На правах рукописи

КРАЕВА Яна Александровна

МАСШТАБИРУЕМЫЕ МЕТОДЫ И АЛГОРИТМЫ ПОИСКА АНОМАЛИЙ ВО ВРЕМЕННЫХ РЯДАХ

2.3.5. — Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей

Автореферат
диссертации на соискание ученой степени
кандидата физико-математических наук

Челябинск — 2024

Работа выполнена на кафедре системного программирования
ФГАОУ ВО «Южно-Уральский государственный университет
(национальный исследовательский университет)»

Научный руководитель: ЦЫМБЛЕР Михаил Леонидович,
доктор физ.-мат. наук, доцент,
профессор кафедры системного программирования
ФГАОУ ВО «Южно-Уральский государственный
университет (национальный исследовательский
университет)» (г. Челябинск)

Официальные оппоненты: БАРКАЛОВ Константин Александрович,
доктор техн. наук, доцент,
заведующий кафедрой математического обеспечения
и суперкомпьютерных технологий Института
информационных технологий, математики
и механики
ФГАОУ ВО «Национальный исследовательский
Нижегородский государственный университет
имени Н.И. Лобачевского» (г. Нижний Новгород)

НИКИТЕНКО Дмитрий Александрович,
кандидат физ.-мат. наук,
ведущий научный сотрудник Лаборатории
параллельных информационных технологий
Научно-исследовательского вычислительного центра
ФГБОУ ВО «Московский государственный
университет имени М.В. Ломоносова» (г. Москва)

Ведущая организация: ФГАОУ ВО «Национальный исследовательский
университет ИТМО» (г. Санкт-Петербург)

Защита состоится 22 мая 2024 г. в 12:00 часов на заседании диссертационного совета 24.2.437.10 при Южно-Уральском государственном университете по адресу: 454080, г. Челябинск, пр. Ленина, 76, ауд. 1001.

С диссертацией можно ознакомиться в библиотеке Южно-Уральского государственного университета и на сайте:
https://www.susu.ru/ru/dissertation/24243710-d-21229818/kraeva-yana-aleksandrovna.

Автореферат разослан «_____» _____ 2024 г.

Ученый секретарь дис. совета Е.В. Иванова

Общая характеристика работы

Актуальность темы диссертационного исследования основывается на следующих основных факторах: всепроникающий характер временных рядов в современном информационном обществе и востребованность задачи поиска в них аномалий; наличие больших временных рядов в широком спектре приложений; массовое распространение кластерных вычислительных систем с графическими процессорами.

Всепроникающий характер временных рядов и востребованность задачи поиска аномалий. Временной ряд представляет собой хронологическую последовательность вещественных значений. В современном информационном обществе задачи анализа временных рядов возникают в широком спектре предметных областей: финансовое прогнозирование, персональное здравоохранение, спорт высших достижений, моделирование климата и предсказание природных катаклизмов, технологии Интернета вещей, цифровое производство и др. Поиск аномалий является одной из основных задач анализа временных рядов. В настоящее время научных исследования, посвященные поиску аномалий во временных рядах, существенно более востребованы по сравнению с темами поиска аномалий в других видах данных.

Большие временные ряды. В современных приложениях, связанных с обработкой и интеллектуальным анализом временных рядов, типичной является ситуация, когда анализу подлежат большие временные ряды (time series big data), которые не могут быть целиком размещены в оперативной памяти и требуют специализированных методов обработки и визуализации. В приложениях Интернета вещей и цифрового производства датчики киберфизических систем имеют высокую частоту (десятки–сотни раз в секунду) и продуцируют временные ряды, состоящие из сотен миллионов элементов. В указанных приложениях своевременное обнаружение аномалий позволяет заблаговременно уведомлять оператора технологического процесса и организовать предиктивное техническое обслуживание оборудования. Эффективный поиск аномалий в больших временных рядах фактически обуславливает необходимость применения параллельных алгоритмов и высокопроизводительных вычислений.

Массовое распространение кластерных вычислительных систем с графическими процессорами. В настоящее время увеличение количества вычислительных ядер является одной из основных тенденций развития процессорной техники, и соответствующие устройства содержат десятки тысяч ядер, поддерживающих векторную обработку данных. Одной из наиболее распространенных многоядерных архитектур являются графические процессоры (GPU, graphics processing unit) компании NVIDIA. GPU хорошо подходят для организации вычислений в соответствии с парадигмой SIMD (Single Instruction for Multiple Data), поскольку они состоят из симметричных потоковых мультипроцессоров, каждый из которых, в свою очередь, состоит

из ядер CUDA (Compute Unified Device Architecture). Интерфейс прикладного программирования CUDA позволяет назначить нескольким потокам выполнение одного набора инструкций над несколькими элементами данных. В настоящее время GPU стали де-факто стандартной платформой для обучения глубоких нейронных сетей. Одной из современных тенденций организации высокопроизводительных вычислений является объединение нескольких вычислительных узлов посредством высокоскоростной соединительной сети в один вычислительный комплекс, называемый кластером. В качестве узла кластера фигурирует центральный процессор, дополненный графическим процессором. Большинство систем, входящих в текущий рейтинг Топ50 суперкомпьютерных систем России, являются высокопроизводительными кластерами с графическими процессорами.

Исходя из изложенного, можно заключить, что *актуальным* является исследование, направленное на разработку методов и алгоритмов поиска аномалий во временных рядах на платформе современных высокопроизводительных вычислительных систем: массово распространенных графических процессоров и кластеров с узлами на их основе.

Степень разработанности темы. Кеог (Keogh) разработал концепцию диссонанса временного ряда и алгоритм HOTSAX для поиска диссонансов ряда в оперативной памяти. В отличие от предшественников, диссонанс имеет интуитивно более понятное определение и более точно находит аномалии. Кеог и др. представили алгоритм DRAG для поиска диссонансов ряда, который не может быть целиком размещен в оперативной памяти. Кеог и др. предложили алгоритм MERLIN, который выполняет поиск диссонансов произвольной длины. Кеогом и др. предложен алгоритм DAMP для обнаружения аномалий в потоковых временных рядах. Однако алгоритмы HOTSAX, DRAG, MERLIN и DAMP являются последовательными. Кроме того, Кеог предложил концепцию матричного профиля временного ряда, локальные максимумы которого указывают на диссонансы этого ряда. Однако вычисление всех матричных профилей ряда для подпоследовательности произвольной длины невозможно за приемлемое время, поскольку алгоритмы вычисления матричного профиля имеют квадратичную вычислительную сложность относительно длины ряда. Бониоль (Boniol), Палпанас (Palpanas), Линарди (Linardi) и Папарризос (Paparrizos) разработали последовательные алгоритмы SAD, NorM, NormA и др. для поиска аномалий временного ряда, а также оболочку TSB-UAD для оценки точности поиска аномалий. Цымблер и др. разработали параллельные алгоритмы поиска диссонансов, однако эти разработки не решают задачу поиска диссонансов произвольной длины и не предназначены для высокопроизводительных кластеров с графическими процессорами.

Исходя из рассмотренного выше, можно заключить, что на сегодняшний день задача разработки эффективных методов и алгоритмов поиска анома-

лий во временных рядах остается в фокусе интенсивных научных исследований и практических разработок.

Цель и задачи исследования. *Цель* данной работы состояла в разработке и исследовании новых методов и алгоритмов поиска аномальных подпоследовательностей временного ряда, основанных на концепции диссонанса, на платформе современных высокопроизводительных вычислительных систем. Данная цель предполагает решение следующих *задач*.

1. Разработать и исследовать следующие параллельные алгоритмы поиска диссонансов временного ряда:
   - поиск диссонансов фиксированной длины на графическом процессоре;
   - поиск диссонансов произвольной длины на графическом процессоре;
   - поиск диссонансов произвольной длины на высокопроизводительном вычислительном кластере с графическими процессорами.
2. Разработать нейросетевую модель для поиска аномалий в потоковом временном ряде на основе концепции диссонанса.
3. Провести вычислительные эксперименты с синтетическими и реальными временными рядами, подтверждающие эффективность предложенных методов и алгоритмов.

Научная новизна работы заключается в следующем:

1. Разработан новый параллельный алгоритм поиска диссонансов временного ряда, имеющих фиксированную длину, для графического процессора. В отличие от известных аналогов, алгоритм позволяет находить во временном ряде все диссонансы, которые имеют заданную длину.
2. Впервые разработан параллельный алгоритм поиска диссонансов временного ряда, имеющих произвольную длину, для графического процессора. Алгоритм позволяет находить во временном ряде все диссонансы, которые имеют длину в заданном диапазоне.
3. Впервые разработан параллельный алгоритм поиска диссонансов временного ряда, имеющих произвольную длину, для высокопроизводительного вычислительного кластера с графическими процессорами. Алгоритм позволяет находить во временном ряде, который не может быть целиком размещен в оперативной памяти вычислительного узла, все диссонансы, имеющие длину в заданном диапазоне.
4. Разработан новый метод поиска аномалий в потоковом временном ряде. В отличие от известных аналогов, метод позволяет выявлять аномальные подпоследовательности ряда, отражающие нетипичную и редко встречающуюся активности исследуемого субъекта.

