На правах рукописи

Мунерман Виктор Иосифович

АЛГЕБРАИЧЕСКИЕ МОДЕЛИ И МЕТОДЫ ДЛЯ РАЗРАБОТКИ ПРОГРАММНО-АППАРАТНЫХ КОМПЛЕКСОВ МАССОВОЙ ОБРАБОТКИ ДАННЫХ

2.3.5 – Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей

АВТОРЕФЕРАТ

диссертации на соискание ученой степени

доктора технических наук

Москва – 2025

Работа выполнена в Федеральном государственном бюджетном образовательном учреждении высшего образования «Смоленский государственный университет»

Официальные оппоненты:

Сухомлин Владимир Александрович
доктор технических наук, профессор,
МГУ им. М.В. Ломоносова, профессор

Карпенко Анатолий Павлович
доктор физико-математических наук, профессор,
МГТУ им. Н.Э. Баумана, профессор

Кожухов Игорь Борисович
доктор физико-математических наук, профессор,
НИУ МИЭТ, профессор

Ведущая организация:

Акционерное общество Научно-производственный центр «Электронные вычислительно-информационные системы» (АО НПЦ «ЭЛВИС»)

Защита диссертации состоится 21 мая 2025 года в 15 часов 00 минут на заседании диссертационного совета 24.1.224.04 при ФИЦ ИУ РАН по адресу: 119333, г. Москва, ул. Вавилова, 44, корп. 2.

С диссертацией можно ознакомиться в библиотеке ФИЦ ИУ РАН по адресу: 119333, г. Москва, ул. Вавилова, 44, корп. 2 и на сайте официальном сайте ФИЦ ИУ РАН http://www.frccsc.ru.

Отзывы на автореферат в двух экземплярах, заверенные печатью учреждения, высылать по адресу: 119333, г. Москва, ул. Вавилова, 44, корп. 2, ученому секретарю диссертационного совета 24.1.224.04.

Автореферат разослан 2025 года.

Ученый секретарь
диссертационного совета 24.1.224.04 Разумчик Р.В.

Общая характеристика работы

Актуальность темы исследования. В настоящее время массовую обработку данных связывают с направлением, получившим название Big data. Big data (большие данные) – общий термин, который обозначает вновь создающиеся структурированные, неструктурированные и полуструктурированные данные сверхбольших и постоянно возрастающих объемов. Загрузка неструктурированных и полуструктурированных данных в обычную (например, реляционную) базу данных и последующая обработка требуют слишком больших затрат ресурсов вычислительных комплексов[1]. Поэтому работа посвящена рассмотрению распространенного класса массовой обработки – обработке высокоактивных структурированных данных. Понятие высокоактивных данных было введено в работах [1-3] для удобства обозначения данных, при выполнении операций, над которыми в обработку включается их большая часть, близкая к ста процентам. Для этого там же определен коэффициент активности данных, который определяется также в книге Ульмана Дж.[2]. Большие данные не только неотъемлемая часть современного мира обработки данных, но и основная его часть. Комплекс сложных научно-технических проблем, связанных с решением задач на основе обработки больших данных, при переходе к информационному обществу не только сохраняет, но и усиливает свою актуальность[3]. Об этом свидетельствуют интенсивные научные исследования в области баз данных, проводимые в России и за рубежом. Требования к обработке больших данных существенно влияют на технологию разработки программных и аппаратных вычислительных средств ее реализующих. Эти вопросы, связанные с решением сложных вычислительных задач, рассматривались в работах Опарина Г. А., Новопашина А. П., Ледянкина И. А., Легкова К. Е., de Carvalho Silva J., de Oliveira Dantas A. B. и др. Решение информационно-

[1] Соколов И.А., Будзко В.И., Калиниченко Л.А., Синицын И.Н., Ступников С.А. Развитие работ в области «Больших данных» в Российской академии наук // Системы компьютерной математики и их приложения, 2015. № 16. С. 103-112.

[2] Ульман Д. Базы данных на Паскале. – М.: Машиностроение, 1990. 368 с.

[3] Naeem M. et al. Trends and future perspective challenges in big data //Advances in Intelligent Data Analysis and Applications, 2022. P. 309-325.

логических задач, к которым относятся обработка транзакционных запросов и запросов, требующих больших рабочих нагрузок рассматривалось в работах Ordonez C., Bellatreche L., Liu C. и др. Задачи в области искусственного интеллекта рассматривали Рабинович З. Л., Sharifani K., Amini M., в том числе для переобучения сверточных нейронных сетей Soori M., Arezoo B., Tandon A. и др. Эти исследования обусловили необходимость использования параллельной обработки и обработки в основной (оперативной) памяти. Важное значение в обработке больших данных имеют параллельные и распределенные базы данных, для которых, как отмечают Pietron M., Wielgosz M., создаются специализированные программно-аппаратные комплексы – машины баз данных[4]. Большинство современных машин баз данных ориентированы на реализацию реляционных СУБД, которые недавно стали широко применяться для подготовки исходных данных в системах глубокого обучения, что подробно рассмотрено в ряде современных работ[5]. При разработке параллельных вычислительных систем особую роль играет повышение эффективности реализации отдельных операций, имеющих большую вычислительную сложность, о чем свидетельствуют многие публикации. К этим операциям относятся, например, многомерное дискретное преобразование Фурье, свертки в процессе обучения нейронных сетей, операция Join в реляционных системах баз данных. Современные аппаратные средства, такие как многопоточные процессоры архитектур подобных x86 и ARMv8, а также графические и тензорные процессоры (GPU, TPU), позволяют проектировать параллельные вычислительные системы, которые обеспечивают решение многих из перечисленных задач. Важное направление основано на использовании современных программируемых

[4] Калиниченко Л. А., Рывкин В. М. Машины баз данных и знаний. – М.:Наука, 1990. 296 с.; DeWitt D., Gray J. Parallel database systems: The future of high performance database systems // Communications of the ACM, 1992. Vol. 35. No. 6. P. 85-98; Соколинский Л. Б. Параллельные машины баз данных // Природа, 2001. №. 8. С. 10-17.

[5] Fey M. et al. Relational deep learning: Graph representation learning on relational databases // Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024; Robinson J. et al. RelBench: A Benchmark for Deep Learning on Relational Databases // 38th Conference on Neural Information Processing Systems (NeurIPS 2024) Track on Datasets and Benchmarks; Zhou L. et al. Serving Deep Learning Models from Relational Databases //Advances in Database Technology-EDBT, 2024. Vol. 27. No. 3. P. 717-724. Zahradník L., Neumann J., Šír G. A deep learning blueprint for relational databases // NeurIPS 2023 Second Table Representation Learning Workshop, 2023.

логических интегральных схем (ПЛИС/FPGA). Относительная простота программирования позволяет использовать их для решения как вычислительных, так и информационно-логических задач, и создания центров обработки данных. Особую роль современные ПЛИС играют в решении проблемы построения быстро реконфигурируемых вычислительных систем. Так в машине баз данных IBM Netezza[6], ПЛИС позволяет быстро перестраивать процессор обмена и подготовки данных S-Blade, превращая его в специализированный процессор для выполнения конкретного запроса, полученного от случайного пользователя. Рассмотренные важные задачи требуют для своего решения построения программно-аппаратных комплексов, основное свойство которых заключается в достижении эффективного баланса программного и аппаратного обеспечения. Эта эффективность определяется зависимостью, которая влияет на скорость обработки данных причем тем существенней, чем сложнее техника – это степень согласованности структуры алгоритмов с ее архитектурой[7]. Формальная постановка этой проблемы может быть сформулирована следующим образом[8]. Существуют два алгоритмических языка и некоторый алгоритм, написанный на одном из них; требуется найти оптимальную по заданным критериям реализацию этого алгоритма на другом языке. В программировании обычно первым является некоторый язык программирования, ориентированный на тот или иной круг задач, а вторым – внутренний язык машины, на которой решаются данные задачи. Поиск причин возникновения трудностей при решении задач на вычислительной технике параллельной архитектуры неизбежно приводит к выводу, что как истоки этих причин, так и пути их преодоления надо искать в математических знаниях об алгоритмах. Эти две посылки позволяют сделать следующее заключение. Алгебраическая система,

[6] How does Netezza Database work with Examples? // EDUCBA, 2022. https://www.educba.com/netezza-database/.
[7] Воеводин В.В. Вычислительная математика и структура алгоритмов. – М.: Национальный Открытый Университет ИНТУИТ, 2024. 145 с.
[8] Глушков В.М., Цейтлин Г.Е., Ющенко Е.Л. Алгебра. Языки. Программирование. – Киев: Наукова думка, 1989. 376 с.

