traffic.

II. Formulation of the Problem of Predicting the Quality of Heterogeneous Channels in Data Transmission Networks

Informally, the problem of predicting the quality of heterogeneous channels in DTNs can be formulated as follows:

Given:

- a set of characteristics of the studied heterogeneous channel (maximum bandwidth, maximum packet loss percentage, etc.);
  1) maximum bandwidth;
  2) maximum packet loss percentage;
  3) nominal range of delay fluctuations.
- a network traffic trace collected on this channel over the time interval $[0; T]$, containing information about the headers of protocol data units from the data link layer (L2) to the transport layer (L4). It may also contain information about application layer (L7) headers to simplify traffic identification and extract information about QoS characteristics.
- a set of QoS indicators:

– bandwidth;
– latency;
– jitter;
– loss ratio.

Required:

- determine the QoS indicator values over the interval $[0; T]$;
- predict the QoS indicators over the time interval $(T; T^*]$.

This problem is broken down into the following subtasks:

1) development of methods for calculating numerical values of QoS indicators for the heterogeneous channel;
2) development of methods for collecting network traffic;
3) development of methods for storing network traffic;
4) development of methods for predicting QoS indicators of the heterogeneous channel.

This article considers subtasks 1, 2, and 3.

III. QoS Parameters for the Heterogeneous Channel

Let's take a closer look at the four QoS indicators and consider ways to calculate them [2].

1) Bandwidth: Bandwidth is a quantitative characteristic that reflects the data transmission capabilities of a particular connection medium. In DTNs, bandwidth refers to the amount of data that can be transmitted from one point to another in a certain period.

2) Type-P-One-way-Delay: Measuring one-way delay instead of round-trip delay is due to the following factors:

- In heterogeneous channels, the path from the source to the destination may differ from the path back to the source ("asymmetric paths"), so different sequences of intermediate network nodes are used for the forward and backward paths. Independent measurement of each path reflects the performance difference between the two paths.
- Even if the two paths are symmetric, they may have differing delays due to the asymmetric organization of the queuing mechanism.
- Application performance may depend mainly on performance in one direction. For example, file transfer using TCP may depend more on the performance of the data flow direction rather than the acknowledgment direction.
- In QoS-enabled networks, provisioning in one direction may radically differ from provisioning in the reverse direction, and hence QoS guarantees differ. Independent measurement of paths allows verifying both guarantees.

3) Jitter: Jitter is the difference in packet delays within the same stream.

If the period before a packet that has reached the device is sent by the device differs from one packet to another in the stream, fluctuations occur, and QoS deteriorates. For some services (voice, video), the presence of high jitter is unacceptable as it causes real-time interruptions.

4) Loss rate: Packet loss occurs when one or more data packets transmitted over a DTN do not reach their destination. Packet loss can be caused by either transmission errors or network congestion. Packet loss is measured as a percentage of lost packets relative to sent packets:

$PLR = \frac{N^{tx} - N^{rx}}{N^{tx}}$

where N_{tx} and N_{rx} are the total number of transmitted and received protocol data units (PDUs) respectively. This estimate can be easily performed by extracting all packet sizes in real-time, which are transmitted and received respectively.

IV. Methods for Collecting and Storing Network Traffic

In the work [3], three main methods of traffic collection with different memory requirements were considered:

- collection of all packets;
- collection of network flow;
- collection of extended flow.

In the first method, the collection, processing, and storage of a copy of each traffic packet for subsequent analysis are performed. These data contain complete information about the packet headers and the information transmitted in the packets. When using the second method, the size of the collected information is reduced due to the fact that the network flow is a set of network packets that meet several conditions (such as packets sent approximately at the same time, having the same source address and port number, having the same destination address and port number, using the same protocol). The extended flow collection includes the collection of all packets and network flow. At the same time, information from the flow is added to the information directly from the packet headers or the information transmitted in the packets. Together, the extended flow may also contain additional information from some external source, such as the geographical location of the source and destination IP addresses. For the task of predicting QoS for heterogeneous channels in DTNs, it is sufficient to collect information only about network packet headers and apply a hash function to user data. When collecting data, the device on which the traffic collection is carried out also anonymizes it, including the anonymization of confidential data and truncation of data not required for further use.

When collecting traffic, each packet arriving at the device must go through processing, including the following stages:

1) filtering;
2) anonymization;
3) compression;
4) storage.

Various data storage methods can be used for storing network traffic, such as files (e.g., log files); databases and their combinations [3]. Each of these methods has its own aspects. Most data collection and analysis applications support the pcap format [4], so it was chosen as the storage format.

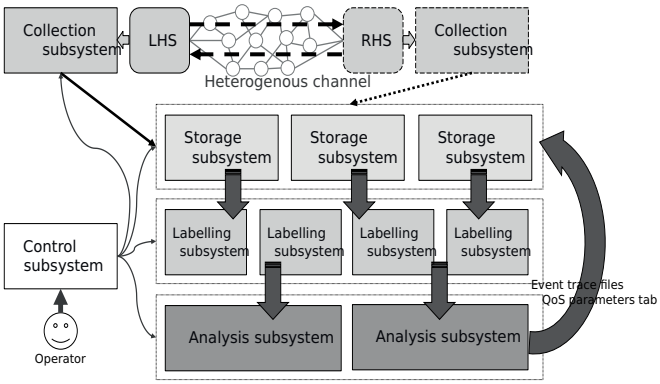

Fig. 1: Diagram of the collection and analysis system architecture

V. Architecture of Tools for Collecting, Storing, and Analyzing Network Traffic

The ability to capture packets and save them for subsequent analysis allows the tasks of information collection and its analysis to be separated.

Thus, the process of data collection can be represented in the form of the following components [5]:

- a tap device that collects information directly from the data stream;
- a server that saves the collected data;
- an analysis server that analyzes the collected data.

The diagram depicts the process of collecting and analyzing network traffic. A tap device is connected to the network and captures all incoming and outgoing traffic. The collected data is sent to the server, where it is stored in pcap files. The analysis server processes the collected data to predict QoS indicators.

VI. Conclusion

This paper has presented an approach to collect and store network traffic for solving the problem of predicting the quality of heterogeneous channels in DTNs. The proposed method collects only the necessary data, ensuring its anonymization and compression. The architecture for collecting and storing traffic separates the collection process from the analysis, enabling efficient and secure data handling. Further work will focus on developing methods for predicting QoS indicators based on the collected traffic data.

References

[1] I.-T. G.-S. Recommendation, "Communications quality of service: A framework and definitions," *International Telecommunication Union*, 2001.

[2] H. T. Co., "Quality of service (qos) technology," *Huawei Technologies Co., Ltd.*, 2017.

[3] A. Shyhaliev, "Overview of traffic collection methods in modern networks," *Networking Technologies*, vol. 24(2), pp. 14–19, 2016.

[4] V. Jacobson, C. Leres, and S. McCanne, "Pcap: Packet capture library," 1989.

[5] Y. Lu, J. Guo, and T. Luo, "A review of network traffic analysis," *IEEE Access*, vol. 9, pp. 3545–3560, 2021.

Models for intelligent management telecommunications into transport logistics of data economy

A.P. Shabanov
Federal research center "Computer Science and Control"
Russian Academy of Sciences
Moscow, Russia
apshabanov@mail.ru

A.A. Zatsarinnyy
Federal research center "Computer Science and Control"
Russian Academy of Sciences
Moscow, Russia
alex250451@mail.ru

***Abstract*—At the stage of development of transport logistics in the conditions of the data economy, the processes of information transmission to railway, automobile and aviation vehicles that carry out the movement of goods or products at distances in a transport corridor equipped with radio access points are being investigated. The task of ensuring continuity of communication with vehicles over the distances of the transport corridor has been set and solved, in which there are differences in the parameters of access to the data transmission network in the geographical zones of this corridor. New information and logical models for intelligent management telecommunications have been developed – data transmission from the control point to the vehicle. The novelty of the models lies in ensuring continuous transmission of information when passing a vehicle in neighboring distances, which differ in the parameters of access to the data transmission network. To implement novel models, it is recommended to apply a method consisting in the implementation of these models in well–known technical solutions - in digital platforms for supporting organizational systems processes. The positive effect of the use of novel models is to ensure continuous communication with vehicles and, as a result, to increase stability when remotely controlling the movement of these vehicles along the distances of the transport corridor.**

I. INTRODUCTION

The development of the logistics industry at the current historical stage of the data economy [1, 2] is characterized by an increasing scale of automation of transport logistics processes, which should ensure the achievement of the targets of projects for the delivery of goods and products to on time and with the declared quality.