Теоретическая значимость диссертационной работы состоит в том, что в разработанных алгоритмах поиска диссонансов временного ряда предложе-

на оригинальные схемы распараллеливания вычислений и организации данных, которые обеспечивают сокращение избыточных вычислительных операций и существенное увеличение быстродействия поиска.

Практическая значимость диссертационной работы заключается в том, что предложенные параллельные алгоритмы поиска диссонансов временного ряда не требуют обучения и разработаны для аппаратной платформы массово распространенных и доступных в настоящее время графических процессоров. В силу этого результаты исследования могут быть применены для эффективного поиска аномалий во временных рядах широкого спектра предметных областей при минимальном участии аналитика-эксперта (от которого требуется задать лишь длины искомых аномальных подпоследовательностей).

Методология и методы исследования. Проведенные в работе исследования базируются на методах машинного обучения и интеллектуального анализа данных, а также теории временных рядов. В реализации параллельных алгоритмов использованы методы параллельного программирования для графических процессоров и распределенной памяти на основе стандартов CUDA и MPI соответственно.

Положения, выносимые на защиту. На защиту выносятся следующие новые научные результаты:

1. Разработан параллельный алгоритм PD3 для графического процессора, позволяющий найти все диссонансы временного ряда, имеющие заданную длину.
2. Разработан параллельный алгоритм PALMAD для графического процессора, позволяющий найти все диссонансы временного ряда, имеющие длину в заданном диапазоне. Предложен алгоритм ранжирования диссонансов различной длины и способ их визуализации в виде тепловой карты диссонансов.
3. Разработан параллельный алгоритм PADDi для высокопроизводительного вычислительного кластера с графическими процессорами, позволяющий найти все диссонансы временного ряда (который не может быть целиком размещен в оперативной памяти одного узла кластера), имеющие длину в заданном диапазоне.
4. Разработан метод DiSSiD поиска аномалий потокового временного ряда, включающий в себя нейросетевую модель и алгоритм построения обучающей выборки для указанной модели. DiSSiD позволяет выявлять аномальные подпоследовательности ряда, отражающие нетипичную и редко встречающуюся активности исследуемого субъекта.
5. Проведены вычислительные эксперименты с синтетическими и реальными временными рядами, подтверждающие эффективность разработанных методов и подходов.

Степень достоверности результатов. Эффективность разработанных методов и алгоритмов подтверждена результатами вычислительных экспериментов над реальными и синтетическими данными, проведенными в соответствии с общепринятыми стандартами. Репродуцируемость результатов, полученных в исследовании, обеспечивается размещением исходных текстов программ, реализующих предложенные разработки, и наборов данных, с которыми проведены вычислительные эксперименты, в сети Интернет в свободно доступных репозиториях.

Апробация работы. Основные положения диссертационной работы, разработанные методы, алгоритмы и результаты вычислительных экспериментов докладывались на следующих международных и всероссийских научных конференциях:

— РСД'2023: Международная научная конференция «Суперкомпьютерные дни в России», 25–26 сентября 2023 г., Москва *(диплом II степени в конкурсе докладов молодых ученых)*;

— ПаВТ'2023: Всероссийская научная конференция с международным участием «Параллельные вычислительные технологии 2023», 28–30 марта 2023 г., Санкт-Петербург *(диплом II степени в конкурсе докладов молодых ученых)*;

— DAMDID'2023: International Conference on Data Analytics and Management in Data Intensive Domains, 25–27 October 2023, Moscow, Russia;

— ЦИСП'2023: Всероссийская научная конференция с международным участием «Цифровая индустрия: состояние и перспективы развития 2023», 21–23 ноября 2023 г., Челябинск;

— ПаВТ'2022: Всероссийская научная конференция с международным участием «Параллельные вычислительные технологии 2022», 29–31 марта 2022 г., Дубна *(диплом II степени в конкурсе докладов молодых ученых)*;

— DAMDID'2022: International Conference on Data Analytics and Management in Data Intensive Domains, 5–7 October 2022, St. Petersburg, Russia;

— ПаВТ'2021: Всероссийская научная конференция с международным участием «Параллельные вычислительные технологии 2021», 30 марта – 1 апреля 2021 г., Волгоград;

— ПаВТ'2020: Всероссийская научная конференция с международным участием «Параллельные вычислительные технологии 2020», 31 марта – 2 апреля 2020 г., Пермь;

— ЦИСП'2020: Всероссийская научная конференция с международным участием «Цифровая индустрия: состояние и перспективы развития 2020», 17–19 ноября 2020 г., Челябинск.

Публикации. Основные результаты диссертации опубликованы в *пяти научных работах* [1–5], в том числе 4 статьи в журналах, входящих в Ядро РИНЦ и категорию К1 Перечня ВАК (где одна из статей опубликована в журнале, который также входит в квартиль Q2 библиографической базы данных Scopus) и одна статья в журнале, входящем в квартиль Q1 библиографической базы данных Web of Science. Получено *два свидетельства Роспатента* о государственной регистрации программ для ЭВМ [6, 7].

Личный вклад. В статьях [1, 4, 5] научному руководителю М.Л. Цымблеру принадлежит постановка задачи, Я.А. Краевой — все полученные результаты.

Соответствие диссертации паспорту научной специальности. Тематика, содержание и полученные результаты диссертационного исследования Я.А. Краевой соответствуют следующим направлениям исследований паспорта специальности 2.3.5 «Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей»:
4. Интеллектуальные системы машинного обучения, управления базами данных и знаний, инструментальные средства разработки цифровых продуктов;
8. Модели и методы создания программ и программных систем для параллельной и распределенной обработки данных, языки и инструментальные средства параллельного программирования.

Структура и объем работы Диссертация состоит из введения, пяти глав, заключения, списков рисунков, таблиц, алгоритмов и приложения. Объем диссертации составляет 158 страниц, объем списка литературы — 151 наименование.

Содержание работы

Во введении приводится обоснование актуальности темы и степень ее разработанности; формулируются цели и задачи исследования; раскрываются новизна, теоретическая и практическая значимость полученных результатов; формулируется методологическая основа диссертационного исследования; дается обзор содержания диссертации.

В первой главе, «Современные методы поиска аномалий во временных рядах», рассматриваются современные методы и алгоритмы поиска аномалий во временных рядах. Дается классификация подходов к обнаружению аномалий: методы с привлечением учителя (supervised), методы без привлечения учителя (unsupervised) и методы с частичным привлечением учителя (semi-supervised). Описаны основные представители двух последних групп методов, поскольку в настоящее время методы с привлечением учителя, как правило, не применяются на практике. Приведены формальные определения и нотация для обозначения основных понятий, связанных

с обнаружением аномалий временных рядов, используемых далее в тексте. Дается обзор публикаций, наиболее близко относящихся к теме диссертации, которые охватывают последовательные и параллельные алгоритмы поиска диссонансов (аномальных подпоследовательностей) временного ряда.

Временной ряд T представляет собой хронологически упорядоченную последовательность вещественных значений длины n: $T = \{t_i\}_{i=1}^{n}$, $t_i \in \mathbb{R}$. *Подпоследовательность* $T_{i,m}$ представляет собой непрерывный промежуток из m элементов ряда T, начиная с позиции i: $T_{i,m} = \{t_k\}_{k=i}^{i+m-1}$, $3 \leqslant m \ll n$, $1 \leqslant i \leqslant N$, где $N = n - m + 1$.

Задача поиска аномальных подпоследовательностей ряда формализуется на основе концепции диссонанса следующим образом. Подпоследовательности $T_{i,m}$ и $T_{j,m}$ называют *не пересекающимися*, если $|i - j| \geqslant m$. Некая подпоследовательность ряда, не пересекающаяся с данной подпоследовательностью C, обозначается как M_C. Подпоследовательность D ряда T является *диссонансом*, если ее *ближайший сосед* M_D (ближайшая в смысле расстояния $\mathrm{Dist}(\cdot\,,\cdot\,)$ и не пересекающаяся с ней подпоследовательность) удален на расстояние не менее чем заданный аналитиком порог r: $\min\limits_{M_D \in T} \mathrm{Dist}(D, M_D) \geqslant r$.

Во второй главе, «Параллельный алгоритм поиска диссонансов фиксированной длины для графических процессоров», представлен новый параллельный алгоритм поиска диссонансов фиксированной длины PD3 (Parallel DRAG-based Discord Discovery) для графических процессоров. Рассмотрены основные принципы распараллеливания и параллельная реализация фаз алгоритма PD3. Представлены результаты вычислительных экспериментов на реальных и синтетических временных рядах, в которых исследуется производительность алгоритма PD3 в сравнении с известными аналогами.