в которой формализована решаемая задача, и модель вычислений (алгебраическая система, реализованная в системе команд вычислительной системы или комплекса) должны в наибольшей степени соответствовать друг другу. Эти алгебраические системы должны быть, по крайней мере, гомоморфными, а в идеальном случае, когда достигается полное соответствие – изоморфными. Для обработки больших данных традиционно используется массовая обработка данных (massively data processing, massively data computing, далее – МОД) – способ параллельной обработки больших объемов данных большим числом процессоров. Для ее реализации проектируются и используются специализированные программно-аппаратные комплексы. Область, в которой применение массовой обработки данных имеет важное значение – это обработка высокоактивных данных. Активность данных определяется отношением числа обращений к элементу данных (записи файла, элементу матрицы, строки таблицы) к общему числу обращений к информации (файлу, матрице, базе данных) в единицу времени. Важным примером применения предложенных в диссертационной работе методов построения программно-аппаратных комплексов для массовой обработки высокоактивных данных служит реляционное глубокое обучение. В реляционной базе данных выполняются запросы на формирование обучающей и тестирующей выборок. Поскольку системы глубокого обучения базируются на алгебре тензоров[9] – частном случае алгебры многомерных матриц[10], то целесообразно для подготовки данных для систем глубокого обучения использовать многомерно-матричные машины баз данных[11].

[9] Cohen N., Sharir O., Shashua A. On the expressive power of deep learning: A tensor analysis // PMLR, 2016. P. 698-728; Рудаков К. В., Чехович Ю. В. Алгебраический подход к проблеме синтеза обучаемых алгоритмов выделения трендов // ДАН, 2003. Т. 388. №. 1. С. 33-36; Дюкова Е. В., Журавлев Ю. И., Рудаков К. В. Об алгебрологическом синтезе корректных процедур распознавания на базе элементарных алгоритмов // Журнал вычислительной математики и математической физики, 1996. Т. 36. №. 8. С. 215-223; Тыртышников Е. Е. Матричный анализ и основы алгебры. – М.: МЦНМО, 2025. 493 с.; Замарашкин Н. Л., Оселедец И. В., Тыртышников Е. Е. Новые приложения матричных методов // Журнал вычислительной математики и математической физики, 2021. Т. 61. №. 5. С. 691-695.

[10] Соколов Н.П. Введение в теорию многомерных матриц. – Киев: Наукова думка, 1972. 176 с.

[11] Гончаров Е.И., Мунерман В.И., Синицын И.Н. Современные технологические средства создания многомерно-матричных машин баз данных // Системы высокой доступности, 2024. Т. 17. №1. С. 5-17.

В соответствии со сказанным работа содержит подробное описание и развитие формализованного математического, а именно, алгебраического аппарата, который обеспечит наилучшее соответствие характера данных и свойств вычислительных средств, для реализации массовой обработки данных, что подтверждает ее актуальность.

Целью диссертационной работы является создание алгебраических моделей и методов синтеза программно-аппаратных комплексов массовой обработки высокоактивных данных путем выбора на их основе подходящих архитектур и способов построения вычислительных процедур.

Для достижения поставленной цели необходимо решить следующие **задачи**:

1. Разработка алгебраического подхода к организации моделей данных и моделей вычислений, выработка системы требований к этому классу алгебраических моделей и проведение анализа универсальных алгебраических систем в большей мере отвечающих предъявляемым требованиям, разработка формальной алгебраической модели массовой обработки данных – теоретико-множественной файловой модели, и обоснование эффективности применения в качестве моделей данных и моделей вычислений алгебры многомерных матриц и реляционной алгебры.

2. Разработка системы методов доказательства соответствия предложенных алгебраических моделей с использованием алгебраических доказательств гомоморфизма или изоморфизма, и аксиоматических – доказательства эквивалентности теорий.

3. Разработка методов синтеза алгебраических моделей процессов массовой обработки данных на основе принципов динамического программирования.

4. Разработка методов и алгоритмов параллельной реализации операций массовой обработки данных, архитектур программно-аппаратных комплексов, реализующих эти операции и анализ приемлемости параллельной ре-

ализации процессов массовой обработки данных на предложенных архитектурах программно-аппаратных комплексов.

5. Разработка объектно- и предметно-ориентированных технологий, и средств методического и программно-технического обеспечения для массовой обработки высокоактивных структурированных данных

6. Реализация решения прикладных задач из различных предметных областей на основе предложенных методов массовой обработки данных.

Методы исследования

Проведенные в работе исследования базируются на математических моделях данных и вычислений, таких как алгебра многомерных матриц, реляционная алгебра, теория множеств. Для решения поставленных задач применялись аппарат математического анализа, теории алгебраических систем и метаматематики, методы системного, модульного, функционального и объектно-ориентированного программирования, а также технология параллельной обработки данных на многопроцессорных программно-аппаратных комплексах.

Основные положения, выносимые на защиту

1. Предложена новая теоретико-множественная (файловая) модель массовой обработки данных. Дано определение файла, как фактор-множества множества однотипных записей по отношению эквивалентности, порожденному множеством ключей. На основе этого определения сделаны формальные определения основных операций массовой обработки данных с использованием алгебраического и объектно-ориентированного подхода, которые позволили формализовать как операции над структурами данных высокого уровня – файлами, так и над составляющими их элементами – записями (кортежами).

2. Разработана алгебраическая модель данных и вычислений на основе алгебры многомерных матриц. Подтверждено соответствие этой модели теоретико-множественной и реляционной моделям.

3. Разработана аксиоматическая теория массовой обработки данных. Показано, что запросы на обработку данных – есть теоремы аксиоматической

теории. На основе теоремы об эквивалентности аксиоматических теорий показано, что многомерно-матричная и реляционная модели данных соответствуют друг другу.

4. Разработано обобщение алгоритма оптимизации последовательного умножения матриц на умножение многомерных матриц. Разработан метод на основе динамического программирования для синтеза оптимизированного процесса массовой обработки данных.

5. Предложен и исследован метод симметричного горизонтального распределения таблиц-операндов операции JOIN для последующего параллельного выполнения этой операции над фрагментами таблиц-операндов.

6. Предложены архитектуры программно-аппаратных комплексов для параллельного выполнения процессов массовой обработки данных, формализованных в различных моделях данных: многомерно-матричной, теоретико-множественной (файловой), реляционной.

Научная новизна

– Предложен новый метод формализации моделей данных и моделей вычислений, основанный на универсальных многоосновных алгебраических системах и объектно-ориентированном подходе к проектированию и разработке программно-аппаратных комплексов для решения задач массовой обработки данных.

– Предложена и разработана теоретико-множественная (файловая) алгебраическая система, которая используется для верификации соответствия известных и используемых на практике моделей данных и моделей вычислений.

– Описана алгебра многомерных матриц и предложен метод – абстрактная алгебраическая машина, который позволяет использовать в качестве элементов многомерных матриц различные, как простые, так и структурные (кортежи), типы данных.

– Разработаны алгебраический и аксиоматический методы подтверждения соответствия (изоморфизма или гомоморфизма) моделей данных и моделей вычислений и осуществлено подтверждение гомоморфизма, а для конкретных задач – изоморфизма, теоретико-множественной, многомерно-матричной и реляционной моделей на основе использования обоих методов.

– Разработано обобщение алгоритма выбора последовательности операций умножения матриц методом динамического программирования для (λ, μ)-свернутого произведения многомерных матриц, показано, что синтез оптимизированного процесса МОД может быть реализован методом динамического программирования. Приведен пример синтеза такого процесса.

– На основе параллельной реализации операций: выбран и обобщен для параллельной реализации (λ, μ)-свернутого произведения многомерных матриц алгоритм Кэннона; для файловой модели разработан алгоритм операции слияния нестрого упорядоченных файлов на основе симметричного горизонтального распределения файлов-операндов; для реализации симметричного горизонтального распределения разработан оригинальный эвристический алгоритм; показано, что на основе симметричного горизонтального распределения может быть эффективно распараллелена реляционная операция Join.

– Разработаны этапы построения программно-аппаратного комплекса для реализации МОД, архитектура для реализации многомерно-матричной модели данных, архитектуры для реализации простых (однопроходных) операций теоретико-множественной модели данных, архитектура для параллельной реализации операции слияния нестрого упорядоченных файлов в теоретико-множественной и реляционной моделях данных для различных алгоритмов, в том числе с использованием ассоциативных вычислительных систем.