In transport logistics, these processes are:

• logistics processes that solve the problems of organizational and financial support of ways of delivering goods and products to along transport corridors [3-5];

• transport (track) processes that solve the tasks of logistical and digital support on railways, highways and aviation lines, for example [6] (the work examines organizational, technical and engineering measures for the development of digital infrastructure);

• information and telecommunication processes that provide solutions to the problems of controlling the movement of vehicles over the distances of the transport corridor, for example [7] (problems of information interaction in railway and maritime transport logistics are studied), [8] (methods and models, technologies and technical solutions designed for adaptive management of information systems are studied).

A. The subject area of the research

In Fig. 1 presents a structural diagram of processes and tasks in a logistics project, which are implemented on the basis of the forces and resources of enterprises-owners of goods and products, -operators and -manufacturers of products, enterprises-purchasers of goods, -product maintenance centers, factories, logistics companies and centers, transport enterprises and companies providing information and telecommunications services in geographical areas (hereinafter referred to as areas) adjacent to the transport corridor.

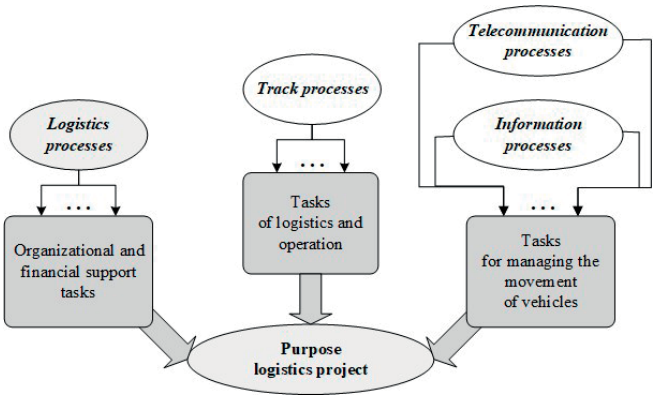

Fig. 1. A block diagram of processes and tasks in a logistics project (projects).

The block diagram of processes and tasks (Fig. 1) clearly shows the logical connection between the tasks that need to be solved and the processes that need to be completed in order to achieve the target of a logistics project:

< The target of a logistics project can be achieved by solving the tasks of organizational, financial, logistical support, operation and management of the movement of vehicles >. (1)

⇩

< The tasks of the logistics project (1) can be solved by performing logistics, track, information and telecommunication processes >. (2)

The subject area of this research is the telecommunication processes of information transmission between the vehicle movement control point and its on-board equipment in the conditions of the information digitalization stage, which is infrastructurally characterized by highways equipped with radio access points to data transmission networks in areas of the transport corridor with the

possibility of accessing the Internet and further to vehicle control points (systems).

B. Relevance of the research

This study is being conducted for the first time and is due to a specific factor inherent in the stage of digitalization of information in the data economy, consisting in various types of activities in the transport corridor zones carried out by small, medium and large enterprises that locate their offices and branches, retail outlets, repair centers, gas stations, stations, including radio access points to data transmission networks in these areas.

Each enterprise requires radio communication services with access to the data transmission networks of the companies providing these services in the areas under consideration, followed by Internet access to conduct digital operations by type of activity.

The diversity factor is inherent, in particular, in companies providing radio communications, data transmission, information and meteorological services, as well as other services, including vehicle location, navigation and video surveillance of vehicles, including aircraft.

Along with the intensive economic development of areas in the transport corridor, which, in fact, is evidenced by this factor, it is also a source of a new objective situation that can have a negative impact on the processes of data transmission to vehicles.

The reason is that the parameters of access to data transmission networks for service providers, as a rule, differ. As a result, communication is interrupted for the time of changing access parameters for vehicles.

In order to clarify and formalize the entities related to this objective situation, based on the analysis of scientific and scientific and technical publications, for example [8, 9], and patents (RU2744296C1, RU2784715C1), a hypothetical scheme for supporting the processes of organizational systems (enterprises, companies, services, ...) participating in logistics projects in the transport corridor has been developed (Fig. 2).

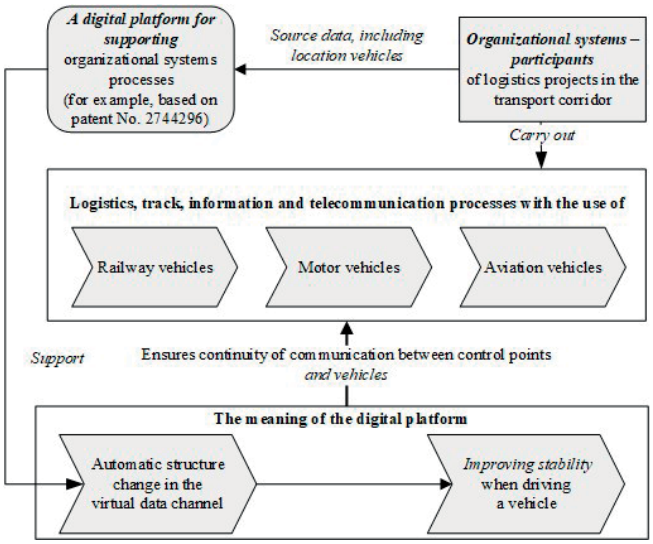

Fig. 2. A hypothetical scheme of support processes for participants in logistics projects.

Based on a cognitive-subjective examination of the content of the circuit components (Fig. 2), conducted taking into account the hypothetical possibility of using the resources indicated in the diagram of the digital platform for supporting organizational systems processes (patent RU2744296C1), the study formulated the following chain of logical judgments (3) – (6):

< The digitalization of information in the data economy is characterized by a factor of diversity of activities in the areas of the transport corridor, including the provision of radio communication and data transmission services >. (3)

⇩

< The technical reality at the distances of the transport corridor, which is due to factor (3), is the heterogeneity of access parameters vehicle equipment to data transmission networks >. (4)

⇩

< Movement of vehicles at transport distances The corridor is carried out in the conditions of technical reality (4) >. (5)

⇩

< During the movement of vehicles in conditions (5), objective situations may arise – at the next distance of the transport corridor, the parameters of access to the data transmission network differ from the parameters previous distance >. (6)

⇩

< If an objective situation arises (6), there is a break in communication between the control point and the vehicle at the time of making changes to the access parameters >. (7)

From the above chain of logical judgments (3) – (7) it follows that in transport logistics, at the stage of digitalization of information in the data economy, new situations objectively arise that can initiate interruptions in the processes of managing the movement of vehicles in transport corridors.

Thus, conducting the research aimed at preventing (preventing) communication interruptions caused by the manifestations of the considered objective situation is an urgent event.

C. Research problems

In order to prevent interruptions in communication between control points and vehicles driven by them, due to the circumstances discussed above, the following scientific and scientific-technical research tasks have been set and solved.

The scientific problem is to ensure the continuity of communication between the control point and the vehicle during its passage along the distances of the transport corridor equipped with radio access points, regardless of the difference in the parameters of access to data transmission networks set in them.

The scientific and technical problems of the research is to develop novel information and logical models for intelligent

management telecommunications - data transmission from the control point to the vehicle.

The information and logical models for intelligent management telecommunications are tools for solving a scientific problem.

The solution of the problems set is designed to ensure the stability of vehicle control by maintaining continuity of communication during the change at the next distance of the parameters of access of its equipment to the data transmission network (for example, the parameters "login" and "password").

The research adopted the following initial assumptions regarding the process of transferring information from the control point to the vehicle.

1. Information is transmitted via a composite data transmission channel, in which the following components are technically compatible:

virtual data transmission channel from the control point of the cargo carrier to the radio access point;

the radio communication channel from the radio access point to the on-board equipment of the vehicle, which is located in the direct radio visibility zone.

2. A virtual channel is understood as a means in a data transmission network based on packet switching, which, during the passage of a distance, ensures the transmission of information from the control point of the cargo carrier to the point of radio access.

3. A radio access point is understood as a remotely controlled radio station equipped with interfaces, on the one hand, with a virtual channel, on the other hand, with the on-board equipment of the vehicle, and which provides information reception from the control point and its transmission to the vehicle.

Examples of composite data transmission channels:

from the control point of the railway enterprise to the telecommunication equipment of the (unmanned) train;

from the control point of the carrier company to the telecommunication equipment of the (unmanned) vehicle;

from the control point or from the station of an external pilot of an aviation company to the telecommunications equipment of an (unmanned) aircraft.