Алгоритм включает в себя следующие три фазы, распараллеливаемые отдельно: предварительная обработка данных, отбор кандидатов и очистка диссонансов. На фазе предобработки данных вычисляются средние арифметические и стандартные отклонения подпоследовательностей ряда, которые далее используются на следующих фазах для вычисления расстояний. На фазах отбора кандидатов и очистки диссонансов выполняются соответственно формирование множества кандидатов в диссонансы и отбрасывание ложноположительных диссонансов из этого множества.

Для обеспечения лучшей производительности в качестве функции $\mathrm{Dist}(\cdot\,,\cdot\,)$ используется квадрат нормализованного евклидова расстояния:

$$\mathrm{ED}^2_{\mathrm{norm}}(T_{i,m}, T_{j,m}) = 2m\left(1 - \frac{\langle T_{i,m},\, T_{j,m}\rangle - m\mu_{T_{i,m}}\mu_{T_{j,m}}}{m\sigma_{T_{i,m}}\sigma_{T_{j,m}}}\right), \qquad (1)$$

где $T_{i,m}, T_{j,m} \in \mathbb{R}^m$, $\mu_{T_{i,m}}$ и $\mu_{T_{j,m}}$, $\sigma_{T_{i,m}}$ и $\sigma_{T_{j,m}}$ — среднее арифметическое и стандартное отклонение $T_{i,m}$ и $T_{j,m}$ соответственно.

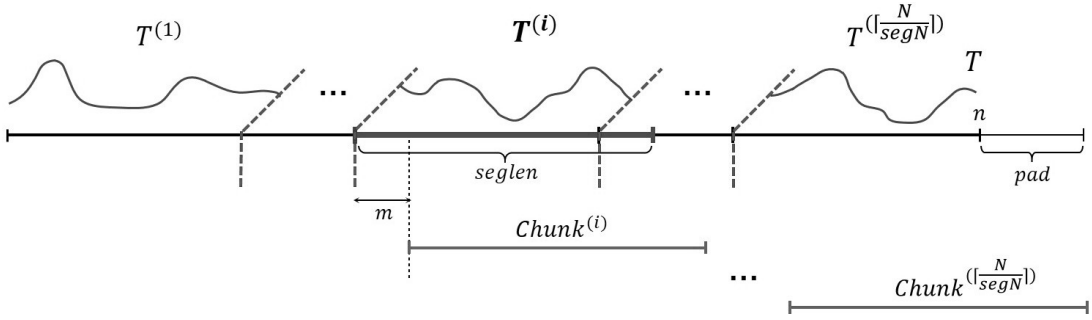


Рис. 1. Схема сегментирования ряда

Для параллельной реализации фаз отбора и очистки предложена следующая схема сегментирования ряда (см. рис. 1). Временной ряд разбивается одновременно на S непересекающихся сегментов $T^{(i)}$ и порций $Chunk^{(j)}$, содержащих $segN$ подпоследовательностей длины m ряда T и имеющих длину $seglen$, где $S = \lceil \frac{N}{segN} \rceil$, $seglen = segN + m - 1$, $1 \leqslant i, j \leqslant S$. Для баланса загрузки нитей в блоке параметр $segN$ устанавливается кратным размеру варпа графического процессора. Порции сдвинуты относительно сегмента на длину подпоследовательности m, что позволяет избежать избыточных проверок на пересечение кандидатов и подпоследовательностей порции.

Для предотвращения потери результатов на стыке сегментов и порций предложен *способ разбиения с перекрытием*. Пусть S — количество сегментов ряда T, $T^{(i)}$ — i-й сегмент ряда, где $1 \leqslant i \leqslant S$. Тогда сегмент $T^{(i)}$ определяется как подпоследовательность $T_{start,\,len}$, где

$$\begin{gathered} start = k \cdot \left\lfloor \frac{N}{S} \right\rfloor + 1, \\ len = \begin{cases} \lfloor \frac{N}{S} \rfloor + (N \bmod S) + m - 1, & k = S - 1 \\ \lfloor \frac{N}{S} \rfloor + m - 1, & \text{otherwise.} \end{cases} \end{gathered} \tag{2}$$

Если количество подпоследовательностей ряда длины m не кратно количеству подпоследовательностей в сегменте $segN$, то справа к временному ряду добавляются pad фиктивных элементов со значением $+\infty$:

$$pad = \begin{cases} m - 1, & N \bmod segN = 0 \\ S \cdot segN + 2(m-1) - n, & \text{otherwise} \end{cases}. \tag{3}$$

Фазы отбора и очистки реализуются в виде CUDA-ядер, каждое из которых формирует одномерную сетку из S блоков по $segN$ нитей в блоке (см. алг. 1). На *фазе отбора кандидатов* блок нитей полагает подпоследовательности сегмента локальными кандидатами и выполняет сканирование и обработку подпоследовательностей ряда, которые не пересекаются с кандидатами и расположены *справа* от данного сегмента. Используя формулу (1), блок вычисляет расстояния от всех кандидатов своего сегмента $T^{(i)}$ до всех под-

Алг. 1 PD3SELECT (IN T, m, r; OUT $\mathcal{C}$)

1: $Cand \leftarrow \overline{\texttt{TRUE}}$; $Neighbor \leftarrow \overline{\texttt{TRUE}}$ ▷ Инициализация битовых карт
2: **for all** $T^{(i)} \in T$ **do** ▷ Обработка сегментов
▷ Сканирование порций, стоящих справа от сегмента
3: **for all** $Chunk^{(j)} \in T^{(i)}$ **where** $i \leqslant j$ **do**
▷ Обработка кандидатов и первой подпоследовательности порции
4: **if** $i = j$ **then**
5: $QTrow \leftarrow$ CALCDOTPRODUCTS$(T^{(i)}_{1,\,m},\, Chunk^{(j)})$
6: **continue**
7: $QTrow \leftarrow$ UPDATEDOTPRODUCTS$(QTrow, T^{(i)}_{1,\,m},\, Chunk^{(j)})$
8: $QTcol \leftarrow$ CALCDOTPRODUCTS$(Chunk^{(j)}_{1,\,m}, T^{(i)})$
9: $dist \leftarrow$ CALCDIST$(Chunk^{(j)}_{1,\,m}, T^{(i)}, QTcol, \bar{\mu}, \bar{\sigma})$
10: **if** $dist < r$ **then**
11: $Cand(i \cdot segN + tid) \leftarrow \texttt{FALSE}$; $Neighbor(j \cdot segN + 1) \leftarrow \texttt{FALSE}$
12: **else**
13: $nnDist(j \cdot segN + 1) \leftarrow \min\big(dist, nnDist(j \cdot segN + 1)\big)$
14: **if not** $\bigvee_{k=i\cdot segN}^{(i+1)\cdot segN} Cand(k)$ **then**
15: **break**
▷ Обработка кандидатов и остальных подпоследовательностей порции
16: **for all** $Chunk^{(j)}_{k,\,m} \in S^{m}_{Chunk^{(j)}} \setminus Chunk^{(j)}_{1,\,m}$ **do**
17: $QTcol \leftarrow$ UPDATEDOTPRODUCTS$(QTcol, QTrow, Chunk^{(j)}_{k,\,m}, T^{(i)})$
18: $dist \leftarrow$ CALCDIST$(Chunk^{(j)}_{k,\,m}, T^{(i)}, QTcol, \bar{\mu}, \bar{\sigma})$
19: **if** $dist < r$ **then**
20: $Cand(i \cdot segN + tid) \leftarrow \texttt{FALSE}$; $Neighbor(j \cdot segN + k) \leftarrow \texttt{FALSE}$
21: **else**
22: $nnDist(j \cdot segN + 1) \leftarrow \min\big(dist, nnDist(j \cdot segN + 1)\big)$
23: **if not** $\bigvee_{k=i\cdot segN}^{(i+1)\cdot segN} Cand(k)$ **then**
24: **break**
▷ Формирование множества кандидатов
25: $\mathcal{C} \leftarrow \big\{\{T_{i,\,m} \in S^{m}_{T};\ nnDist(i)\} \mid 1 \leqslant i \leqslant n - m + 1,\, Cand(i) = \texttt{TRUE}\big\}$
26: **return** $\mathcal{C}$