Диссертационное исследование **соответствует следующим пунктам паспорта специальности** 2.3.5. «Математическое и программное обеспечение

вычислительных систем, комплексов и компьютерных сетей»:

– Модели, методы, архитектуры, алгоритмы, языки и программные инструменты организации взаимодействия программ и программных систем;

– Модели и методы создания программ и программных систем для параллельной и распределенной обработки данных, языки и инструментальные средства параллельного программирования;

– Модели, методы, алгоритмы, облачные технологии и программная инфраструктура организации глобально распределенной обработки данных.

Теоретическая и практическая значимость

Теоретическая значимость диссертации состоит в том, что на основе разработанных новых (файловой и многомерно-матричной) моделей данных, методов синтеза и оптимизации процессов массовой обработки данных, становится возможным построение программно-аппаратных комплексов для решения важных задач параллельной реализации массовой обработки данных.

Практическая значимость заключается в том, что предложенные подходы, методы и алгоритмы могут быть использованы для проектирования и разработки программно-аппаратных комплексов МОД на базе широкого спектра многоядерных и многопроцессорных систем с архитектурами SMP, MPP, NUMA (стоечные серверы и локальные грид-системы). Разработанные алгебраические модели могут применяться для разработки технологии коллективного проектирования больших многопроцессорных программно-аппаратных комплексов и для создания сложных программных систем различного назначения. На созданное на основе алгебраических моделей и методов программное обеспечение получены авторские свидетельства [51,52], на модели и методы – патенты [53,54]. Результаты работы нашли применение при проектировании и разработке линейки серверов и MPP-систем на основе сети рабочих станций для высокопроизводительных вычислений и дата-центров с целью повышения их производительности и уровня соответствия решаемым с их использованием задачам в АО «Т-Платформы». Было разработано и внедрено

программное обеспечение, соответствующее архитектурам этих вычислительных систем.

Достоверность полученных в диссертационной работе результатов подтверждается проведенными испытаниями программного обеспечения, реализующего разработанные методы и алгоритмы.

Апробация результатов работы осуществлена на более, чем 30 международных и всероссийских научных конференциях, в том числе:

– Международная научная конференция Системы компьютерной математики и их приложения, г. Смоленск, 2008-2024 гг.;

– Международная научно-практическая конференция «Современные информационные технологии и ИТ-образование», г. Москва, 2014-2017 гг.;

– Международная научная конференция «Конвергентные когнитивно-информационные технологии» в рамках международного конгресса «Современные проблемы компьютерных и информационных наук», г. Москва, 2018-2023 гг.;

– IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), г. Зеленоград, 2019-2021 гг.;

– III научно-практическая конференция с международным участием «Актуальные проблемы информатизации в цифровой экономике и научных исследованиях – 2022» г. Зеленоград;

– XVII International conference «Data Analytics and Management in Data Intensive Domains», DAMDID/RCDL'2015, г. Обнинск.

Личный вклад

Все выносимые на защиту результаты получены лично автором. В работах, опубликованных в соавторстве, вклад соискателя состоит в следующем:

– в [1-3] соискателем впервые предложена возможность использования алгебры многомерных матриц для моделирования процессов массовой об-

работки данных, введены понятия коэффициента активности данных и высокой активности данных;

– в [12] соискателем приведено доказательство возможности параллельной реализации умножения многомерных матриц как распределенной совокупности их сечений по скоттовым индексам;

– в [21, 35, 40] приводятся результаты исследований предложенного соискателем алгоритма симметричного горизонтального распределения данных для параллельной реализации операций типа Inner Join.

Публикации

По теме диссертации автором опубликовано 73 работы, в том числе 33 – в журналах, входящих в Перечень ВАК. Из них в течение последних пяти лет две ([33, 45]) – в изданиях, входящих в список К1, 11 ([28-32, 43, 44, 46-49]) – в изданиях, входящих в список К2; 9 работ ([34-42]) –в рецензируемых научных изданиях, входящих в системы цитирования Web of Science и Scopus, 1 монография. Получено 2 свидетельства о регистрации программ, 2 патента Российской Федерации на изобретение.

Структура и объем диссертации

Диссертация состоит из введения, 6 глав, заключения, списка литературы и приложений. Общий объем диссертации – 286 стр., в том числе 50 рисунков, 25 таблиц, 4 приложения. Список литературы состоит из 244 наименований.

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во введении обоснована актуальность темы диссертации, сформулированы основные задачи и научные результаты, дано краткое содержание глав.

Первая глава посвящена общим вопросам проектирования архитектуры параллельных систем баз данных, описана проблемная область и определены объекты исследования. В начале главы, в разделе 1.1 дается определение МОД как способа параллельной обработки больших объемов данных большим

числом процессоров. Констатируется, что в работе рассматривается и исследуется один из видов МОД – обработка структурированных данных. В разделе 1.2 в качестве основного объекта для формализации МОД принят ***структурированный файл***, состоящий из однотипных записей (структур). Файл рассматривается как некоторое абстрактное понятие, которому в реальности могут соответствовать не только «физические» файлы в операционных системах, но и таблицы в базах данных и более формальные объекты типа матриц. Введено понятие коэффициента активности файла как отношения $K_a = \frac{L_e}{L}$, где L – общее число записей в файле, L_e – число записей того же файла, подвергающихся обработке в процессе выполнения операции над этим файлом. Коэффициент активности определяет метод доступа к файлу. Если он высокий – близок к единице, то более эффективным будет последовательный доступ к файлу, в противном случае, когда он близок к нулю, целесообразно использовать произвольный доступ. Сделан вывод о том, что основное свойство МОД состоит в том, что коэффициенты активности всех обрабатываемых файлов близки к единице. Приведены неформальные описания основных операций над файлами: сортировка, выборка, сжатие, слияние строго упорядоченных файлов и слияние нестрого упорядоченных файлов. Приводятся примеры задач этого класса и сделан вывод о том, что при их решении в обработку включаются практически все данные, характеризующие объекты этих задач, и объемы обрабатываемых данных очень велики. Проведен анализ архитектур известных промышленных программно- аппаратных комплексов IBM Netezza и Oracle Exadata с позиции использования их для реализации МОД. Формируются цели работы, состоящие в разработке моделей данных, обеспечивающих наибольшее соответствие существующим моделям данных и архитектурам вычислительных комплексов, и разработке на основе этих моделей способов организации данных во внешней памяти и алгоритмов реализации операций, в наибольшей степени соответствующих структурам данных. Рассмотрена связь моделей МОД с архитектурами программно-аппаратных комплексов. Характерная

особенность рассматриваемой в работе разновидности МОД заключается в том, что данные, будучи высоко активными в контексте операций их обработки, в незначительной мере подвержены изменениям. Как правило, изменения накапливаются в специальном файле – корректуре, и перед решением основных задач производится необходимая корректировка файлов. Операция корректировки файла реализуется при помощи слияния строго упорядоченных файлов. Эта особенность позволяет рассматривать все файлы, участвующие в процессе обработки данных, как последовательные и упорядоченные. Упорядоченность обеспечивает ускорение выполнения всех операций над файлами. В самом медленном случае, когда при реализации не используется параллелизм, упорядоченность файлов позволяет выполнять все операции за один проход (просмотр) исходных и выходного файлов. При использовании параллельных вычислительных комплексов упорядоченность обеспечивает лучшее распределение файлов и их фрагментов по ресурсам комплекса и эффективное использование этих ресурсов. Повышение эффективности обработки данных при такой организации данных происходит по следующим причинам: файл занимает меньше места на внешнем запоминающем устройстве, поскольку нет необходимости использовать списковую структуру и хранить удаленные записи (мусор); уменьшается время чтения/записи, поскольку файл в идеальном случае занимает непрерывное пространство, но даже при необходимости фрагментации, если обслуживание внешней памяти организовано правильно, число фрагментов невелико. Поскольку внешняя память относится к типу блочных устройств ввода-вывода, за одно обращение к ней читается/записывается блок, содержащий большое число записей; упрощается обслуживание данных, так как становятся ненужными операции сборки мусора и переформирования файлов. При работе с системами управления базами данных (СУБД) невозможно точно знать, какие способы организации данных и методы доступа в них используются. Поэтому вводятся понятия *логически последовательной* организации файла и *логически последовательного* метода доступа. Это позволяет учитывать тот факт, что способы организации данных и

доступа к ним в различных СУБД могут существенно различаться. Рассмотрены основные для логически последовательного метода доступа к данным и выбираются, на основе известных классификаций, архитектуры программно-аппаратных комплексов, наилучшим образом реализующие этот метод доступа. В разделе 1.3 рассмотрены проблемы оптимизации процессов МОД. В разделе 1.4 для решения задачи формализации процессов МОД предложены теоретико-множественная (файловая), и многомерно-матричная модели данных и сформулированы требования, которым эти модели должны удовлетворять. А именно: соответствие моделям данных и вычислений; процедурность; параллелизм алгебраических операций; оптимизация запросов; объектно-ориентированная парадигма.