II. Analysis of Models for Intelligent Management

In this research, models for intelligent management telecommunications are understood as models of managing information and telecommunications systems developed using the cognitive efforts of subjects and modern information technologies for structuring semantic knowledge bases and network management of processes, for example [10-12]. Examples of such models are intellectual property objects – digital platforms to support the activities of organizational systems with capabilities:

(a) automatic development and application of solution scenarios to prevent the negative impact of "hidden problems" – possible problem situations (patent RU2744296C1); in this research, the "hidden problem" is the threat of a break in communication between the control point and the vehicle;

(b) automatic recognition of the address of the incoming system based on the "need for information", the formation of a data transmission channel and the transmission of the required information from the outgoing system (patent RU2784715C1); in this study, "need for information" is the need to control the movement of a vehicle.

The analysis of the entities of the subject area related to the solution of the problem was carried out on an array of scientific articles, including [13-19] and patents RU2715104C1, RU2767605C1, RU2769359C2, RU2805539C2, RU2814573C1, taking into account logical judgments (3) – (7).

In the analysis carried out, attention is drawn to the influence exerted on the development of novel models by external, in relation to the considered, information transfer processes that accompany the logistics and track processes of transport logistics (Fig. 1), business processes of other enterprises operating in the areas of the transport corridor.

External information transmission processes use the productive resources of enterprises providing information and telecommunication services in the areas of the transport corridor.

At the same time, the number of productive resources may be limited due to the following circumstances (8) – (11).

< The development of the economy in the areas of the transport corridor is accompanied by an increase in the number of users of information and telecommunication services. Resources in the areas of the transport corridor >. (8)

⇩

< Increasing the information load on radio access points and on the data transmission network in the areas of the transport corridor >. (9)

⇩

< The incommensurability of user needs in terms of the permissible time of information transmission with the growth rate of information load and productivity of information and telecommunication resources in most of the areas of the transport corridor >. (10)

⇩

< When developing novel models for intelligent management telecommunications, there is a need to justify the requirements for the data transmission channel to ensure timely transmission time information formulated in contractual agreements >. (11)

Based on the results of the analysis, logical judgments (3) – (7) and circumstances (8) – (11), the following works were performed in the research.

1. A typical model of telecommunications architecture in transport logistics for unmanned aircraft systems has been developed that meets the requirements of the international standard [20].

2. The closest analogue (prototype) for the novel models for intelligent management telecommunications of a technical solution for preparing a scenario to prevent communication interruptions between the control point and the vehicle in the event of a threat from a "hidden problem"

has been determined. It is a "Digital platform for supporting organizational systems processes" – patent RU2744296C1.

3. Based on the productive resources of the prototype (point 2), a novel information model for intelligent management telecommunications has been developed – data transmission from the control point (external pilot station) to the vehicle.

4. The closest analogue (prototype) for the new models for intelligent management telecommunications has been identified for a technical solution for the formation of a new virtual data transmission channel in the area of the transport corridor in the event of a threat from a "hidden problem". It is a "Digital platform for supporting organizational systems" – patent RU2784715C1.

5. Based on the productive resources of the prototype (point 4), a logical model for intelligent management telecommunications has been developed – data transmission from the control point (external pilot station) to the vehicle.

III. Typical Architectural Models for Telecommunications in Transport Logistics

In the analysis of well–known publications and technical solutions (section 2), the search was conducted of telecommunications schemes similar in function used to control the movement of "specific" means - railway, automobile and aviation in transport logistics.

As a result of the conducted research of analogies and differences in the requirements for telecommunications facilities due to the belonging of a vehicle to a particular type, the following provisions are formulated.

< The objectives of telecommunication facilities for the implementation of functions in the processes of information transmission are similar in the "specific" requirements >; (12)

< The parameters of telecommunication facilities *can be different* >. (13)

Based on the provisions (12) and (13), the structural transformation of well-known telecommunications models in transport logistics into a single architectural form was carried out and standard models of telecommunications architecture in transport logistics were developed.

Fig. 3 shows an example of a typical telecommunications architecture model built for remote control of aircraft flight in accordance with the general requirements of the international standard [20].

In Fig. 2 is marked:

T1, T3, …, T2N-1 – the time of entry of aircraft V1, V2, …, VN into the radio visibility zone, respectively, of points R1, R2, …, RN of radio access;

T2, T5, …, T2N – the time of departure of aircraft V1, V2, …, VN from the radio visibility zone, respectively, points R1, R2, …, RN of radio access.

At the same time, various options for radio access modes are acceptable.

A logical understanding of the typical model of telecommunications architecture for unmanned aviation (Fig. 3) allowed us to formulate the following particular

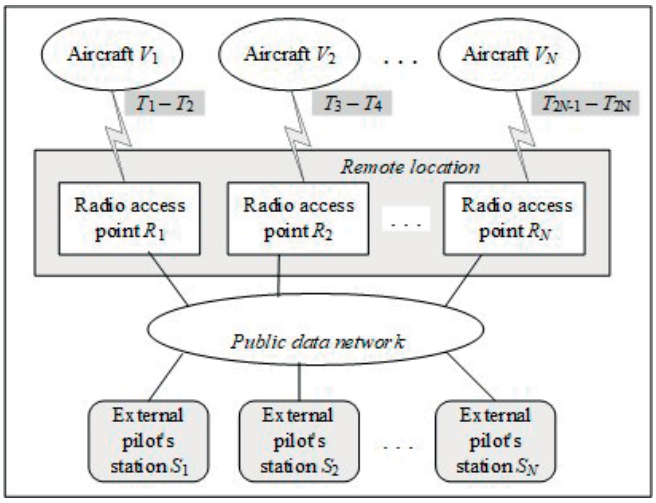

Fig. 3. A typical model of telecommunications architecture for unmanned aircraft systems.

statements that are important for the development of unmanned aviation transport:

< Each radio access point can be tuned to the same radio frequency when the conditions are met: $T_3 > T_2$; $T_5 > T_4$; …, $T_{2N} > T_{2N-1}$ >. (14)

< As a consequence of (14). It is enough to place one at each radio access point a radio station remotely controlled by one external pilot >. (15)

< As a consequence of (15). A significant reduction in telecommunications, information and management resources in aviation transport logistics >. (16)

IV. Methodological Approach to Intelligent Management Telecommunications in Logistics

The following models, technological and technical solutions were used in the selection of essential components in the novel models for intelligent management telecommunications and in determining information flows between components and with the external environment.

1. The architecture of a typical telecommunications model in transport logistics (Fig. 3) – as a sample for displaying.

2. Technological solutions to ensure compatibility between their information systems of organizational systems, for example [21] – for conditions of dynamic parameters of access to data transmission networks in order to ensure continuity of communication between control points and vehicles.

3. A digital platform model with the ability to automatically develop a solution scenario to prevent a negative impact on business processes, for example patent RU2744296C1 – as a sample when creating a digital platform for intelligent management telecommunications with the ability to develop and apply a solution scenario to prevent a break in communication between the control point and the vehicle.

4. A digital platform model with the ability to automatically determine the address of the incoming system and transfer data to it, for example patent RU2784715C1 –

as a sample when creating a digital platform for intelligent management telecommunications with the ability to automatically determine the radio access point, form a virtual channel and transmit data from the control point (external pilot station) over it.

In order to methodically study the issues of choosing essential components and their relationships and determining information flows, the issue of applying the well-known methodology of adaptive management of an information system [8] to the subject area of this study was considered.

As a result, it seems possible to apply in the study those provisions of the methodology [8] that relate to its target settings. As a result, a novel methodic approach to intelligent telecommunications management in transport logistics has been developed.

In Fig. 4 shows the purpose and objectives of the methodic approach to intelligent telecommunications management in transport logistics. The purpose of this methodic approach is to develop novel information and logical models for intelligent management telecommunications in transport logistics.

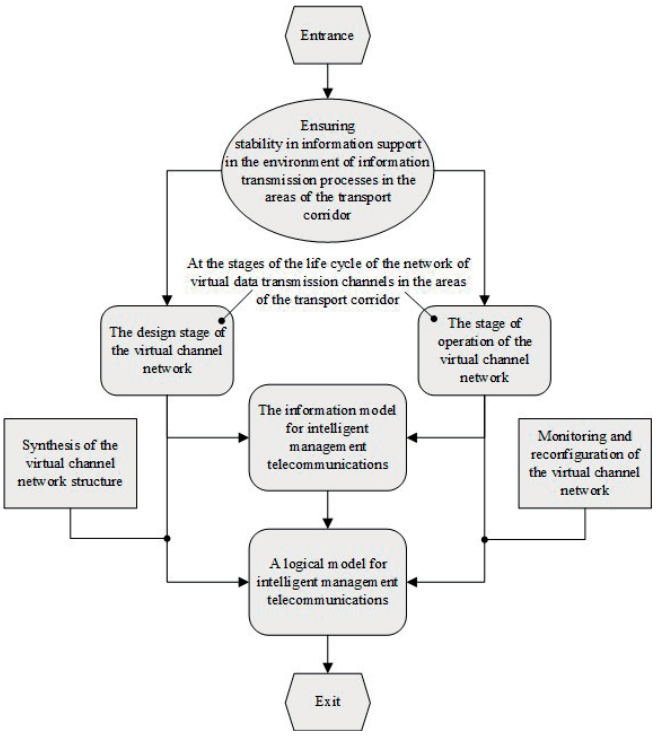

Fig. 4. Purpose and objectives of the methodic approach to intelligent management telecommunications.