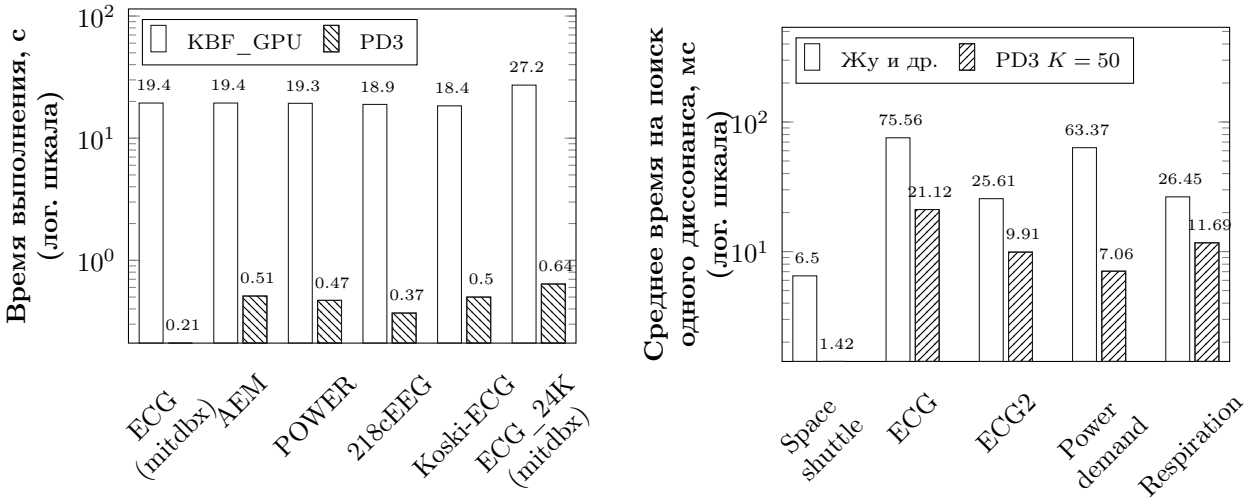
последовательностей текущей порции $Chunk^{(j)}$ следующим образом. Сначала нити блока вычисляют скалярные произведения между первым кандидатом сегмента и всеми подпоследовательностями текущей порции, далее — между первой подпоследовательностью текущей порции и всеми кандидатами сегмента, сохраняя результаты в векторах $QTrow$ и $QTcol \in \mathbb{R}^{segN}$, хранящихся в разделяемой памяти GPU. Далее на основе полученных результатов $QTrow$, $\overline{\mu}$, $\overline{\sigma}$ вычисляются расстояния между первой подпоследовательностью порции и всеми кандидатами сегмента. Если расстояние от кандидата до подпоследо-

вательности меньше порога r, то они исключаются из дальнейшей обработки как заведомо не являющиеся диссонансами. Если все локальные кандидаты отброшены, блок досрочно завершает работу.

После этого нити выполняют аналогичные действия над оставшимися подпоследовательностями текущей порции, вычисляя скалярные произведения более эффективно: на основе вычисленных векторов $QTcol$ и $QTrow$. Нить с номером tid в блоке, вычисляющая одно скалярное произведение между k-й ($1 < k \leqslant segN$) подпоследовательностью порции $Chunk^{(j)}$ и кандидатом сегмента $T^{(i)}$, использует следующую формулу:

$$QTrow(tid) = \begin{cases} QTrow(tid-1) - T^{(i)}_{tid-1,\,m}(1) \cdot Chunk^{(j)}_{k-1,\,m}(1) + \\ \qquad + T^{(i)}_{tid,\,m}(m) \cdot Chunk^{(j)}_{k,\,m}(m), & 1 < tid \leqslant segN\,. \\ QTcol(k), & tid = 1 \end{cases} \tag{4}$$

Фаза очистки диссонансов схожа с описанной выше фазой отбора кандидатов. В очистке участвуют сегменты ряда, множество локальных кандидатов которых не пусто. Очистка кандидатов сегмента заключается в сканировании и обработке подпоследовательностей ряда, которые не пересекаются с кандидатами и расположены *слева* от данного сегмента. Если расстояние от кандидата до подпоследовательности порции меньше порога r, то кандидат отбрасывается.



(a) Сравнение PD3 с KBF_GPU

(b) Сравнении PD3 с алгоритмом Жу и др.

Рис. 2. Производительность алгоритма PD3 в сравнении с аналогами

В вычислительных экспериментах выполнялось сравнение PD3 с известными аналогами: KBF_GPU и алгоритмом Жу и др. — на графических процессорах NVIDIA Tesla V100 и NVIDIA Tesla P100 суперкомпьютеров «Нейрокомпьютер» ЮУрГУ и «Ломоносов-2» МГУ соответственно. Взятые временные ряды идентичны тем, которые использованы авторами алгоритмов-

конкурентов. Эксперименты показали (см. рис. 2), что по общему времени выполнения PD3 опережает алгоритм KBF_GPU более чем на два порядка и по среднему времени на поиск одного диссонанса алгоритм Жу и др. более чем в два раза при поиске top-50 диссонансов.

В третьей главе, «Параллельный алгоритм поиска диссонансов произвольной длины для графических процессоров», представлен новый параллельный алгоритм PALMAD (Parallel Arbitrary Length MERLIN-based Anomaly Discovery), предназначенный для поиска диссонансов временного ряда, имеющих длину в заданном диапазоне. Описаны принципы применения разработанного ранее алгоритма PD3 для распараллеливания указанной задачи. Доказано утверждение о рекуррентных формулах, позволяющие сократить объем вычислений на фазе предварительной обработки данных. Предложены метод построения тепловой карты диссонансов для их визуализации и алгоритм ранжирования найденных диссонансов независимо от их длин. Представлены результаты вычислительных экспериментов, исследующие эффективность разработанного алгоритма PALMAD.

Общая схема вычислений алгоритма PALMAD предполагает разбиение диапазона длин диссонанса $minL..maxL$ на три неравные части и выполнение трех шагов поиска диссонансов соответствующих длин. На каждом шаге выполняются следующие действия: инициализация порога r, предварительная обработка данных, поиск диссонансов D_m с заданным порогом r с помощью алгоритма PD3 и подбор порога r, пока поиск не закончится успехом. По завершении шага найденные диссонансы D_m добавляются в результирующее множество $\mathcal{D} = \cup_{m=minL}^{maxL} D_m$.

Для избежания повторных вычислений в фазу параллельной предобработки данных добавлена процедура обновления средних арифметических μ и стандартных отклонений σ подпоследовательностей ряда, имеющих длину $minL < m \leqslant maxL$. Доказано утверждение, в котором выводятся рекуррентные формулы, связывающие вычисление μ и σ подпоследовательностей соседних длин m и $m+1$.

Утверждение. Пусть имеется временной ряд T длины n и две его подпоследовательности $T_{i,\,m}$ и $T_{i,\,m+1}$, имеющие длины m и $m+1$ соответственно, где $1 \leqslant i \leqslant n-m$ и $3 \leqslant m \leqslant n$. Тогда среднее арифметическое $\mu_{T_{i,\,m+1}}$ и стандартное отклонение $\sigma^2_{T_{i,\,m+1}}$ подпоследовательности $T_{i,\,m+1}$ вычисляются по следующим рекуррентным формулам:

$$\mu_{T_{i,\,m+1}} = \frac{1}{m+1}\big(m\mu_{T_{i,\,m}} + t_{i+m}\big), \tag{5}$$

$$\sigma^2_{T_{i,\,m+1}} = \frac{m}{m+1}\Big(\sigma^2_{T_{i,\,m}} + \frac{1}{m+1}\big(\mu_{T_{i,\,m}} - t_{i+m}\big)^2\Big). \tag{6}$$

Для визуализации найденных аномалий предложен метод построения *тепловой карты диссонансов* произвольной длины. Сначала строится матрица матричных профилей $MMP \in \mathbb{R}^{(maxL-minL+1)\times(n-minL)}$, являющаяся результатом работы алгоритма PALMAD. Строкой матрицы MMP является матричный профиль $MP_m \in \mathbb{R}^{n-m+1}$ — ряд, i-й элемент которого есть расстояние от i-й подпоследовательности длины m до ее ближайшего соседа, в котором обнулены элементы, где соответствующие подпоследовательности не являются диссонансами:

$$MMP(m,\, i) = \begin{cases} MP_{minL+m-1}(i), & T_{i,\, minL+m-1} \in D_{minL+m-1} \\ 0, & \text{otherwise} \end{cases}, \text{где} \tag{7}$$

$$\begin{gathered} MP_m(i) = \mathrm{Dist}(T_{i,\,m},\, Neighbor), \\ Neighbor = \arg \min_{T_{j,\,m} \in D_{T_{i,\,m}}} \mathrm{Dist}(T_{i,\,m},\, T_{j,\,m}). \end{gathered} \tag{8}$$