Вторая глава посвящена рассмотрению методов формализации алгебраических моделей данных и вычислений. В разделе 2.1 приводятся основные определения универсальных многоосновных алгебраических систем и приводятся примеры таких систем. Один из них – обычная алгебра матриц, которая представляет собой очень важный с точки зрения массовой обработки данных пример универсальной двухосновной алгебраической системы

$$\boldsymbol{U_M} = <M, R; \Omega; \Pi>,$$

где M – множество матриц, а R– множество действительных чисел (элементов матриц). Сигнатура Ω универсальной двухосновной алгебраической системы $\boldsymbol{U_M}$ имеет следующий вид:

$+ : R \times R \to R$ – аддитивная операция над элементами матриц;

$\times : R \times R \to R$ – мультипликативная операция над элементами матриц;

$\textcircled{t} : M \to M$ – транспонирование матрицы;

$\times : R \times M \to M$ – умножение матрицы на число;

$+ : M \times M \to M$ – сумма матриц;

$\times : M \times M \to M$ – произведение матриц;

$\textcircled{D} : M \to R$ – определитель матрицы.

Сигнатура предикатов П может содержать такие предикаты как «матрица M вырожденная» или «матрица M_1 может быть умножена на матрицу M_2». Проведен анализ современных интуитивных подходов к объектно-ориентированному моделированию, проектированию и программированию и осуществляется переход к формальному – алгебраическому подходу к объектно-ориентированным технологиям. А именно, абстрактный тип данных (объект, класс), в дальнейшем – АТД, определяется как универсальная многоосновная алгебраическая система. Приведено описание объектно-ориентированной технологии программирования для создания программно-аппаратных комплексов на основе универсальной (абстрактной) алгебраической машины – универсальной двухосновной алгебраической системы вида $\boldsymbol{E}$=<S, T; Ω; П>. Дан пример построения такого АТД для различных задача, основанных на поиске путей в графе. Основа S называется ***структурой***, а основа T – ***типом***. На основе этого подхода разработана технология построения алгебраических моделей сложных структур данных – кортежей. Кортеж – это конечный набор (t_1, …, t_n) длины n, каждый элемент которого t_i принадлежит некоторому типу T_i ($1 \le i \le n$). Нуль-кортеж рассматривается как кортеж, состоящий только из нейтральных элементов типов T_1, …, T_n,. На T_1,…, T_m – основных множествах универсальных алгебр или алгебраических систем, которые принято в языках программирования называть простыми типами определяется система функций: $f^{j}_{\alpha_1\dots\alpha_k,\beta_1\dots\beta_l}(x_{\alpha_1},\dots,x_{\alpha_k},y_{\beta_1},\dots,y_{\beta_l}):(T_{\alpha_1}\times\dots\times T_{\alpha_k}\times T_{\beta_1}\times\dots\times T_{\beta_l})\to T_i$. Сокращенная запись этих функций – $f^{j}_{\alpha_1\dots\alpha_k,\beta_1\dots\beta_l}$. Пусть кортеж c_1 длины p и кортеж c_2 длины q составлены из переменных x_1, …, x_p и y_1,…, y_q соответственно. Тогда кортеж c_3 длины r, построенный по правилу $(f^{1}_{\alpha_1\dots\alpha_k,\beta_1\dots\beta_l},\dots,f^{r}_{\alpha_1\dots\alpha_k,\beta_1\dots\beta_l})$, ее обозначение $f^{j}_{c_1,c_2}$, есть результат бинарной операции над кортежами c_1 и c_2 (c_3=c_1*c_2). Такая универсальная алгебраическая система есть АТД *Кортеж* – базовый объект, который может стать родоначальником множества объектов,

соответствующих реальным задачам. Приведено описание объектно-ориентированной технологии программирования для создания программно-аппаратных комплексов на основе универсальной (абстрактной) алгебраической машины – универсальной двухосновной алгебраической системы вида ***E***=<*S*, *T*; Ω; П>. Основа *S* называется ***структурой***, а основа *T* – ***типом***. Проведен анализ известных определений типизированного файла как интуитивных, так и формализованной на основе формы Бэкуса-Наура (БНФ). Дано формальное алгебраическое определение файла. Для этого даны определения: ***поля записи*** как пары $F=<N_i, A_i>$ ($i=1, \ldots, p$), где N_i – имя, а A_i – множество значений поля; з***аписи типа R*** как кортежа $R=\{F_1, \ldots, F_p\}$; ***экземпляра записи типа R*** как кортежа вида

$$R^* = \{< N_1, A_1^* >, ..., < N_p, A_p^* >\},$$

где $(A_i^* \in A_i,\ i = 1, ..., p)$; множество ***X*** экземпляров записей типа *R* как множества записей типа *R* или ***множества однотипных записей***. Для идентификации и сравнения записей введено понятие ***множества ключей*** $K = \{K_1, ..., K_m\}, (m < p)$ – множество полей записи *R*, такое, что $K_1 = F_{\alpha_1}, ..., K_m = F_{\alpha_m}$, причем все A_{α_i} $(i = 1, ..., m)$ – типы, на которых заданы отношения эквивалентности и порядка; элементы множества *K* называются ***ключами***, а кортеж $K^* = \{K_1^*, ..., K_m^*\}$, для элементов которого выполняется правило $K_i^* \in A_{\alpha_i}\ (i = 1, ..., m)$ – ***экземпляром множества ключей*** (K_i^* называется ***экземпляром ключа***). Две однотипные записи называются эквивалентными, если они содержат одинаковые экземпляры множества ключей. Задание на множестве однотипных записей ***X*** множества ключей *K* индуцирует разбиение ***X*** на ***классы эквивалентности***. Тогда ***файлом*** X_K называется фактор-множество множества однотипных записей *X* по отношению эквивалентности, порожденному множеством ключей *K*. Для определения операций над записями они рассматривается как пары кортежей $R=\{K, D\}$, где кортеж

$K = \{K_1, ..., K_m\}, (m < p)$ – множество ключей, а кортеж $D = \{F_{m+1}, ..., F_p\}$ – множество неключевых полей, участвующих в вычислениях, но не участвующих в идентификации и сравнении записей. Записи, в которых кортеж D – нуль-кортеж, физически в файле не присутствуют. Логически таким экземплярам множества ключей K соответствуют универсальные неопределенные записи Θ и классы эквивалентности Θ_{K^*}. Для построения бинарных операций над записями выбрана следующая модель бинарной операции над кортежами:

$$c_1(x_1,...,x_p) * c_2(y_1,...,y_q) = \begin{cases} c_3(f^{\gamma_1}_{\alpha_1...\alpha_{k_1},0},...,f^{\gamma_n}_{\alpha_1...\alpha_{k_n},0}),\ q = 0; \\ c_3(f^{\gamma_1}_{0,\beta_1...\beta_{l_1}},...,f^{\gamma_n}_{0,\beta_1...\beta_{l_n}}),\ p = 0; \\ c_3(f^{\gamma_1}_{\alpha_1...\alpha_{k_1},\beta_1...\beta_{l_1}},...,f^{\gamma_n}_{\alpha_1...\alpha_{k_n},\beta_1...\beta_{l_n}}),\ p \neq 0, q \neq 0. \end{cases}$$

Предложен следующий набор операций над файлами: сортировка (sort), выборка (sel), сжатие (quant), слияние строго упорядоченных файлов (ms) и слияние нестрого упорядоченных файлов (mns). Для всех операций введены строгие определения в терминах теоретико-множественной модели, то есть определяется функция *f*, определяющая вид класса эквивалентности выходного файла в зависимости от классов эквивалентности исходных файлов. Например, для операции слияния строго упорядоченных файлов построение функции *f* в общем случае может быть следующим:

$$f(X_{K^*}, Y_{K^*}) = \begin{cases} g_1(Y_{K^*}), \textit{если } X_{K^*} = \Theta_{K^*} \\ g_2(X_{K^*}), \textit{если } Y_{K^*} = \Theta_{K^*} \\ g_3(X_{K^*}, Y_{K^*}), \textit{если } X_{K^*} \neq \Theta_{K^*}, Y_{K^*} \neq \Theta_{K^*} \end{cases}$$

Эта формула обеспечивает формирование записей выходного файла Z_K по следующим правилам: одноместные функции g_1 и g_2 определены на записях файлов X_K и Y_K, соответственно, и реализуются при помощи функций $f^j_{0,\beta_1...\beta_l}$, $f^j_{\alpha_1...\alpha_k,0}$; двухместная функция g_3 определена на парах записей, принадлежащих декартову произведению $X_K \times Y_K$, и реализуется при помощи функций $f^j_{\alpha_1...\alpha_k,\beta_1...\beta_l}$; функции g_1, g_2 и g_3 обеспечивают формирование записи выходного файла с вычислением новых значений неключевых полей, из значений