V. INNOVATIVE SOLUTIONS FOR TELECOMMUNICATIONS IN LOGISTICS TRANSPORTATION

A. The information model for intelligent management telecommunications

The information model for intelligent management telecommunications at transport corridor distances was developed using a methodological approach (Fig. 4) and characterizes the composition of essential components used in:

synthesis of the network structure of virtual data transmission channels in the transport corridor;

reconfiguration of this network at the operational stage, depending on the location of vehicles on the distances of the transport corridor.

The block diagram of the intelligent control information model is shown in Fig. 5.

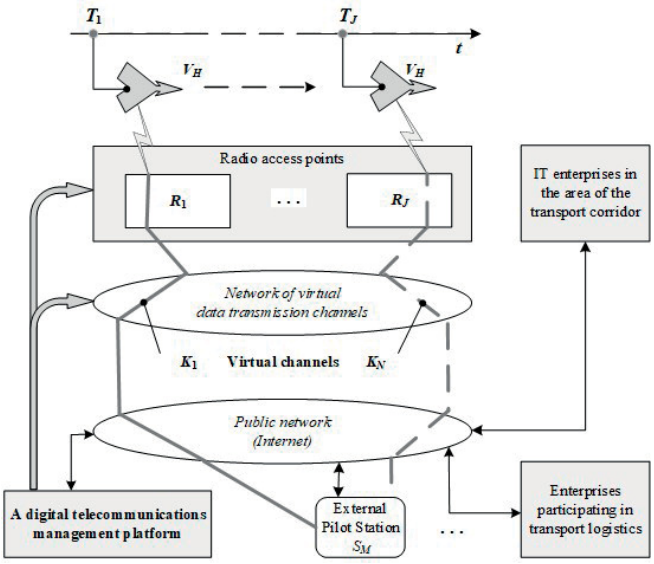

Fig. 5. Block diagram of the information model for intelligent management telecommunications.

The rule for reconfiguring virtual channels in the data transmission network of the transport corridor is shown in Fig. 5 in relation to the VH vehicle as follows:

< By the time T_1, a virtual channel K_1 is formed to transmit information from the station SM of the external pilot to the vehicle VH and back >. (17)

< By the time T_J, a virtual channel K_N is formed to transmit information from the station S_M of the external pilot to the vehicle V_H and back >. (18)

The rule < (17) ∩ (18) > of reconfiguration of virtual data transmission channels corresponds to all vehicles and stations of external pilots (control points) that control the movement of vehicles over the distances of the transport corridor

In general, this rule is structured in the form of a logical model for intelligent management telecommunications, presented in section 6.

The table:

shows a brief description of the components in the information model for intelligent management telecommunications (Fig. 5), essential in relation to the information transmitted from the stations of the external pilot (control points) to vehicles at the distances of the transport corridor, taking into account the time constraints on the availability of virtual data transmission channels;

examples of specific virtual channels K_1 and K_N of data transmission, the station S_M of the external pilot, the vehicle V_H and the time T_1 and T_J of the readiness of the channels for transmitting information from the station SM to the VH vehicle are given

Table. Characteristics of the essential components in the information model.

Essential components	Purpose (in relation to the information transfer process)	Active state (main fragments)
Virtual channel K_1.	Readiness to transmit data to the vehicle V_H via the radio access point R_1 by time T_1.	Receiving data with information from the station S_M of the external pilot and transmitting it to the vehicle V_H in the time period from T_1 to T_J.
Virtual channel K_N.	Readiness to transmit data to the vehicle V_H via the radio access point R_J by time T_J.	Receiving data with information from the station S_M of the external pilot and transmitting them to the vehicle V_H from time T_J.
External pilot stations and control points (example – station S_M).	Readiness to receive control information from enterprises participating in transport logistics, its structuring, analysis, formation of a solution scenario and transfer of control information.	Transmission of control information to a vehicle via virtual data transmission channels (for example, transmission of control information to a vehicle V_H).
A digital platform for telecommunications management.	Readiness to reconfigure virtual channels in the data transmission network in accordance with the requirements for the time of network reconfiguration (examples – T_1 and T_J) and with the locations of vehicles at the distances of the transport corridor.	Receiving data with control information about the locations of vehicles from enterprises participating in transport logistics, and about the resource parameters of IT enterprises in the areas of the transport corridor.
IT enterprises in the areas of the transport corridor.	Readiness to transfer control information about the parameters of its resources to a digital telecommunications management platform.	Transfer of control information about the parameters of its radio communication resources and data transmission networks to a digital telecommunications management platform.
Enterprises participating in transport logistics.	Readiness to transfer control information about the locations of vehicle at the distances of the transport corridor to a digital telecommunications management platform.	Transfer of control information about the locations of vehicle at the distances of the transport corridor to a digital telecommunications management platform.

B. The logical model for intelligent management telecommunications in a logistics project

Based on

the conducted cognitive analysis of the characteristics of the essential components in the information model for intelligent management telecommunications (Section 5.1),

using a methodological approach to intelligent telecommunications management in logistics (Section 4),

and guided by the rule < (17) = (18) > (Section 5.1) on the configuration of virtual data transmission channels, a typical logical model for intelligent management telecommunications has been developed for use in a logistics project at transport corridor distances.

The term "typical model" refers to a property of the logical model for intelligent management that allows it to be replicated for use in other logistics projects when managing the movement of their funds in the same transport corridor.

The logical model, along with the information model, is the basis on which data processing processes are built in a digital platform (Fig. 5) for telecommunications management.

This model is shown in Fig. 6 and represents a formal structure with connections of essential components in an information model with operations (functions) reproduced in a digital platform.

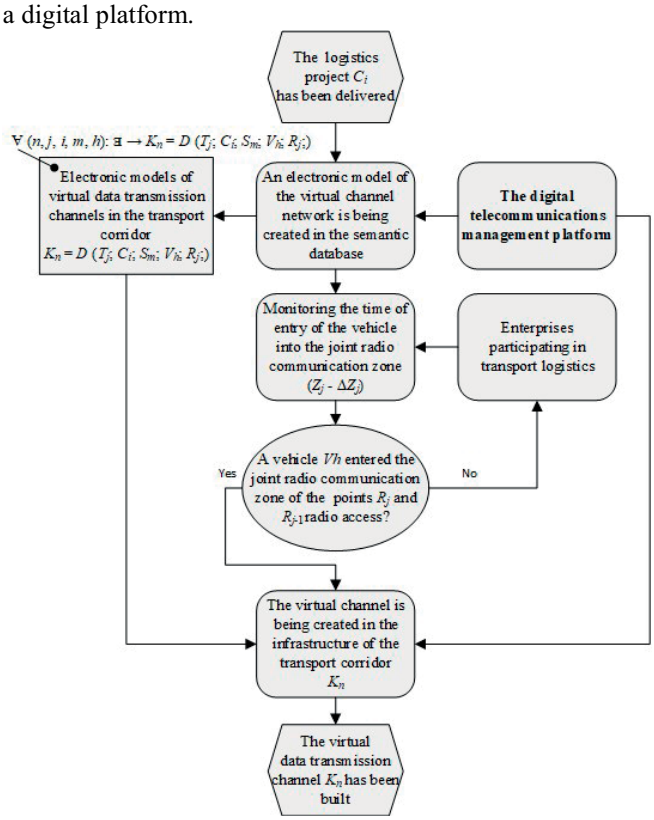

Fig. 6. Block diagram of the logical model for intelligent management telecommunications.

Based on a survey of the semantic content of the logical model for intelligent management telecommunications (Fig. 6) its essential property is determined and the formula (17) of the model is developed:

essential property: the control object in the logical model is the data transmission channel K_n between the

station Sm of the external pilot (control point) and the vehicle V_h,

running in the transport corridor through the radio access point R_j,

created in the logistics project C_i,

based on a digital telecommunications management platform,

by transformations over the operator D (Tj; Li; Sm; Vh; Rj;),

performed from the time Tj of the vehicle's entry into the radio communication zone (Zj - ΔZj).