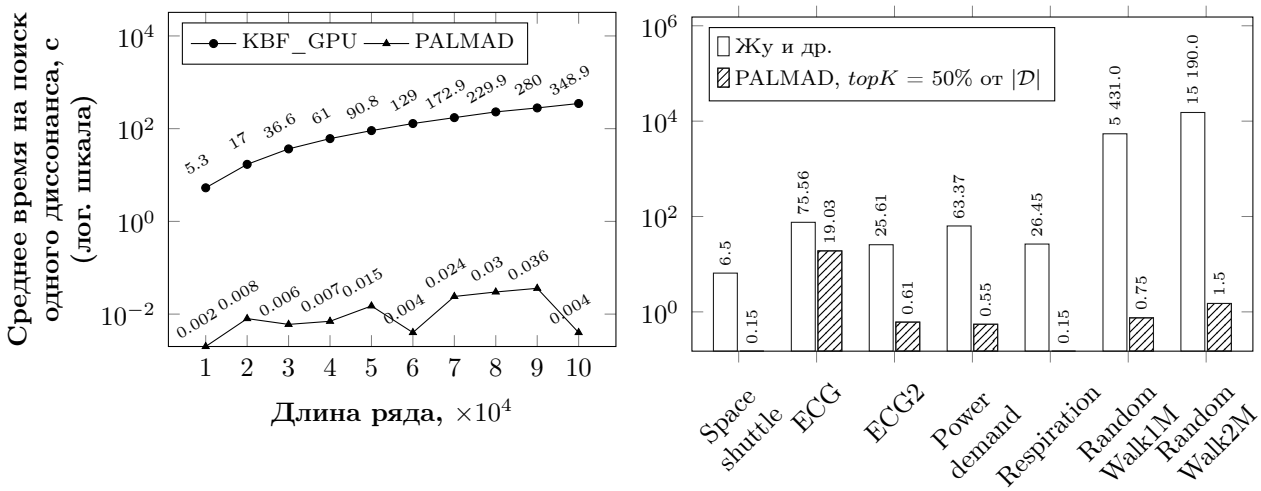
Далее каждая строка матрицы $MMP(m, \cdot)$ нормализуется с помощью множителя $\frac{1}{2m}$. Результатом является матрица яркости пикселей тепловой карты, где яркость пропорциональна степени аномальности диссонанса, $heatmap \in \mathbb{R}^{(maxL-minL+1)\times(n-minL)}$:

$$heatmap(m,\, i) = \frac{MMP(m,\, i)}{2m}. \tag{9}$$

Предложен алгоритм ранжирования найденных диссонансов независимо от их длин на основе построенной тепловой карты. Алгоритм упорядочивает диссонансы по убыванию степени аномальности, исключая при этом пересечения диссонансов: $\arg \max_{1 \leqslant i \leqslant n-m+1} \max_{minL \leqslant m \leqslant maxL} heatmap(m, i)$.

Исследовано применение алгоритма PALMAD на сенсорных данных, взятых из различных предметных областей цифровой индустрии: машиностроение, металлургия и энергетика. Для каждого тематического исследования построена тепловая карта диссонансов и найдены 10 наиболее значимых диссонансов. Найденные аномалии свидетельствуют о нештатной ситуации, отказах, сбоях и износе технологического оборудования.

Для исследования эффективности разработанного алгоритма проведены вычислительные эксперименты, где сравнивалась производительность PALMAD с известными аналогами: KBF_GPU и алгоритм Жу и др. — на графических процессорах NVIDIA Tesla V100 и NVIDIA Tesla P100 суперкомпьютеров «Нейрокомпьютер» ЮУрГУ и «Ломоносов-2» МГУ. Эксперименты показали (см. рис. 3), что алгоритм PALMAD опережает KBF_GPU и алгоритм Жу и др. по среднему времени на поиск одного диссонанса более чем на два порядка и в три раза соответственно.



(a) Сравнение PALMAD с KBF_GPU

(b) Сравнение PALMAD с алгоритмом Жу и др.

Рис. 3. Производительность алгоритма PALMAD в сравнении с аналогами

В четвертой главе, «Параллельный алгоритм поиска диссонансов произвольной длины для кластерных вычислительных систем с графическими процессорами», представлен новый параллельный алгоритм PADDi (PALMAD-based Anomaly Discovery on Distributed GPUs) для поиска диссонансов, имеющих длину в заданном диапазоне, во временном ряде, который не может быть целиком размещен в оперативной памяти. Описаны принципы применения разработанных ранее алгоритмов PD3 и PALMAD для распараллеливания указанной задачи. Рассмотрена реализация фазы очистки кандидатов в диссонансы, найденных с помощью алгоритмов PD3 и PALMAD. Представлены результаты вычислительных экспериментов, исследующих эффективность разработанного алгоритма PADDi.

Алгоритм PADDi обладает двухуровневым параллелизмом по данным. На первом уровне временной ряд T разбивается на фрагменты $\{T^{(i)}\}_{i=1}^{p}$ равной длины, распределяемые по дискам p узлов вычислительного кластера. Второй уровень параллелизма реализуется посредством разбиения каждого фрагмента $T^{(i)}$ на сегменты $\{T_j^{(i)}\}_{j=1}^{q}$ равной длины, обрабатываемые q графическими процессорами p узлов кластера, $1 \leqslant i \leqslant p$. Фрагментация и сегментация выполняется с помощью предложенного способа разбиения с перекрытием. Для обменов данными между узлами используется технология MPI, для организации вычислений на графических процессорах — технология CUDA.

Для параллельной обработки всех сегментов фрагмента порождается количество CUDA потоков, равное числу GPU, установленных на узле. В каждом CUDA потоке производится асинхронное копирование данных из закрепленной памяти CPU в память GPU, запускаются CUDA-ядра, реализующие фазы алгоритма PADDi, и выполняются асинхронные передачи результатов вычислений CUDA-ядер из GPU на CPU. В качестве строительных блоков

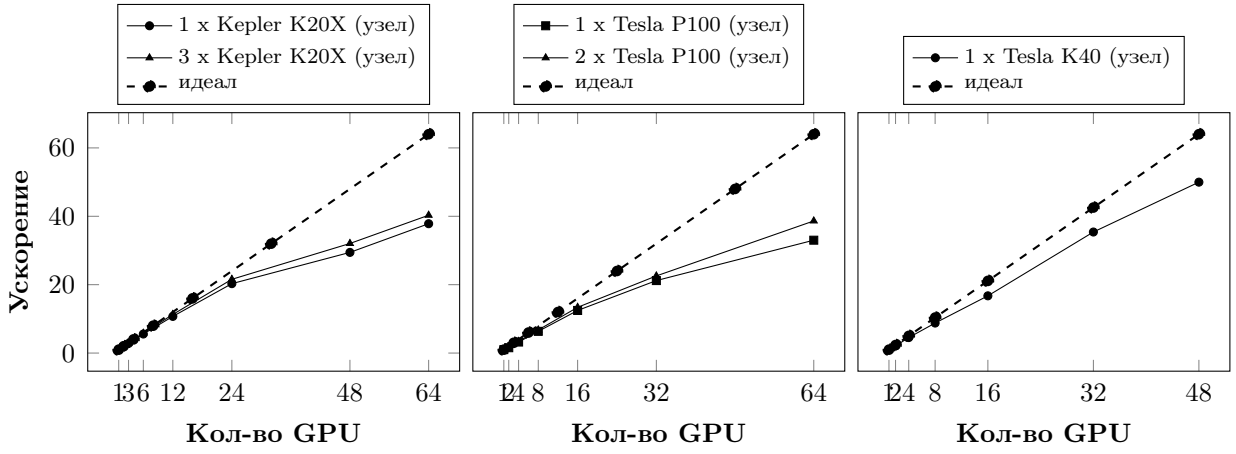
алгоритм PADDi включает в себя параллельные алгоритмы PD3 и PALMAD, описанные в главах 2 и 3 соответственно.