неключевых полей записей X_{K^*} и Y_{K^*}; значения ключей все три функции переносят в выходную запись Z_{K^*} без изменений. Определенные таким образом операции над файлами позволяют построить двухосновную алгебраическую систему

F_{am}=<$\mathscr{F}$, $\mathscr{R}$; *sort*, *sel*, *quant*, *ms*, *mns*, *get*, *put*, +, ×; *eof*, *pred*, *eq*, *succ*>

или АТД, называемый абстрактной алгебраической файл-машиной. Здесь $\mathscr{F}$ – множество всех возможных файлов, $\mathscr{R}$ – множество всех записей, операции *get* :$\mathscr{F}\rightarrow \mathscr{R}$ и *put* : $\mathscr{F}\times \mathscr{R}\rightarrow\mathscr{F}$ – операции чтения и записи файла. Над записями определяются аддитивная и мультипликативная операции +, ×:$\mathscr{R}\times \mathscr{R}\rightarrow\mathscr{R}$ и предикаты *pred*, *eq*, *succ*:$\mathscr{R}\times \mathscr{R}\rightarrow$ {0, 1}, которые задают на множестве записей бинарные отношения *предшествования*, *эквивалентности* и *следования за.* В разделе 2.3, на основе теории многомерных матриц Н.П. Соколова, разрабатывается алгебраическая модель данных (вычислений). Доказан гомоморфизм теоретико-множественной и многомерно-матричной моделей, но основе которого доказано соответствие многомерно-матричной и реляционной моделей. Построена абстрактная алгебраическая многомерно-матричная машина

M_{am}=<$\mathscr{M}$, $\mathscr{E}$, $\mathscr{T}$, $\mathcal{S}$, $\mathcal{C}$, +, ×, ⊕, ⊗; $\mathbf{z}$, *ac*, *mc*, *pred*, *eq*, *succ*>,

где $\mathscr{M}$ – множество многомерных матриц, $\mathscr{E}$ – множество их элементов, $\mathscr{T}$ – транспонирование. В разделе 2.4 дано описание аксиоматической теории МОД. Показано, что запросы на получение данных есть теоремы в аксиоматических теориях – моделях МОД. На основе теоремы об эквивалентности аксиоматических теорий доказывается соответствие моделей МОД.

В третьей главе рассмотрены вопросы, связанные с алгебраическими методами оптимизации запросов на решение задач МОД посредством преобразования алгебраических выражений запросов и их параллельной реализации. В разделе 3.1 рассматриваются известные методы оптимизации запросов на обработку данных. В качестве примера сделано обобщение алгоритма оп-

тимизации последовательности умножения матриц[12] на случай (λ, μ)-свернутого произведения многомерных матриц.

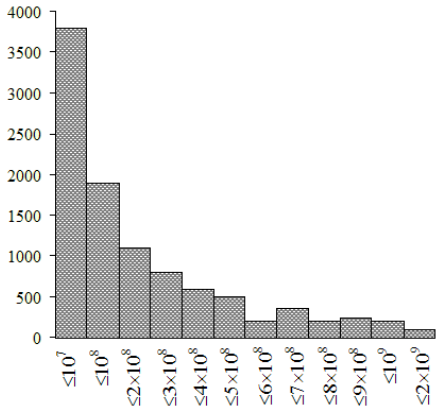


Рис. 1 Гистограмма распределения процессов МОД по сложности (суммарные длины просмотров)

В разделе 3.2 на основе применения принципа инвариантного погружения проводится доказательство того, что для синтеза оптимизированного алгебраического выражения запроса на МОД может быть использован метод динамического программирования. На основе экспериментального анализа показано, что особенности процессов МОД позволяют эффективно использовать этот метод оптимизации, а полученный процесс может быть улучшен использованием алгоритма ветвей-границ. Считая полученное значение сложности процесса «рекордом» можно привести перебор возможных процессов с целью поиска процесса меньшей сложности. Приведено описание вычислительного эксперимента, в ходе которого строились процессы обработки системы из десяти исходных файлов. Случайным образом генерировались группы по 10^4 процессов, и вычислялась стоимость каждого процесса. Были сгенерированы 10^3 групп, в которых распределения процессов по сложности отличались незначительно. Результат, полученный для одной из таких групп, приведен на рис. 1. В разделе 3.3 рассмотрены параллельные алгоритмы, которые могут быть использованы для реализации операций МОД. Показано, что, если в ис-

[12] Ахо А., Хопкрофт Дж., Ульман Дж. Построение и анализ вычислительных алгоритмов. – М.: Мир, 1979. 536 с.

ходных файлах каждый ключ принимает все, или почти все, значения из возможных, то матрицы – модели исходных файлов будут плотными, то есть не будут содержать (или будут содержать незначительное количество) нейтральных элементов, которые соответствуют универсальным неопределенным записям. В таком случае наилучшей будет многомерно-матричная модель. В противном случае, многомерная матрица будет разреженной, и использование файловой модели или реляционной модели SQL будет более эффективным. Проведено обобщение алгоритма Кэннона параллельного умножения матриц на случай (λ, μ)-свернутого произведения многомерных матриц. Предложен способ представления многомерных матриц в виде условно-трехмерных. Доказано, что результат (λ, μ)-свернутого произведения двух многомерных матриц A и B размерностей p и q, при наличии скоттовых индексов (λ>0), есть r-мерная матрица C ($r=p+q-\lambda-2\mu$), составленная из ($r-\lambda$)-кратных сечений, каждое из которых есть произведение ($p-\lambda$)-кратного сечения матрицы A и ($q-\lambda$)-кратного сечения матрицы B. При этом наборы значений индексов $s_1, ..., s_\lambda$ сечений-сомножителей и сечения-результата совпадают. В разделе 3.4 проведен анализ алгоритмов, реализующих операцию слияния нестрого упорядоченных файлов. Предложен способ параллельной реализации этой операции на основе принципа симметричного горизонтального распределения файлов-операндов. если операция реализуется посредством N параллельных процессов, то файлы X_K и Y_K должны быть разделены на N фрагментов, которые удовлетворяют следующим требованиям: ***фрагменты*** файлов X_K и Y_K с номером i (i=1, …, N) связаны с процессом P_i и содержат классы эквивалентности с одинаковыми значениями экземпляров множества ключей K, для любого i из $X_{K^*_{(i)}} \neq \Theta$ следует $Y_{K^*_{(i)}} \neq \Theta$; ***каждый*** класс эквивалентности полностью расположен в одном фрагменте соответствующего файла; ***каждый*** процесс формирует свой фрагмент файла-результата Z_K, эти фрагменты не пересекаются, а файл-результат Z_K формируется как их объединение. Реализация

распределения классов эквивалентности файлов X_K и Y_K по фрагментам осуществляется на основе системных метаданных, которые обеспечивают индексно-последовательное представление данных из этих файлов. Предложен эвристический алгоритм распределения данных, который обеспечивает равномерное распределение фрагментов по процессорам, и приведено описание ряда экспериментов, подтвердивших его эффективность. Приведена его реализация языковыми средствами реляционных СУБД. Предложена стратегия повышения эффективности процессов МОД, состоящая в том, что: выбирается модель данных уровня проектирования и разрабатывается архитектура базы данных; проводится анализ данных и выбирается одна из промежуточных моделей по следующему правилу: если реальные исходные данные таковы, что агрегаты данных содержат практически все значения ключей, то выбирается многомерно-матричная модель, в противном случае, выбирается файловая модель данных; выражения запросов транслируются из представления на языке модели данных, использованной для проектирования, в язык выбранной промежуточной модели данных; производится оптимизация оттранслированных выражений запросов либо методом синтеза эффективных выражений запросов, либо преобразования имеющихся выражений запросов в выражения, порождающие более эффективные процессы обработки данных; выбирается модель вычислений и алгоритмы составляющих запросы операций, которые соответствуют выбранной модели вычислений и эффективно в ней реализованы.

В четвертой главе в разделе 4.1 рассмотрены этапы построения программно-аппаратных комплексов, реализующих МОД. А именно: выбор одной из трех (файловой, реляционной или многомерно-матричной) модели данных на основе свойства «плотности» ключей данных, которые присущи решаемой задаче; на основе выбранной модели данных определяется модель вычислений, то есть определяется набор операций МОД, выбираются или разрабатываются алгоритмы реализации операций, оценивается вычислительная сложность этих алгоритмов; на основе выбранной модели данных и архитек-

туры программно-аппаратного комплекса "собирается" из имеющихся вычислительных средств действующая модель программно-аппаратного комплекса, определяются операционные системы, системы программирования и системы управления базами данных; разрабатывается программное обеспечение, реализующее алгоритмы распределения данных и взаимодействия процессоров и потоков команд и данных, разрабатывается прикладное программное обеспечение, реализующее операции и процессы МОД для конкретного класса задач.