The formula:

$$\forall (n, j, i, m, h): \exists \rightarrow K_n = D(T_j; C_i; S_m; V_h; R_j;),$$

$$T_j = F(Z_j - \Delta Z_j), \quad (17)$$

Where

Z_j – coordinates of the radio access point R_j;

ΔZ_j – radius (distance) from the radio access point R_j;

(Z_j - ΔZ_j) – a joint radio communication zone with vehicle access to the Z_j and Z_{j-1} radio access points in the transport corridor.

VI. Conclusion

The research of the information transmission processes in the conduct of logistics projects in the areas of the transport corridor equipped with radio access points to data transmission networks of enterprises doing business in these areas, including IT enterprises. The features of the infrastructure of these areas for the stage of digitalization of the data economy, which is characterized by the expansion of the use of logistics, track, information and telecommunication processes, are considered.

The subject area of the research is defined – the processes of information transfer between control points for the movement of vehicles and their on-board telecommunications equipment in the conditions of the stage of digitalization of information and with the unity of approaches to railway, automobile and aviation modes of transport.

It was found that the technical reality at the distances of the transport corridor is the heterogeneity of the parameters of vehicle access to data transmission networks. This reality made it possible to identify the "hidden problem" of the formation of a break in communication between the control point and the vehicle, at least for the time of making changes to the access parameters.

The tasks of preventing the manifestation of this problem have been set and solved. This is a scientific task – to ensure the continuity of communication between the control point and the vehicle during its passage along the distances of the transport corridor equipped with radio access points, regardless of the difference in the parameters of access to data transmission networks set in them, and a scientific and technical task - the development of novel information and logical models for intelligent management telecommunications.

The structural transformation of well-known telecommunication models in transport logistics into a single architectural form has been carried out and typical models of telecommunications architecture in transport logistics have been developed. The article provides, as an example, a typical model of telecommunications architecture for unmanned aircraft systems (Fig. 3).

A novel methodic approach to intelligent management telecommunications in logistics has been developed (Fig. 4). The application of this approach has allowed to carry out the necessary research and achieve a scientific result – to develop novel information and logical models for intelligent management telecommunications in transport logistics.

A novel information model for intelligent management telecommunications has been developed (Fig. 5). The model characterizes the components and information flows characterizing the subject of the research.

A novel logical model for intelligent management telecommunications has been developed (Fig. 6). The model is a formal structure with connections of essential components in the information model and with operations (functions) reproduced in the digital platform.

The primary effect of using novel information and logical models is to ensure the continuity of aircraft traffic control in conditions of heterogeneity of access parameters to data transmission networks in the areas of the transport corridor. The effect is achieved by timely preventing the negative impact of "hidden problems" caused by heterogeneity of access parameters.

References

[1] Ekonomika dannih i tsifrovaja transformatsija gosudarstva (natsionalniy proekt) [Data economy and digital transformation of the state (national project)]. In: TADVISER. State. Business. Technologies [Electronic resource], publ. 2024/05/18. URL: https://www.tadviser.ru/.

[2] Kolbanev, O.M., I.I., Palkin, I.I., Tatarnikova, T.M. Date. In: Gidrometeorologiya i Ekologiya. Russian J. of Hydrometeorology and Ecology (Proceedings of the Russian State Hydrometeorological University). **59**, 124–136 (2020). URL: https://doi.org/10.33933/2074-2762-2020-59-124-136.

[3] Borisova, V.V., Borodina, A.S. Digital transport corridors: opportunities to enter new markets. In: Proceeding of the St. Petersburg State University of Economics. **139** (1), 96–100 (2023).

[4] Golovnich, A.K. Modeling the interaction of railway and marine transport within digital transport corridors of the Arctic regions of Russia. In: Ekonomika Severo-Zapada: problemy i perspektivy razvitiya (Economy of the North-West: problems and prospects of development). **75** (4), 76–80 (2023). URL: https://doi.org/10.52897/2411-4588-2023-4-76-80.

[5] Lukashevich, A.A. Tsifrovaya transformatsija logisticheskikh biznes-protsessov v sovremennoy neftegazovoy otrasli posredstvom IT-importozamescheniya (Digital transformation of logistics business processes in modern oil and gas industry through IT import substitution). In: Ekonomika, predprinimatelstvo i parvo. **14** (3), 757–768 (2024). URL: https://doi.org/10.18334/epp.14.3.120591.

[6] Filippova, N.A., Abakarov, A.A., Amirov, A.T., Igitov, Sh.M. Improving Transportation and Traffic Safety with Digital Infrastructure and Telematic Systems for Mountain Vehicle Operations. In: World of Transport and Transportation. **21** (4), 207–216 (2023). URL: https://doi.org/10.30932/1992-3252-2023-21-4-7.

[7] Nikiforova, G.I. Research of Informational Interaction Between Railway and Sea Transport in Logistic Chains of Cargo Delivery. In: Proceedings of Petersburg Transport University. **19** (1), 82–89 (2022). URL: https://doi.org/10.20295/1815-588X-2022-1-82-89.

[8] Shabanov, A.P. Adaptive Control Axe: "Information System - Organizational Structures of Queuing". In: Business Informatics. **13** (3), 19–26 (2010). URL: https://www.elibrary.ru/download/elibrary_15518265_57332894.pdf.

[9] Zatsarinnyy, A.A, Shabanov, A.P. Methodological approach to quality control of data transformation in an interdisciplinary digital platform. In: Systems and Means of Informatics. **33** (3), 129–140 (2023). URL: https://doi.org/10.14357/08696527230311.

[10] Liu, J., Yan, J. Filling structural holes? Guanxi-based facilitation of knowledge sharing within a destination network. In: Journal of Organizational Change Management. **35** (2), 264–279 (2021). URL: https://doi.org/10.1108/JOCM-11-2020-0358.

[11] Stock, G.N., Tsai, J.C.A., Jiang, J.J., Klein G. Coping with uncertainty: Knowledge sharing in new product development projects. In: International Journal of Project Management. **39** (1), 59–70 (2021). URL: https://doi.org/10.1016/j.ijproman.2020.10.001.

[12] Liu, L., Zhao, M., Fu, L., Cao, J. Unraveling local relationship patterns in project networks: A network motif approach. In: International Journal of Project Management. 39 (5), 437–448 (2021). URL: https://doi.org/10.1016/j.ijproman.2021.02.004.

[13] Golub, I.V., Sergeev, S.M. Influence of data economy on information flow of logistics networks. In: Ekonomika i menegement sistem upravlenija [Economics and management of management systems]. **50** (4), 23–28 (2023). URL: https://www.elibrary.ru/item.asp?edn=wngytc&ysclid=m23ccqmhqz399736491.

[14] Yakushev, D. Digital model of the railway for automatic train operation. In: Automation, Communications, Informatics. **4**, 35–39 (2021). URL: https://doi.org/10.34649/AT.2021.4.4.005.

[15] Lyakhovenko, Yu., Popov, I. Model of information interaction of elements of the system unmanned rail way transport. In: Information and Space. **1**, pp. 84–92 (2023). URL: https://www.elibrary.ru/item.asp?id=50485733.

[16] Kozlov, S.V., Shabanov, A.P., Kubankov, A.N. Unmanned aircraft systems in logistics processes with network management. In: Wave Electronics and Its Application in Information and Telecommunication Systems. **6** (1), 1–7 (2023). URL: https://doi.org/10.1109/WECONF57201.2023.10148019.

Optimizing Parameters for Blockchain Storage for the RunOS SDN Controller

Viktor Piskovski*, vpiskovski@lvk.cs.msu.ru,
Pavel Shibaev*, pshibaev@lvk.cs.msu.ru,
*Lomonosov Moscow State University,
Computational Mathematics and Cybernetics Department,
Moscow, Russia

***Abstract*—This paper explores the use of blockchain technology to increase the reliability and fault tolerance of data storage in applications for the RunOS Software Defined Networking (SDN) controller. An overview of existing bapproaches and blockchain technologies has been conducted, and a theoretical model for integrating blockchain with RunOS has been developed. An experimental stand has been created, where tests have been carried out to assess the performance of the proposed solution. Recommendations on the optimal configuration as well as the model to obtain approximate optimal parameters for the Tendermint consensus algorithm are proposed.**

***Index Terms*—blockchain, data storage, RunOS, SDN, network controller, Tendermint**

I. Introduction

In recent years, blockchain technology has gained significant popularity due to its ability to provide reliable and secure data storage. The blockchain is a distributed database that stores information in the form of a chain of blocks, each of which is protected by cryptographic methods. This technology is widely used in various fields, including finance, logistics and healthcare.