Алг. 2 PADDiDiscordSearch (in: $T^{(i)}$, m, q, r, $topK$; out: $\mathcal{D}_m$)

1: $i \leftarrow$ MPI_Comm_rank()
2: $nnDist_m \leftarrow -\infty$
3: **while** $nnDist_m < 0$ **and** $|\mathcal{D}_m| < topK$ **do**
4: **for** $j \leftarrow 1$ **to** q **do** ▷ Отбор кандидатов фрагмента
5: $\mathcal{D}_j^{(i)} \leftarrow \text{PD3}(T_j^{(i)}, \mu_j^{(i)}, \sigma_j^{(i)}, r^2)$
6: $\mathcal{C}^{(i)} \leftarrow \cup_{j=1}^{q} \mathcal{D}_j^{(i)}$
7: **for** $j \leftarrow 1$ **to** q **do** ▷ Очистка диссонансов фрагмента
8: $\mathcal{D}_j^{(i)} \leftarrow \text{LocalRefine}(T_j^{(i)}, \mathcal{C}^{(i)})$
9: $\mathcal{C}_m^{(i)} \leftarrow \cap_{j=1}^{q} \mathcal{D}_j^{(i)}$
10: $\mathcal{C}_m \leftarrow$ MPI_Allgatherv($\mathcal{C}_m^{(i)}$, MPI_FLOAT)
11: **for** $j \leftarrow 1$ **to** q **do** ▷ Очистка диссонансов ряда
12: $\mathcal{D}_j^{(i)} \leftarrow \text{GlobalRefine}(T_j^{(i)}, \mathcal{C}_m)$
13: $\widetilde{\mathcal{C}}_m^{(i)} \leftarrow \cap_{j=1}^{q} \mathcal{D}_j^{(i)}$
14: $\mathcal{D}_m \leftarrow$ MPI_Gatherv($\widetilde{\mathcal{C}}_m^{(i)}$)
15: **if** $i =$ MASTER_NODE **then**
16: $nnDist_m \leftarrow \min_{d \in \mathcal{D}_m} d.nnDist$
17: MPI_Send($nnDist_m$)
18: **else**
19: MPI_Recv($nnDist_m$)
20: **return** $\mathcal{D}_m$

Поиск диссонансов (см. алг. 2) осуществляется в три фазы, выполняемых на каждом узле кластера: отбор и очистка диссонансов сегмента, очистка диссонансов фрагмента и ряда в целом. *Первая фаза* задействует алгоритм PD3, который применяется к каждому сегменту ряда $T_j^{(i)}$, обрабатываемому на отдельном j-м графическом процессоре i-го узла (см. строки 4, 5). Далее на каждом узле кластера формируется множество кандидатов в диссонансы $\mathcal{C}^{(i)}$ как объединение множеств диссонансов всех сегментов $\mathcal{C}^{(i)} = \cup_{j=1}^{q} \mathcal{D}_j^{(i)}$ (см. строку 6). Затем *на второй фазе* каждый GPU узла выполняет отбрасывание ложноположительных диссонансов фрагмента из множества кандидатов $\mathcal{C}^{(i)}$ (строки 7, 8), формируя множество $\mathcal{D}_j^{(i)}$. Далее на стороне CPU создается множество кандидатов фрагмента $\mathcal{C}_m^{(i)} = \cap_{j=1}^{q} \mathcal{D}_j^{(i)}$ (строка 9). Все обмены данными между графическими процессорами выполняются через память центрального процессора узла кластера. Затем полученные множества кандидатов всех фрагментов $\{\mathcal{C}_m^{(i)}\}_{i=1}^{p}$ объединяются и рассылаются каждому узлу с помощью функции MPI_Allgatherv (строка 10). После пересылок

на третьей фазе полученное множество кандидатов $\mathcal{C}_m$ очищается от ложноположительных экземпляров, которые не являются диссонансами в ряде в целом (строки 11, 12). Очистка диссонансов фрагмента и ряда на второй и третьей фазах алгоритма PADDi реализуется с помощью параллельного блочного алгоритма умножения матриц кандидатов и подпоследовательностей сегмента. Результирующее множество диссонансов заданной длины $\mathcal{D}_m$ формируется на узле-мастере как пересечение множеств кандидатов в диссонансы ряда $\mathcal{D}_m = \cap_{i=1}^{p} \widetilde{\mathcal{C}}_m^{(i)}$, полученных узлами на третьей фазе алгоритма и переданных с помощью функции MPI_Gatherv (строки 13–19).



(a) NVIDIA Kepler K20X, Лобачевский

(b) NVIDIA Tesla P100, Ломоносов-2

(c) NVIDIA Tesla K40, Ломоносов-2

Рис. 4. Ускорение алгоритма PADDi (ряд ECG, $n = 2 \cdot 10^6$, $m \in [64, 128]$)

Проведены вычислительные эксперименты с реальными и синтетическими временными рядами на платформе суперкомпьютеров «Ломоносов-2» МГУ и «Лобачевский» ННГУ для трех различных конфигураций, задействующих от 48 до 64 графических процессоров. Результаты экспериментов показали (см. рис. 4), что ускорение сохраняет линейный характер и его стагнация или деградация отсутствуют.

В пятой главе, «Метод поиска аномалий в потоковых данных», представлен новый метод поиска аномальных подпоследовательностей в потоковом временном ряде, получивший название DiSSiD (Discord, Snippet, and Siamese Neural Network-based Detector of anomalies). Описаны компоненты метода DiSSiD. Для обучения нейросетевой модели предложена модифицированная функция контрастных потерь. Описана модификация алгоритма поиска сниппетов, позволяющая более точно находить указанные подпоследовательности временного ряда. Предложена эвристика по подбору гиперпараметров для модифицированного алгоритма поиска сниппетов. Представлены результаты вычислительных экспериментов над временными рядами из различных предметных областей.

Метод DiSSiD базируется на концепциях диссонанса и сниппета, которые формализуют соответственно понятия аномальных и типичных подпоследовательностей временного ряда. Предложенный метод включает в себя следующие компоненты: нейросетевую модель, определяющую степень аномальности входной подпоследовательности потокового ряда, и алгоритм автоматизированного построения обучающей выборки для этой модели.

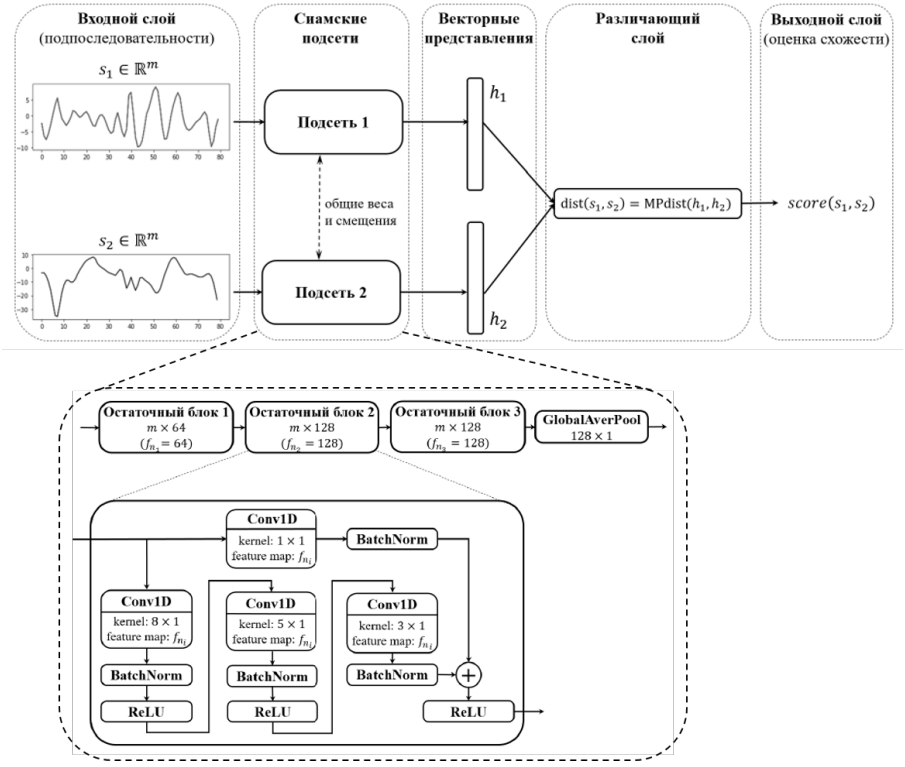


Рис. 5. Нейросетевая модель DiSSiD

Нейросетевая модель представляет собой сиамскую нейронную сеть, которая объединяет в себе две идентичные подсети, представляющие собой модификацию нейросетевой модели ResNet (см. рис. 5). На вход сиамской сети подается пара подпоследовательностей $< s_1, s_2 >$, $s_1, s_2 \in \mathbb{R}^m$, каждая из которых обрабатывается отдельной подсетью. Подсети формируют векторные представления h_1 и h_2 входных подпоследовательностей s_1 и s_2 соответственно. На выходе модель выдает MPdist-расстояние между сформированными векторными представлениями, $score(s_1, s_2) = \mathrm{MPdist}(h_1, h_2)$, $score(s_1, s_2) \in [0, 1]$.

Подсеть включает в себя три одинаковых остаточных блока и за ними слой глобальной усредняющей агрегации, формирующий векторное представление. Каждый остаточный блок включает в себя три сверточных слоя, на которых применяются фильтры (ядра) с размерами 8×1, 5×1 и 3×1 соответственно. Каждый сверточный слой чередуется со слоем пакетной нормализации, к которому применяется функция активации ReLU. После прохождения всех слоев остаточного блока формируются карты признаков (первый

блок — 64 карты, остальные два блока — по 128 карт), которые далее складываются с входом этого остаточного блока, пропущенного через сверточный слой с ядром размера 1×1. Размерность итогового векторного представления определяется количеством карт признаков последнего слоя в последнем остаточном блоке.