В разделе 4.2 предлагается архитектура программно-аппаратного комплекса для реализации многомерно-матричной модели данных. Подробно рассматривается архитектура, ориентированная на реализацию операции (λ, μ)-свернутого произведения многомерных матриц на основе сечений по совокупности скоттовых индексов. В качестве примера рассмотрен случай (1, 1)-свернутого произведения условно-трехмерных матриц, где E=3, $A = \|A_{lsc}\|$, $B = \|B_{scm}\|$ и $C = \|C_{lsm}\|$ – трехмерные матрицы, индексы которых принимают значения от 0 до 2, а элементы – блоки исходных трехмерных матриц и матрицы-результата.

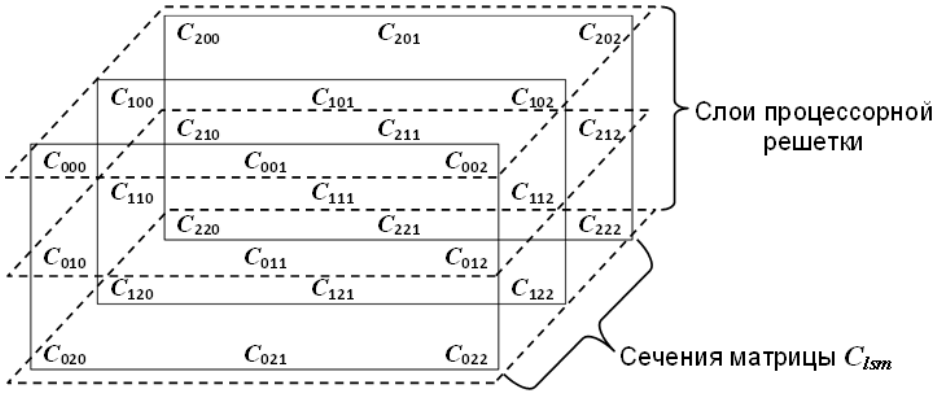


Рис. 2 Распределение блоков матрицы C_{lsm} по слоям процессорной решетки

В разделе 4.3 рассмотрены возможные архитектуры программно-аппаратных комплексов для реализации простых операций теоретико-множественной модели данных: внешней сортировки, выборки, сечения и слияния строго упорядоченных файлов. В разделе 4.4 предложена архитектура программно-

аппаратного комплекса для параллельной реализации операции слияния нестрого упорядоченных файлов в теоретико-множественной и реляционной моделях данных на основе принципа симметричного горизонтального распределения таблиц-операндов и приведено описание параллельной реализации этой операции для высокоактивных данных с использованием алгоритма черпака и метода доступа ISAM. Хост-машина читает индексные файлы и сравнивает экземпляры множества ключей M^*. Индексы классов эквивалентности, экземпляры множества ключей которых совпали, передаются одному из вычислителей, параллельно реализующих декартово произведение. Процесс вычисления состоит из нескольких потоков (рис. 3).

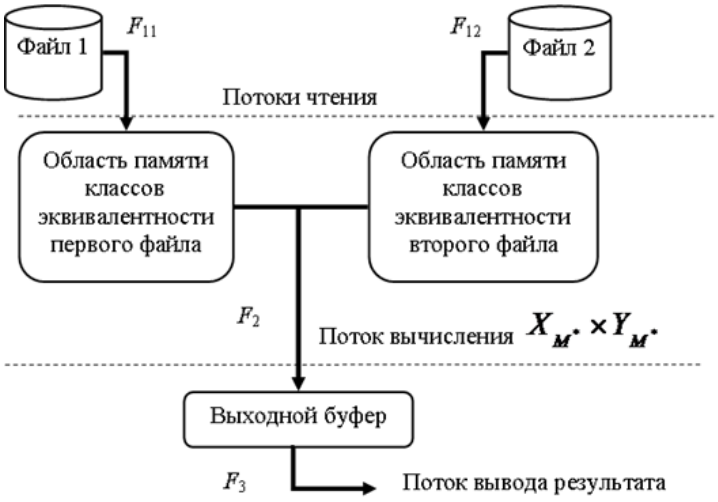


Рис. 3 Взаимодействие процессов в операции соединения

1. Потоки чтения файлов в память F_{11} и F_{12} независимо друг от друга считывают классы эквивалентности файлов в буферные зоны памяти. Адреса классов эквивалентности и количество записей в них получены от хост-машины.

2. Поток F_2 вычисляет декартово произведение классов эквивалентности X_{M^*} и Y_{M^*}. Этот поток может состоять из нескольких одинаковых параллельных нитей, каждая из которых соединяет одну запись класса эквивалентности X_{M^*} со всеми записями класса эквивалентности Y_{M^*}, помещая полученные записи в выходной буфер файла Z_M.

3. Поток вывода F_3 по мере поступления записей в выходной буфер перемещает их на внешний накопитель, на котором должен располагаться класс эквивалентности Z_{M^*}.

Предложена и рассмотрена двухуровневая параллельная конвейерная реализация последовательности операций слияния нестрого упорядоченных файлов на основе стандартных архитектур вычислительных систем и алгоритма черпака. Исходные файлы разбиваются на фрагменты, которые содержат классы эквивалентности, соответствующие одним и тем же экземплярам множества ключей. Для каждой пары фрагментов исходных файлов выделяется свой фрагмент-процессор, который выполняет декартовы произведения классов эквивалентности и формирует записи выходного файла (рис. 4).

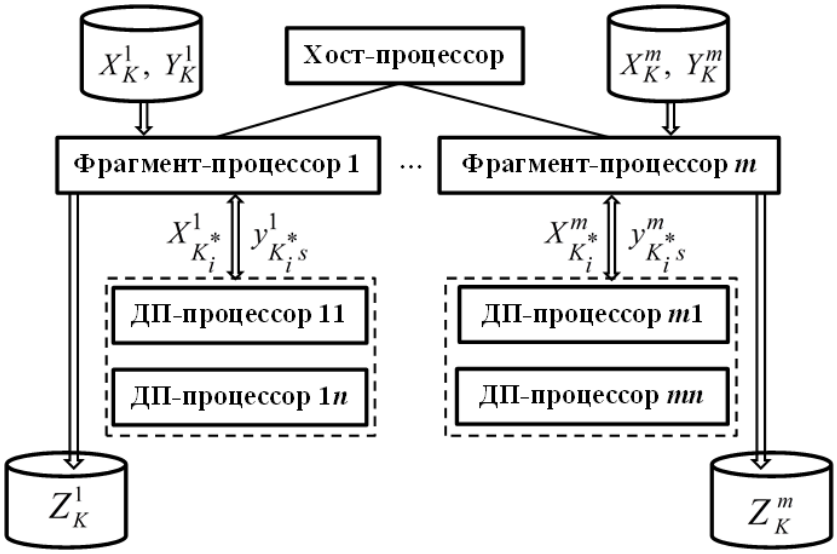


Рис. 4 Архитектура программно-аппаратного комплекса для слияния нестрого упорядоченных файлов

Второй уровень распараллеливания обусловлен тем, что с каждой записью ведомого файла обрабатываются все, хранящиеся в черпаке, записи ведущего файла. Если имеется некоторое количество процессоров, которые называются ДП-процессорами, так как реализуют декартово произведение классов эквивалентности, черпак делится на такое же количество частей.