One of the promising areas of blockchain application is to ensure fault tolerance and reliability of network applications. In particular, this paper discusses the use of blockchain for storing application data of the RunOS network controller. RunOS is a programmable platform for creating network applications that allows to flexibly manage network resources and quickly respond to changes in the network.

The main purpose of this study is to demonstrate the potential of using blockchain as a fault-tolerant data storage mechanism for RunOS applications. To achieve this goal, an experimental stand consisting of a RunOS controller and a blockchain network was assembled. Tests were carried out at this stand, which made it possible to identify the most effective configuration of the blockchain network consensus algorithm, as well as to obtain data on the functional relationship between the transaction flow rate, the average transaction size and the proportion of rejected transactions.

As part of the study, a review of the subject area was conducted, including existing work in the field of SDN (Software-Defined Networking) and blockchain technologies. An experimental setup and test procedure are described, experimental results are presented, and a model is developed to select the optimal values of the blockchain configuration based on the data obtained.

This work contributes to understanding how blockchain can be integrated into network applications to increase their reliability and fault tolerance, and offers practical recommendations on the use of this technology in network controllers.

II. Problem statement

There is a self-learning switch of the second level (l2-learning-switch, hereinafter - L2 LS) in the form of an application for the RunOS network controller and a blockchain for storing application data. The task is to identify dependencies that would allow to configure the blockchain network (in particular, the consensus parameters in this network) in an optimal way.

To achieve this goal, we define the following variables:

- throughput of the blockchain network (T) — the number of transactions entering the distributed registry, per unit of time;
- transaction flow rate (f) — the number of transactions received by validator nodes during the experiment per unit of time;
- share of rejected transactions ($Q = 1 - \frac{T}{f}$) — share of transactions per unit of time that entered the network but are not included in blocks;
- average transaction size (s) — average transaction size in the transaction stream;
- configuration of the consensus algorithm ($C_i, i \in 1, 2, 3$) there is a tuple (b, p, τ), b — the speed of the byte stream of transaction transmission, p — the number of permanent connections, τ — interval voting; the configuration options presented in the table I are considered.

Consensus algorithms configurations are based on the priori knowledge acquired from running blockchain networks in production environments. A similar set of variables has proven to be efficient in consensus algorithm performance assessment [1].

It is required to identify the dependencies $T(C_i, f)$ and $Q(C_i, f)$, $i \in 1, 2, 3$. For a better configuration, investigate the dependencies $T(s, f)$, $Q(s, f)$. Tabular data on these dependencies allows us to approximate the functions $f(s, Q)$ and $s(f, Q)$ in order to find a solution to the following problems. The first task is to maximize f for a given $s = s_0$

TABLE I
POSSIBLE CONFIGURATIONS OF THE CONSENSUS ALGORITHM

Parameter	**Option 1** (C_1)	**Option 2** (C_2)	**Option 3** (C_3)
b, Mb/s	6	1	6
p	15	10	10
τ, s	4	15	15

and a given limit on the proportion of rejected transactions Q_0:

$$\begin{cases} max\ f(s_0, Q) \\ Q \leq Q_0 \end{cases} \quad (1)$$

The second task, which does not depend on the first, is to maximize s for a given $f = f_0$ with a given limit on the proportion of rejected transactions $Q \leq Q_0$.

$$\begin{cases} max\ s(f_0, Q) \\ Q \leq Q_0 \end{cases} \quad (2)$$

III. SURVEY

It was made based on the following goals and criteria:

- features of data storage in blockchain storage compared to a centralized database (availability of a trusted party, data confidentiality, reliability and fault tolerance, redundancy, security);
- integration of SDN controllers and blockchain systems (controller, topology construction method, blockchain technology used);
- consensus algorithms (synchronization, Byzantine fault tolerance, scalability, throughput, transaction finalisation).

A. Features of data storage in blockchain storage

One of the main advantages of blockchain storage compared to centralized databases is the absence of the need for a trusted third party. This is achieved due to the decentralized nature of the blockchain, where each node of the network stores a copy of the entire block chain. The main aspects covered in this part of the review include:

- **Data Privacy**: Blockchain provides mechanisms to ensure data privacy, which is especially important in network applications.
- **Reliability and fault tolerance**: Thanks to decentralization, the blockchain provides a high degree of reliability and fault tolerance of data.
- **Backup**: All nodes of the network contain a complete copy of the blockchain, which ensures reliable data backup.
- **Security**: The use of cryptographic algorithms and consensus mechanisms ensures a high level of data security.

In practice, the preference in favor of one or another method of storing the data of the SDN controller is chosen depending on the system requirements. The choice in favor of blockchain storage should be made in case of increased requirements for data confidentiality, reliability, redundancy and security, provided that a decrease in performance is acceptable.

TABLE II
COMPARISON OF APPROACHES: BLOCKCHAIN AND CENTRALIZED DATABASE

Comparison criteria	**Blockchain**	**Central Database**
Trusted Party	Can work without a trusted party	Requires a trusted party
Data privacy	All nodes have access to all data	One can set access levels for data
Reliability/ fault tolerance	The data is distributed between the nodes	The data is stored in the central database
Efficiency	It takes time to reach consensus and complete the transaction	No overhead for consensus
Reservation	Each node has a complete last copy	Some nodes may have an incomplete copy of the data
Security	Cryptographic tools are used	Traditional access control is used

B. Examples of integration of SDN and blockchain systems

In recent years, various approaches to the integration of software-configurable network controllers (SDN) with blockchain systems have been actively explored. These studies show that combining the flexibility and manageability of SDN with the reliability and security of the blockchain can significantly improve the efficiency and stability of networks. Examples of successful integration include using the Ryu controller with the Ethereum blockchain, the POX controller with Ethereum, and ONOS with Multichain. The main aspects of such integrations, including the controller used, the way the topology is built, and the blockchain technology, are presented in the table III.

TABLE III
EXAMPLES OF INTEGRATION OF PC CONTROLLERS AND BLOCKCHAIN SYSTEMS

	A. Rahman, Md. Nashidul Islam [8]	**Praveen Fernando, JinWei [9]**	**P.K. Sharma, S. Singh, S. Jang, J.H. Park [10]**	**S. Book ra, M. Geurroumi, I. Romdhani [11]**
Controller	Ryu	Ryu	POX	ON
Building a Topology	Mininet	Mininet	Mininet	Mininet
Type of Blockchain	Ethereum	Ethereum	Ethereum	Multichain

The table shows that Mini net is often used to build a topology, and Ethereum is a popular blockchain technology due to its support for smart contracts and wide distribution. However, depending on the specific requirements of the project, other combinations of controllers and blockchain systems can be used. For example, integration with ONOS and Multichain may be preferable for corporate networks that require a high level of security and access control.

C. Consensus algorithms

The results of the review of consensus algorithms are presented in the table IV. In the process of analyzing various consensus algorithms for blockchain applications used in SDN network controllers, the Tendermint algorithm stood out as the most suitable option to achieve the goals of this study. One of the key advantages of Tendermint is its asynchronous operation model, which ensures the stability of the system in conditions of high network load and instability, which is critically important for applications in RunOS network controllers. The algorithm is also able to withstand up to 1/3 of malicious nodes, ensuring reliability and security in network conditions with possible attacks.

TABLE IV
CONSENSUS ALGORITHMS COMPARISON

	PoW [2]	PoS [3] [4]	PBFT [5]	Tendermint [6]	Hashgraph [7]
Validators number scaling	High	High	Low	Med.	Low
Tx. number scaling	Low	Variable	High	High	High
Through-put	Low	Med. / High	High	High	High
Tx. finality	Probabi-listic within 10 min	Several seconds	1–3 s	100ms–3 s	Several seconds

Tendermint's scalability and throughput make it suitable for use in high-load SDN applications where fast processing and reliable data storage are required. In addition, Tendermint provides fast transaction completion, which is critical to maintaining process continuity in real-world use cases.

D. Survey Conclusion

In the context of this work, where both the flexibility of setting the access level and high bandwidth are important to support the efficient operation of network applications, Tendermint (together with the Cosmos blockchain development tools) is the preferred choice. These aspects make it a suitable platform for the development and implementation of blockchain applications, especially in environments where both scalability and a private operating environment are required.