Для обучения модели DiSSiD из заданного временного ряда формируется обучающая выборка, определяемая следующим образом:

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{\text{true}} \cup \mathcal{L}_{\text{false}} \setminus \mathcal{L}_{\text{anomaly}} \\ \mathcal{L}_{\text{true}} &= \{< s_1; s_2; 1 > \mid\ s_1, s_2 \in C_i.NN, \quad 1 \leqslant i \leqslant K\} \\ \mathcal{L}_{\text{false}} &= \{< s_1; s_2; 0 > \mid\ s_1 \in C_i.NN,\ s_2 \in C_j.NN, \quad i \neq j,\ 1 \leqslant i, j \leqslant K\}, \end{aligned} \tag{10}$$

где множество $\mathcal{L}_{\text{true}}$ — пары подпоследовательностей, входящих в множество ближайших соседей одного и того же сниппета, множество $\mathcal{L}_{\text{false}}$ — пары подпоследовательностей из множеств ближайших соседей разных сниппетов, множество $\mathcal{L}_{\text{anomaly}}$ содержит аномальные подпоследовательности, не включаемые в обучающую выборку.

Для обучения модели DiSSiD предложена следующая модифицированная функция контрастных потерь:

$$\begin{aligned} L(s_1, s_2) &= \delta_{s_1 s_2} \cdot \text{MPdist}(h_1, h_2) + \\ &\quad +(1 - \delta_{s_1 s_2})\big(\max(\tau - \text{MPdist}(h_1, h_2), 0)\big)^2 \end{aligned} \tag{11}$$

$$\delta_{s_1 s_2} = \begin{cases} 1, & s_1, s_2 \in C_i.NN, 1 \leqslant i \leqslant K \\ 0, & s_1 \in C_i.NN, s_2 \in C_j.NN, 1 \leqslant i \neq j \leqslant K \end{cases} \tag{12}$$

$$\tau = \min_{\{s_1, s_2 \mid \delta_{s_1 s_2} = 0\}} \text{MPdist}_\ell(h_1, h_2), \tag{13}$$

где s_1 и s_2, h_1 и h_2 — исходные подпоследовательности и их векторные представления соответственно; кронекериан $\delta_{s_1 s_2}$ принимает значение 1, если исходные подпоследовательности являются ближайшими соседями одного и того же сниппета, и 0 в противном случае; τ — минимальное расстояние MPdist между векторными представлениями исходных подпоследовательностей, являющихся ближайшими соседями разных сниппетов (параметр модели).

Применение модели DiSSiD происходит следующим образом. Сначала формируется набор пар $\{< s; C_i >\}_{i=1}^{K}$ «входная подпоследовательность, сниппет». Затем элементы данного набора параллельно падаются на вход экземплярам модели, запущенными на отдельном графическом процессоре, каждая из которых выдает соответствующую оценку схожести векторных представлений элементов входной пары $score(s, C_i)$. Далее оценка аномальности входной подпоследовательности получается как минимальное значение по указанному набору $anomaly(s) = \min\limits_{1 \leqslant i \leqslant K} score(s, C_i)$. Входная подпоследова-

тельность s считается аномальной, если ее оценка превышает значение порога, определяемого как 95-й процентиль по набору оценок схожести для кортежей валидационной выборки.

Алг. 3 CleanData (in: T, m, α, φ, K; out: $\mathcal{L}$)

1: $C_T^m \leftarrow \mathrm{SFA}(T, m, K)$ ▷ Поиск типичной активности
2: $C_{\mathrm{weak}} \leftarrow \{C_i \in C_T^m \mid C_i.frac \leqslant \varphi, 1 \leqslant i \leqslant K\}$ ▷ Поиск нетипичной активности
3: $C_T^m \leftarrow C_T^m \setminus C_{\mathrm{weak}}$
4: **for** $i \leftarrow 1$ **to** $|C_T^m|$ **do** ▷ Поиск шумов
5: $\quad O \leftarrow \mathrm{IsolationForest}(C_i.profile) \cup O$
6: $n_{\mathrm{discord}} \leftarrow \lceil \alpha \cdot (n - m + 1) \rceil$
7: $D \leftarrow \mathrm{PALMAD}(T, m, m, n_{\mathrm{discord}})$ ▷ Поиск аномальной активности
8: $\mathcal{L}_{\mathrm{anomaly}} \leftarrow C_{\mathrm{weak}} \cup C_{\mathrm{weak}}.NN \cup D \cup O$
9: $\mathcal{L} \leftarrow S_T^m \setminus \mathcal{L}_{\mathrm{anomaly}}$ ▷ Очистка
10: **return** $\mathcal{L}$

Входными данными для *алгоритма автоматизированного формирования обучающей выборки* является репрезентативный временной ряд, адекватно отражающего типичную деятельность субъекта, противоположную аномалиям. Формирование обучающей выборки включает в себя два шага: очистка и генерация.

Очистка подразумевает формирование множества подпоследовательностей ряда заданной аналитиком длины и удаление из указанного множества аномальных подпоследовательностей. Псевдокод шага очистки представлен в алг. 3. Данный алгоритм имеет следующие задаваемые аналитиком параметры: репрезентативный ряд T ($|T| = n$), длина подпоследовательности m ($m \ll n$), предполагаемая доля аномальных подпоследовательностей в ряде α ($0 < \alpha \ll 1$, типичное значение $\alpha = 0.05$), предполагаемое количество сниппетов K ($K > 1$), пороговая мощность сниппета φ ($0 < \varphi < 1/K$).

Поиск сниппетов выполняется с помощью модифицированного алгоритма поиска сниппетов SFA (см. строку 1). Модификации заключаются в улучшении процедуры отбора сниппетов и автоматическом подборе гиперпараметра, используемого при вычислении расстояния MPdist:

$$k = \max\left(1 + \frac{1 - 2\ell}{2(m - \ell + 1)},\ k_{\mathrm{default}}\right). \tag{14}$$

Затем из найденного множества сниппетов исключаются маломощные сниппеты (см. строки 2, 3). Далее в множестве ближайших соседей каждого сниппета выполняется нахождение подпоследовательностей-шумов, реализуемое методом изолирующего леса (см. строки 4, 5). Наконец, с помощью разработанного параллельного алгоритма PALMAD выполняется поиск диссонансов,

реализующий нахождение подпоследовательностей аномальной активности (см. строки 6, 7). Мощность множества диссонансов вычисляется на основе параметра α. В завершении очистки из множества подпоследовательностей ряда исключаются найденные аномальные подпоследовательности (см. строки 8, 9), формируя тем самым множество, используемое для генерации обучающей выборки модели.

На шаге генерации из очищенного множества подпоследовательностей формируются два множества $\mathcal{L}_{\text{true}}$ и $\mathcal{L}_{\text{false}}$, объединение которых дает искомую выборку. Элементами первого из них являются пары подпоследовательностей одного и того же сниппета, второго — ближайших соседей разных сниппетов.

	Метод	Метрика							
		R-AUC-ROC	Средний R-AUC-ROC	R-AUC-PR	Средний R-AUC-PR	VUS-ROC	Средний VUS-ROC	VUS-PR	Средний VUS-PR
Без учителя	IForest		0.8941 (2)		0.6178 (2)		0.8771 (2)		0.5578 (2)
	LOF		0.791 (9)		0.4116 (9)		0.7534 (9)		0.3479 (11)
	MP		0.8274 (7)		0.4578 (8)		0.7927 (8)		0.3957 (8)
	DAMP		0.6757 (15)		0.1805 (17)		0.6682 (14)		0.1799 (17)
	NormA		0.6587 (16)		0.3177 (13)		0.6402 (16)		0.2969 (13)
	PCA		0.8355 (6)		0.5986 (3)		0.8203 (6)		0.5411 (3)
	POLY		0.7692 (10)		0.5221 (5)		0.7524 (10)		0.4847 (5)
		0 0.5 1		0 0.5 1		0 0.5 1		0 0.5 1	
С частичным привлечением учителя	AE		0.8432 (5)		0.4784 (7)		0.8222 (5)		0.4181 (7)
	Bagel		0.7542 (11)		0.2502 (14)		0.7361 (11)		0.2422 (14)
	DeepAnT		0.7206 (12)		0.2294 (15)		0.715 (12)		0.2288 (15)
	IE-CAE		0.8472 (4)		0.5501 (4)		0.8366 (4)		0.5186 (4)
	LSTM-AD		0.6876 (13)		0.188 (16)		0.6807 (13)		0.1894 (16)
	OceanWNN		0.8214 (8)		0.3837 (11)		0.8186 (7)		0.3886 (9)
	OCSVM		0.6508 (17)		0.3607 (12)		0.6258 (17)		0.3098 (12)
	TanoGAN		0.6791 (14)		0.3911 (10)		0.6625 (15)		0.3851 (10)
		0 0.5 1		0 0.5 1		0 0.5 1		0 0.5 1	
	DiSSiD (L1)		0.8477 (3)		0.5087 (6)		0.8403 (3)		0.4669 (6)
	DiSSiD (MPdist)		**0.9446 (1)**		**0.6992 (1)**		**0.9334 (1)**		**0.6242 (1)**
		0 0.5 1		0 0.5 1		0 0.5 1		0 0.5 1	

Рис. 6. Точность метода DiSSiD в сравнении с аналогами

Вычислительные эксперименты, проведенные на графическом процессоре NVIDIA Tesla V100 суперкомпьютера «Нейрокомпьютер» ЮУрГУ, на реальных временных рядах из различных предметных областей показывают (см. рис. 6), что предлагаемый метод DiSSiD по сравнению с известными аналогами показывает в среднем наиболее высокую точность обнаружения аномалий по метрикам R-AUC-ROC, R-AUC-PR, VUS-ROC, VUS-PR.