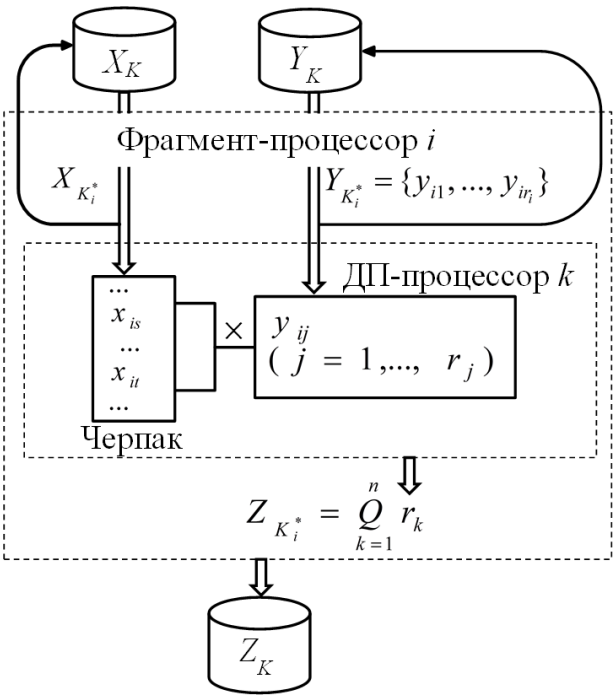


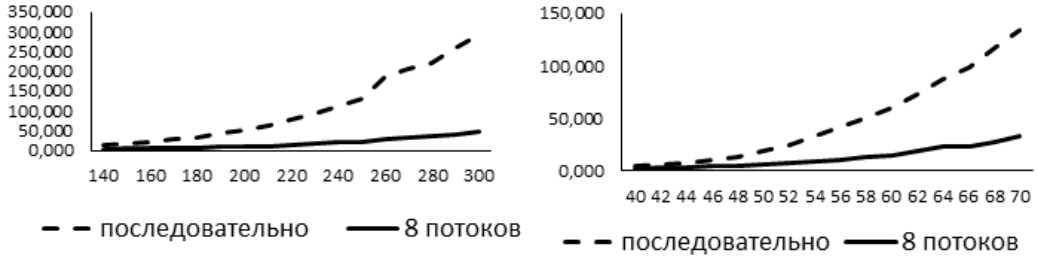
Рис. 5. Взаимодействие фрагмент-процессора с ДП-процессорами

Каждый из процессоров выполняет декартово произведение своей части черпака и класса эквивалентности ведомого файла. ДП-процессор поочередно обрабатывает все записи класса эквивалентности ведомого файла со всеми записями черпака, формируя при этом последовательность записей выходного файла (r_1, …, r_n), где $n = L(X_K^i) \times L(Y_K^i)$ – произведение объемов i-тых классов эквивалентности исходных файлов. Эти записи передаются фрагмент-процессору, который формирует из них класс эквивалентности выходного файла (рис. 5).

Предложены параллельная реализация последовательности операций слияния нестрого упорядоченных файлов на основе конвейерной архитектуры и параллельная реализация операции слияния нестрого упорядоченных файлов на основе архитектуры вычислительных систем с ассоциативным распределением ресурсов.

В пятой главе приведены результаты экспериментов, проведенных с целью проверки полученных в предыдущих главах результатов. В разделе 5.1 приведены результаты эксперимента для анализа параллельной реализации операции умножения многомерных матриц. Эксперимент проводился для

трехмерных (A_{lsc}, B_{scm}) и четырехмерных ($A_{ls_1s_2c}$, $B_{s_1s_2cm}$) числовых матриц. Индексы трехмерных матриц принимали значения от 10 до 300, а четырехмерных – 10 до 70. На каждом шаге эксперимента выполнялись последовательное и параллельное (8 и 27 потоков) умножение многомерных матриц. Для трехмерных матриц выполнялось (1, 1)-свернутое произведение, для четырехмерных – (2, 1)-свернутое произведение. Программно-аппаратные комплексы создавались как виртуальные машины Microsoft Azure на основе процессора Xeon E5-2673. Использовались конфигурации с восьмью и шестнадцатью ядрами. Результаты экспериментов (рис. 6 и 7), показывают, что с увеличением размерности индексов возрастает эффективность параллельного алгоритма.



А) трехмерные матрицы Б) четырехмерные матрицы

Рис. 6 Зависимость производительности алгоритма от размерности индексов

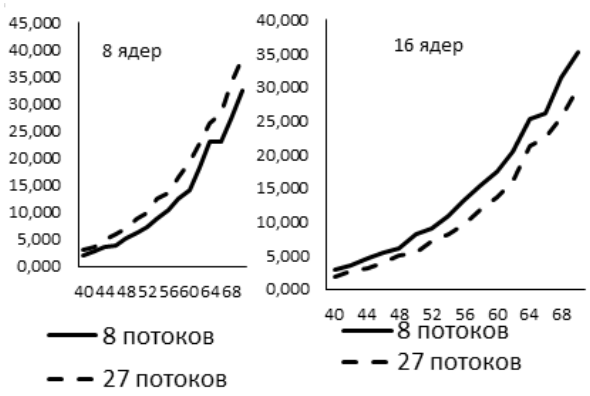


Рис. 7 Зависимость производительности алгоритма от числа вычислителей

В разделах 5.2 и 5.3 приведены результаты вычислительного эксперимента для анализа параллельной реализации операции слияния нестрого упорядоченных файлов на SMP-системе. Исходные данные представляли собой

таблицы базы данных Microsoft Sql Server, а операция слияния нестрого упорядоченных файлов реализовывалась операцией Join. Подфайлы-операнды каждой пары содержали классы эквивалентности с одинаковыми значениями экземпляров множества ключей, но различными количествами записей. На каждом шаге генерировалось по четыре пары подфайлов, которые одновременно обрабатывались четырьмя параллельными процессами, результаты которых объединялись в один файл-результат. Затем все подфайлы-операнды объединялись в два файла операнда, и выполнялся один процесс, реализовавший операцию последовательно. Количество обрабатываемых на каждом этапе записей приведено в табл. 1.

Табл. 1 Объемы данных на каждом шаге эксперимента

Шаг	Количество записей (тыс.)	
	Подфайлы пар (X_M, Y_M)	Файлы (X_M, Y_M)
1	200	400
2	400	800
3	600	1200
4	800	1600
5	1200	2400

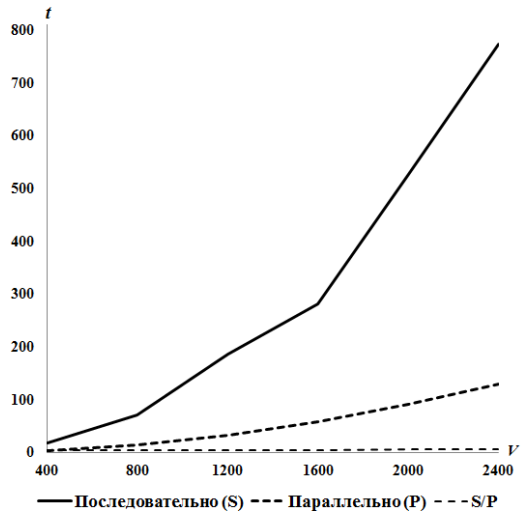


Рис. 8 Зависимость времени выполнения операции слияния от объема данных

В ходе эксперимента были получены приведенные на рис. 8 соотношения между объемами базы данных (в тысячах записей) и временами обработки

(в сек.) этой базы четырьмя параллельными процессами и одним последовательным процессом. Параллельная реализация операции слияния нестрого упорядоченных файлов четырьмя процессами требует в среднем в пять раз меньше времени, чем последовательной реализация. Анализ загрузки процессами ресурсов (рис. 9), показал, что каждый процесс выполняется на одном виртуальном ядре, не более чем на половину используя его ресурсы. Из этого следует, что число параллельно выполняющихся процессов может быть вдвое большим, чем число выделенных для реализации операции ядер.

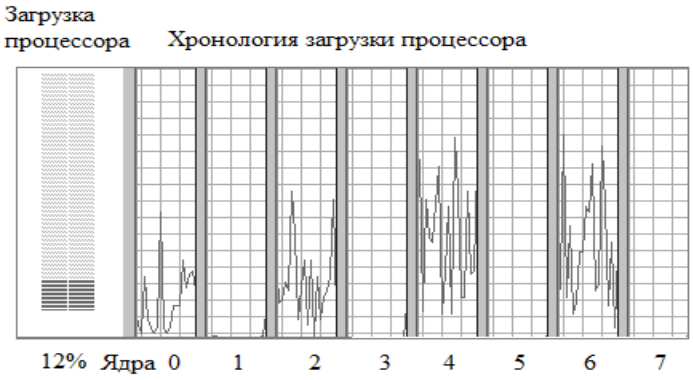


Рис. 9 График загрузки ресурсов процессора

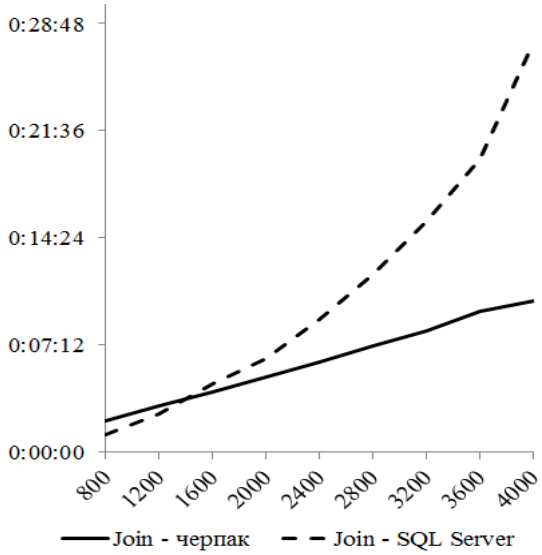


Рис. 10 Время реализации операции слияния нестрого упорядоченных файлов и стандартной операции Join

Результаты эксперимента на рис. 10 подтверждают, что применение предложенного метода параллельной реализации операции слияния нестрого

упорядоченных файлов алгоритмом черпака для МОД и высоко активных данных более эффективно, чем реализация стандартными средствами распараллеливания операции Join, реализованными в СУБД. При реализации операции слияния нестрого упорядоченных файлов алгоритмом черпака загрузка ядер процессора оказалась весьма высокой, что показано на рис. 11.