This work is based on the guarantees of security and fault tolerance of the Tendermint consensus algorithm and the Cosmos SDK library for creating blockchains.

IV. THE BLOCKCHAIN DEVICE–STORAGE AND APPLICATIONS FOR THE CONTROLLER

A. Blockchain Application architecture

The RunOS blockchain application follows the modular architecture of the Cosmos SDK, organized into several key components that are common to blockchain applications. The most important of them are shown in the figure 1. The actual

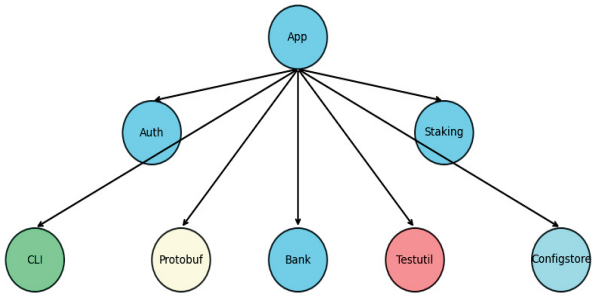

Fig. 1. Blockchain application architecture

application (app) contains the basic logic and parameters for the blockchain application. The main functionality of the blockchain is implemented here. The command module includes command line tools (Command Line Interface, CLI) for interacting with the blockchain. There are directories for the blockchain daemon (RunOS chain) and the REST server (RunOS chain REST) to interact with the blockchain via HTTP requests.

The Protobuf module contains definitions of Protocol Buffers for the blockchain. Protocol Buffers is a method of serialization of structured data, often used in blockchain applications to determine the structure of transactions, blocks and other data.

The L2 LS application stores the basic data structure presented in the listing 1 in the blockchain storage.

Listing 1
THE RECORD STORED IN THE BLOCKCHAIN

```
1 type Record struct {
2     dpid string,
3         mac string,
4         import string,
5 }
```

However, it is possible to switch to "flexible" untyped fields to quickly connect an arbitrary Runos application without implementing a separate type of stored data and recompiling the application. This data structure is presented in the listing

2. The disadvantage of this approach is the rejection of field type control.

Listing 2
THE STRUCTURE OF THE FLEXIBLE RECORD

```
type Flex Record struct {
    Flex Data
        map[interface{}]interface{}
}
```

The test util provides testing utilities, including assistants for working with the keeper (state management), network simulation and data sample generation.

A separate directory contains custom modules for blockchains based on the Cosmos SDK. These modules extend the functionality of the blockchain application with specific capabilities, such as token management, voting, or other domain-specific logic. In particular, the config store module has been implemented, which provides configuration management functionality in the blockchain. This may include processing blockchain parameters, management suggestions, or other configuration data.

B. Application architecture for RunOS

The include/L2LearningSwitch.hpp header file defines a class for L2LS that inherits from the Application class, indicating that it is a standalone application component within a larger system. The class is designed to interact with network switches, handle events such as connecting the switch to the network (onSwitchUp) and initializing the application with the necessary configurations (init).

The L2LearningSwitch class contains methods for initializing the application, responding to network events, and private members for managing network packets and interactions with the switch. The HostsDatabase class manages a database of MAC addresses of hosts and their associated ports on the switch. This functionality is important for the learning aspect of the switch, allowing it to remember where hosts are located.

The src/L2LearningSwitch.cc source file implements the functionality declared in L2LearningSwitch.hpp, providing the logic for L2LS to work. The file begins by registering the L2 Learning Switch application with dependencies such as the controller and switch manager. The Qt network access functions (for example, QNetworkAccessManager, QNetworkRequest) are used to interact with external services or databases to obtain and configure port information based on MAC addresses. The init method configures event listeners for switch-related events, and the onSwitchUp method configures new switches with the necessary flow rules for packet forwarding. Finally, the basic logic for studying MAC addresses and packet forwarding has been implemented. This includes processing incoming packets, updating the host database, and making forwarding decisions based on the information received.

V. EXPERIMENTAL STUDIES

In order to evaluate the performance of the blockchain storage at the experimental stand, a series of experiments were performed.

A. Description of the experimental stand

The architecture of the experimental stand is shown in the figure 2. The blockchain application is deployed in 16 instances on separate virtual machines. Each of the machines has Ubuntu 22.04 and Golang 1.21 OS installed. Each of the virtual machines has 2 cores and 4 GB of RAM. dlt01–dlt04 virtual machines are running on a cluster with an Intel Xeon E5645 processor. dlt04–dlt16 virtual machines are running on a cluster with an Intel Xeon E5-2630 v3 processor. The blockchain application is compiled using the Cosmos SDK version v0.50. There is also an instance of the RunOS network controller running the L2 Learning Switch application written in C++. The RunOS instance interacts with OpenvSwitch Mininet v2.2.2. Mininet — software for deploying virtual networks. In order to ensure application compatibility, a REST API server is written in the Go language, which is accessed by the L2 Learning Switch. In turn, the server sends requests to the gRPC node of the blockchain. On the blockchain tuples of the form $(dp\_id, eth\_addr, port)$ are stored. Test runs were carried out, which demonstrated the compatibility and correct operation of the stand components.

The initial deployment of the blockchain network was performed using Ansible scripts. Ansible is a machine configuration management system written in the Python3 programming language that implements part of the approaches of the “infrastructure as a code” class (“IaaC”).

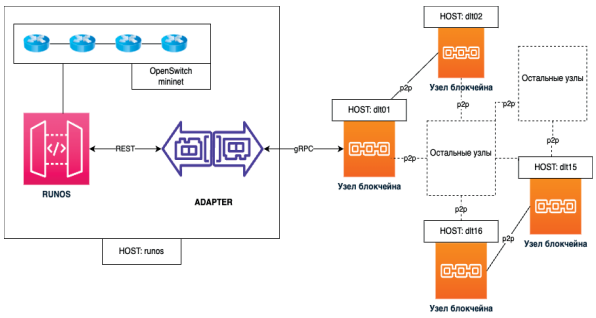

Fig. 2. The scheme of the experimental stand

B. Objectives of the experimental study

Three experiments are being conducted with the following goals:

1) compare the performance of Redis-based storage and blockchain storage with different number of nodes;
2) collect empirical data to identify the dependence of network bandwidth and the proportion of rejected transactions on the configuration of the consensus algorithm and the total flow of transactions entering the network;

3) collect empirical data to identify the dependence of network bandwidth and the proportion of rejected transactions on the average transaction size in the stream and the total flow of transactions entering the network.

C. Experimental research methodology

In all experiments, a utility based on tm-load-test was used to create write transactions. For the read operation, a CLI utility was used that calls the RunOS_chaind query module.

The first type of experiments consisted of a stream of read/write requests, first to a Redis node, then to one of the nodes of the blockchain network with a different number of nodes in the network. 10 series of 10000000 transactions were submitted at a rate of 10000 transactions per second.

The second type of experiments was performed with a blockchain network of 16 nodes. Three configurations of the consensus algorithm were investigated. Accordingly, there were three sub-series of experiments. In each of the sub-series, the network was restarted with a new configuration. For each unique set of (C, f), 10 launches of 10000000 transactions each were performed.

The third type of experiments was performed with a blockchain network of 16 nodes, in which the consensus algorithm had the best configuration selected in the previous experiment. The launches were performed at different average sizes of a single transaction (s) and different values of the total transaction flow (f) entering the network. For each unique set of (s, f), 10 launches of 10000000 transactions each were performed.

D. The results of the experimental study

Data on the speed of performing read and write operations on the Redis node and in the blockchain network with a different number of nodes are presented in the table V.

TABLE V
THE BANDWIDTH OF THE BLOCKCHAIN NETWORK WITH DIFFERENT NODE CONFIGURATIONS

Node	Read (pcs/s)	Write (MB/s)
Redis	12610	14290
4 nodes	11473	11904
8 nodes	10412	9917
16 nodes	9649	8418

The table VI presents empirical data on the dependence of $T(C_i, f)$.

TABLE VI
THE BANDWIDTH OF THE BLOCKCHAIN NETWORK WITH DIFFERENT CONFIGURATIONS OF THE CONSENSUS ALGORITHM

	Transaction generation rate (pcs/s)							
Conf.	**100**	**200**	**400**	**800**	**1600**	**3200**	**6400**	**12800**
C_1	92	136	252	472	816	224	320	640
C_2	91	144	260	496	880	384	512	1024
C_3	93	140	272	512	944	416	512	1024

The table VII presents empirical data on the dependence of $Q(C_i, f)$.