В заключении в краткой форме излагаются итоги выполненного диссертационного исследования; представляются отличия данной работы от ранее выполненных родственных работ других авторов; формулируются основные результаты диссертационной работы, выносимые на защиту; приводятся дан-

ные о публикациях и апробациях автора по теме диссертации; рассматриваются направления дальнейших исследований в данной области.

Заключение

В диссертации разработаны и исследованы новые методы и алгоритмы поиска аномальных подпоследовательностей временного ряда на платформе современных высокопроизводительных вычислительных систем. В настоящее время эффективный поиск аномалий во временных рядах является актуальной проблемой, решение которой востребовано в широком спектре научных и практических приложений. Выполненное исследование базируется на концепции диссонанса временного ряда.

В результате исследования разработаны параллельные алгоритмы поиска диссонансов временного ряда PD3, PALMAD, PADDi и метод обнаружения аномалий в потоковом временном ряде DiSSiD. Алгоритм PD3 обеспечивает поиск всех диссонансов временного ряда, имеющих заданную длину, на графическом процессоре. Алгоритм PALMAD обеспечивает поиск всех диссонансов временного ряда, имеющих длину в заданном диапазоне, на графическом процессоре. Алгоритм PADDi обеспечивает поиск всех диссонансов временного ряда, который не может быть целиком размещен в оперативной памяти, имеющих длину в заданном диапазоне, на высокопроизводительном кластере с графическими процессорами. Метод DiSSiD обеспечивает поиск аномалий потокового временного ряда и включает в себя нейросетевую модель и алгоритм построения обучающей выборки для указанной модели. Проведены вычислительные эксперименты с синтетическими и реальными временными рядами, подтверждающие эффективность разработанных методов и алгоритмов.

Результаты исследования могут быть применены для эффективного поиска аномалий во временных рядах в широком спектре предметных областей, поскольку разработанные параллельные алгоритмы поиска диссонансов временных рядов и метод обнаружения аномальных подпоследовательностей в потоковом временном ряде не зависят от предметной области и минимизируют участие аналитика-эксперта, от которого требуется задать длины искомых аномальных подпоследовательностей.

Основные результаты, полученные в ходе выполнения диссертационного исследования, являются новыми и не покрываются ранее опубликованными научными работами других авторов, обзор которых был дан в главе 1. Основные отличия заключаются в следующем.

1. Алгоритм PD3 поиска диссонансов фиксированной длины на графическом процессоре, в отличие от известных аналогов (алгоритма Жу и др. и алгоритма KBF_GPU) позволяет находить все диссонансы временного

ряда, имеющие заданную длину (а не один, наиболее важный из них). При этом PD3 опережает известные аналоги в разы по среднему времени поиска одного диссонанса. В соответствии с этим алгоритм PD3 будет более ценен, чем известные аналоги, в приложениях, где требуется поиск всех аномалий фиксированной длины.

2. Алгоритмы PALMAD и PADDi поиска диссонансов произвольной длины соответственно на графическом процессоре и высокопроизводительном вычислительном кластере с графическими процессорами не имеют аналогов среди параллельных алгоритмов. Алгоритм Жу и др. и алгоритм KBF_GPU являются относительно похожими, поскольку предназначены для поиска одного наиболее важного диссонанса. PALMAD и PADDi в сравнении с алгоритмами Жу и др. и KBF_GPU быстрее в разы по среднему времени поиска одного диссонанса. Рассмотренные в главе 1 алгоритмы DDD и PDD поиска диссонансов фиксированной длины на высокопроизводительных кластерах существенно (более чем на порядок) уступают алгоритму PADDi по быстродействию, поскольку предполагают интенсивные обмены данными между узлами кластера. В соответствии с этим алгоритмы PALMAD и PADDi будут более ценны, чем известные аналоги, в приложениях, где требуется поиск всех аномалий произвольной длины.
3. Метод DiSSiD обнаружения аномалий потокового временного ряда применяет концепции диссонанса и сниппета, которые формализуют понятия аномальных и типичных подпоследовательностей соответственно. Метод включает в себя нейросетевую модель распознавания аномальной подпоследовательности, представляющую собой сиамскую нейронную сеть, для обучения которой предложена модифицированная функция контрастных потерь, и в качестве подсети которой предложена модификация нейросетевой модели ResNet. В соответствии с этим, в отличие от известных методов с частичным привлечения учителя, DiSSID позволяет выявлять аномальные подпоследовательности ряда, отражающие нетипичную и редко встречающуюся активности исследуемого субъекта.

Теоретические исследования и практические разработки, выполненные в рамках этой диссертационной работы, предполагается продолжить по следующим направлениям: обобщение разработанных моделей, методов и подходов на многомерные временные ряды; разработка версий предложенных параллельных алгоритмов для многоядерных центральных процессоров и программируемых логических интегральных схем.

Публикации автора по теме диссертации

1. Kraeva, Ya. A Parallel Discord Discovery Algorithm for a Graphics Processor / Ya. Kraeva, M. Zymbler // Pattern Recognition and Image Analysis. – 2023. – Vol. 33, no. 2. – P. 101–112. DOI: 10.1134/S1054661823020062. (Перечень ВАК К1 – Scopus Q2)
2. Краева, Я.А. Поиск аномалий в сенсорных данных цифровой индустрии с помощью параллельных вычислений / Я.А. Краева // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. – 2023. – Т. 12, № 2. – С. 47–61. DOI: 10.14529/cmse230202. (Перечень ВАК К1 – RSCI)
3. Краева, Я.А. Обнаружение аномалий временного ряда на основе технологий интеллектуального анализа данных и нейронных сетей / Я.А. Краева // Вестник ЮУрГУ. Серия: Вычислительная математика и информатика. – 2023. – Т. 12, № 3. – С. 50–71. DOI: 10.14529/cmse230304. (Перечень ВАК К1 – RSCI)
4. Краева, Я.А. Поиск аномалий в больших временных рядах на кластере с GPU узлами / Я.А. Краева, М.Л. Цымблер // Вычислительные методы и программирование. – 2023. – Т. 24, № 3. – С. 291–304. DOI: 10.26089/NumMet.v24r320. (Перечень ВАК К1)
5. Zymbler, M. High-Performance Time Series Anomaly Discovery on Graphics Processors / M. Zymbler, Ya. Kraeva // Mathematics. – 2023. – Vol. 11, no. 14. – Article 3193. DOI: 10.3390/math11143193. (Перечень ВАК К1 – Web of Science Q1)

Свидетельства о регистрации программы

6. Краева, Я.А. DiSSiD: детектор аномалий временного ряда в режиме реального времени / Я.А. Краева // Свидетельство Роспатента о государственной регистрации программы для ЭВМ № 2023685034 от 22.11.2023.
7. Краева, Я.А. PALMAD: детектор аномалий различной длины во временном ряде на графическом процессоре / Я.А. Краева, М.Л. Цымблер // Свидетельство Роспатента о государственной регистрации программы для ЭВМ № 2022667716 от 23.09.2022.

Исходные тексты программ, реализующих разработанные в рамках диссертационного исследования алгоритмы и методы, а также наборы данных, с которыми проведены вычислительные эксперименты, подтверждающие эффективность указанных разработок, размещены в сети Интернет в свободно доступных репозиториях:
https://github.com/kraevaya/PD3, https://github.com/kraevaya/PALMAD,
https://github.com/kraevaya/PADDi, https://github.com/kraevaya/DiSSiD.

Диссертационное исследование выполнено при финансовой поддержке Российского научного фонда (грант № 23-21-00465).

Формат 60 × 84 1/16. Бумага офсетная. Печать офсетная. Авт. л. 1,0. Уч.-изд. л. 1,2.
Подписано в печать 14.03.2024. Тираж 100 экз.
Типография «Активист». 454080, г. Челябинск, пр. Ленина, 74б