TABLE VII
THE PERCENTAGE OF REJECTED TRANSACTIONS IN DIFFERENT CONFIGURATIONS OF THE CONSENSUS ALGORITHM

	Transaction generation rate (pcs/s)							
Conf.	**100**	**200**	**400**	**800**	**1600**	**3200**	**6400**	**12800**
C_1	0.08	0.32	0.37	0.41	0.49	0.72	0.93	0.95
C_2	0.09	**0.29**	0.45	0.38	0.45	**0.69**	0.88	**0.92**
C_3	**0.07**	0.30	**0.32**	**0.36**	**0.41**	0.71	**0.87**	**0.92**

The table VIII presents empirical data on the dependence of $T(s, f)$.

TABLE VIII
THE THROUGHPUT OF THE BLOCKCHAIN NETWORK AT DIFFERENT TRANSACTION FLOW RATES AND AVERAGE TRANSACTION SIZES

	Transaction generation rate (pcs/s)							
Tx size, bytes	**100**	**200**	**400**	**800**	**1600**	**3200**	**6400**	**12800**
500	91	192	348	776	1504	3008	5696	12160
10^3	95	170	376	752	1488	2592	6016	11392
10^4	94	114	348	704	1504	2944	4544	4608
10^5	88	148	340	656	1040	1344	1664	2688
10^6	93	140	272	512	944	928	832	1024

The table IX contains empirical data on the dependence of $Q(s, f)$.

TABLE IX
THE PERCENTAGE OF REJECTED TRANSACTIONS AT DIFFERENT TRANSACTION FLOW RATES AND AVERAGE TRANSACTION SIZES

	Transaction generation rate (pcs/s)							
Tx size, bytes	**100**	**200**	**400**	**800**	**1600**	**3200**	**6400**	**12800**
500	0.09	0.04	0.13	0.03	0.06	0.06	0.11	0.05
10^3	0.05	0.15	0.06	0.06	0.07	0.19	0.06	0.11
10^4	0.06	0.43	0.13	0.12	0.06	0.08	0.29	0.64
10^5	0.12	0.26	0.15	0.18	0.35	0.58	0.26	0.21
10^6	0.07	0.30	0.32	0.36	0.41	0.71	0.87	0.92

The figure 3 shows a graphical illustration of the dependence of $Q(s, f)$.

VI. BUILDING A DEPENDENCY TO MATCH THE TRANSACTION FLOW RATE AND THE AVERAGE TRANSACTION SIZE IN THE STREAM

Applications for RunOS may have different network bandwidth requirements. To achieve optimal throughput without increasing the proportion of unconnected transactions, one can reduce the transaction size. On the contrary, if one reduces the speed of transaction generation, one can transfer larger transactions. Multidimensional approximation based on the data obtained above allows us to obtain the value of s at a known value of f and vice versa, imposing certain quality requirements of Q. These considerations are formally reflected in the problem statements (1) and (2).

To approximate their solution, one can use empirical data from the table VIII. Using to the data from the table, approximations are made using the two-dimensional spline method

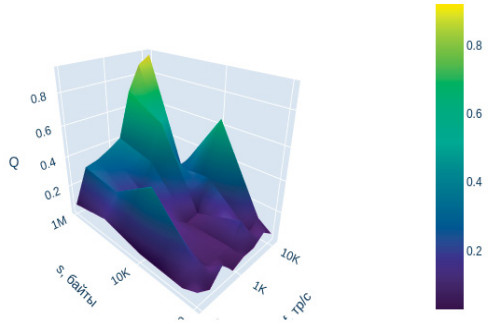

Fig. 3. Approximation of dependence $Q(s, f)$

on a rectangular grid. The implementation from the scipy intrepolate package is used. Next, an objective function with a condition is set (examples of functions are shown in listing 3 and listing 4), and an approximate numerical solution is found using the scipy optimize package.

Listing 3
FUNCTION FOR (1)

```
def objective(f):
    Q_val = Q_interp(s0, f)
    if Q_val >= Q0:
        return -f
    return np.inf
```

Listing 4
FUNCTION FOR (2)

```
def objective(s):
    Q_val = Q_interp(s, f0)
    if Q_val >= Q0:
        return -s
    return np.inf
```

It is worth noting that the found value is a reference value and should serve as a starting point for calibration of the stand, therefore, strict requirements are not imposed on the accuracy of the solution.

VII. CONSLUSION

The work done allowed us to obtain the following results:

- based on the results of the review the Tendermint consensus algorithm was selected;
- based on the specifics of the subject area, two independent mathematical problems were formulated ((1) and (2));
- based on the technical specifications of RunOS and Tendermint, a blockchain application has been developed for storing arbitrary data by applications of the RunOS controller;
- experiments were conducted on the developed stand in order to study the performance of the blockchain network; Based on the analysis of the experimental results obtained, a method has been developed for the approximate solution of the set mathematical problems ((1) and (2)), which is based on spline approximation.

The Tendermint consensus algorithm has shown resistance to exponential growth of the transaction flow with an average size of these transactions up to 10,000 bytes. At the same time, the most stable configuration in terms of the percentage of rejected transactions is the C_3 configuration (transaction transfer rate of 5 Mb/s, 5 permanent connections, voting interval of 10 seconds). It can be used in applications that generate up to 3200 transactions per second with a transaction size of up to 1 MB. This configuration should be used as a reference in further research.

REFERENCES

[1] J. Karamachoski and L. Gavrilovska, "Tendermint Performance with Large Transactions: The Healthcare System Scenario," 2021 International Balkan Conference on Communications and Networking (BalkanCom), Novi Sad, Serbia, 2021, pp. 85-89, doi: 10.1109/BalkanCom53780.2021.9593267.
e

[2] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," 2008.

[3] M. Syed and Z. Ul Abadin, "A Pattern for Proof of Stake Consensus Algorithm in Blockchain," *Proceedings of the 27th European Conference on Pattern Languages of Programs*, pp. 1-5, 2022.

[4] P. Schaaf, F. Rezabek, and H. Kinkelin, "Analysis of Proof of Stake flavors with regards to The Scalability Trilemma," *Network Architectures and Services*, 2021.

[5] M. Castro and B. Liskov, "Practical byzantine fault tolerance," *OsDI*, vol. 99, no. 1999, pp. 173-186, 1999.

[6] E. Buchman, "Tendermint: Byzantine fault tolerance in the age of blockchains," University of Guelph, 2016.

[7] M. Alahmad, I. Alshaikhli, A. Alkandari, A. Alshehab, M. R. Islam, and M. Alnasheet, "INFLUENCE OF HEDERA HASHGRAPH OVER BLOCKCHAIN," *Journal of Engineering Science and Technology*, School of Engineering, Taylor's University, 2022.

[8] Md A. Rahman and Md Z. Islam, "A hybrid clustering technique combining a novel genetic algorithm with K-Means," School of Computing and Mathematics, 2014.

[9] P. Fernando and J. Wei, "Blockchain-Powered Software Defined Network-Enabled Networking Infrastructure for Cloud Management," *2020 IEEE 17th Annual Consumer Communications & Networking Conference (CCNC)*, 2020.

[10] P. K. Sharma, S. Singh, Y. S. Jeong, and J. H. Park, "DistBlockNet: A Distributed Blockchains-Based Secure SDN Architecture for IoT Networks," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 78-85, 2017.

[11] S. Boukria, M. Guerroumi, and I. Romdhani, "BCFR: Blockchain-based Controller Against False Flow Rule Injection in SDN," *2019 IEEE Symposium on Computers and Communications (ISCC)*, 2019.

Научное издание

«Современные сетевые технологии — 2024»

Пятая международная научно-техническая конференция

Москва, Россия, 29–31 октября, 2024

Сборник трудов

Доклады сессии «Моделирование и анализ трафика гетерогенных каналов»

Стендовые доклады

Главный редактор
Р. Л. Смелянский

«Modern Network Technologies — 2024» (MoNeTeC)

The Fifth International Science and Technology Conference

Moscow, Russia, October 29–31, 2024

Proceedings

Reports of the session «Modeling and analysis of heterogeneous channel traffic»

Poster presentations

Editor-in-chief
R. L. Smelyansky

Художественное оформление *К. В. Саутенков*

Макет предоставлен Оргкомитетом конференции

Электронное издание сетевого распространения

Макет утвержден 16.01.2025. № 12944.

Издательство Московского университета

119991, Москва, ГСП-1, Ленинские горы, д. 1, стр. 15
Тел.: (495) 939-32-91; e-mail: secretary@msupress.com
https://msupress.com. Отдел реализации:
тел.: (495) 939-33-23; e-mail: zakaz@msupress.com