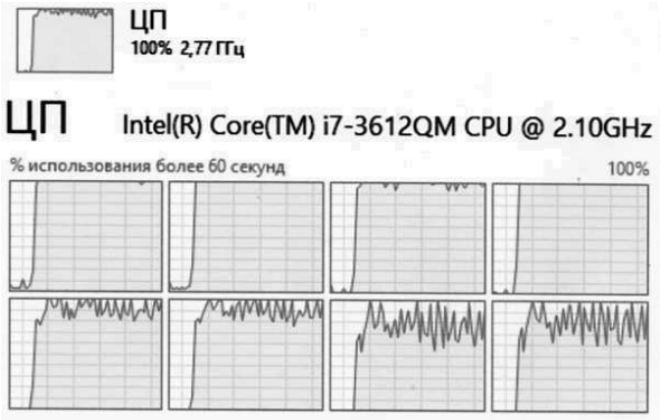


Рис. 11 Загрузка процессоров (ядер) в виртуальном программно-аппаратном комплексе

В разделе 5.4 проведен анализ параллельной реализации операции слияния нестрого упорядоченных файлов с использованием MPP-архитектуры в облачной среде Microsoft Azure. Приведена архитектура программно-аппаратного комплекса (рис. 12), в состав которого входили пять независимых баз данных на различных виртуальных машинах. Из них четыре использовались для хранения фрагментов основной БД, полученных в результате симметричного горизонтального распределения таблиц. Эксперимент проводился для таблиц P и Q, число строк в которых изменялось от 4×10^5 до 25×10^5. На рис. 13 приведены экспериментальные данные, подтверждающие тот факт, что применение предложенного автором метода параллельного выполнения операции JOIN дает существенное повышение производительности при реализации задач массовой обработки данных.

<table>
<tr><td>Приложение</td><td colspan="3">Облако</td></tr>
<tr><td rowspan="2">Процедура симметричного горизонтального распределения</td><td rowspan="2">Основная БД</td><td>Таблицы</td><td>P, Q</td></tr>
<tr><td>Запросы на создание индексных таблиц</td><td>indP, indQ, CommonInd</td></tr>
<tr><td rowspan="2">Поток 1</td><td rowspan="2">Фрагмент 1 БД</td><td>Таблицы</td><td>P_1, Q_1</td></tr>
<tr><td>Запрос</td><td>P_1 Inner Join Q_1</td></tr>
<tr><td>…</td><td></td><td></td><td></td></tr>
<tr><td rowspan="2">Поток N</td><td rowspan="2">Фрагмент N БД</td><td>Таблицы</td><td>P_N Inner Join Q_N</td></tr>
<tr><td>Запрос</td><td></td></tr>
</table>

Рис. 12 Структура виртуального программно-аппаратного комплекса для массовой обработки данных

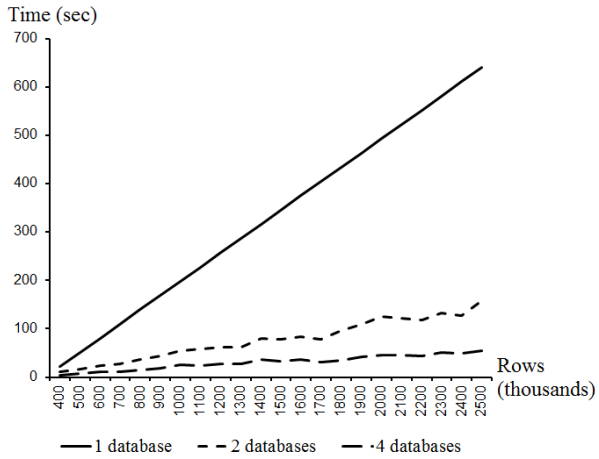
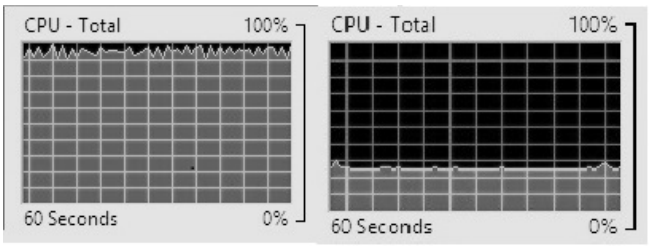


Рис. 13 Сравнительный анализ времени последовательного и параллельного выполнения операции JOIN

В разделе 5.5 приведен анализ параллельной реализации операции слияния нестрого упорядоченных файлов с использованием SMP-архитектуры в облачной среде. Была создана виртуальная машина, на которой производилось сравнение реализации алгоритма черпака с фиксированным размером класса эквивалентности и реализации операции Join в Microsoft SQL Server. Для реализации предложенного параллельного алгоритма операции слияния нестрого упорядоченных файлов предложена архитектура программно-аппаратного комплекса, основанного на архитектуре SMP и реализованного в облачной системе Microsoft Azure. В роли файлов выступали таблицы в базах данных СУБД Microsoft SQL Server.

Рис. 14 Зависимость времени выполнения операции слияния нестрого упорядоченных файлов от размеров классов эквивалентности



А) алгоритма черпака Б) стандартными средствами

Рис. 15 График загрузки процессора

Результаты эксперимента, представленные на рис. 14 и 15, показывают, что предложенные алгоритм и архитектура программно-аппаратного комплекса обеспечивают более эффективную обработку высоко активных данных, чем стандартные средства СУБД. В разделе 5.6 приведен анализ параллельной реализации операции слияния нестрого упорядоченных файлов с использованием SMP-архитектуры и многоядерных графических процессоров.

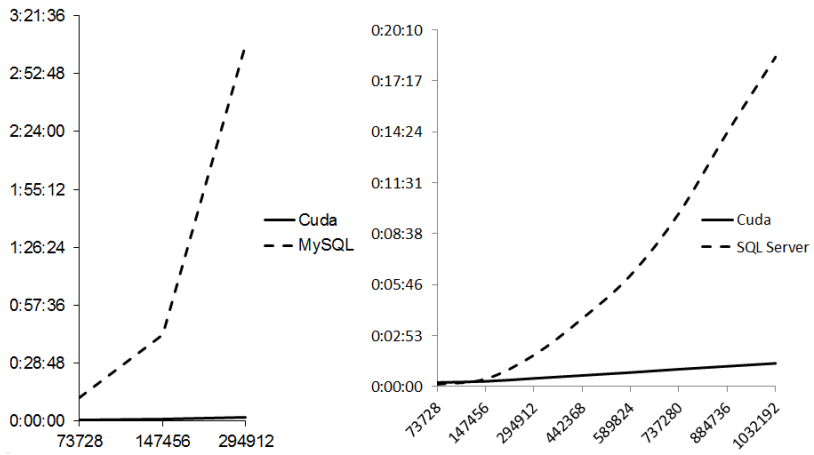


Рис. 16. Сравнение времени выполнения операции JOIN средствами СУБД и технологии CUDA

Проведенный экспериментальный анализ предложенных в предыдущих главах алгебраических методов массовой обработки данных показал их эффективность.

В шестой главе рассмотрено применение предложенного подхода к решению практических прикладных и системных задач массовой обработки данных. В разделе 6.1 рассматривается использование предложенного метода для решения задач о кратчайшем пути. Рассмотрено решение традиционной задачи с использованием матричной формы метода Флойда-Уоршелла. Показано, что при последовательном возведении матрицы весов ребер графа в (1, 0)-свернутую степень, набор значений индексов отличного от нейтрального элемента k-той степени этой матрицы есть последовательность номеров вершин графа. Это есть путь, начинающийся в вершине, номер которой соответствует значению первого индекса, заканчивающийся в вершине, номер которой соответствует значению последнего индекса, и проходящий через k–2 вершины. Значение элемента есть стоимость этого пути. Рассмотрен случай, когда число ребер таково, что матрица весов сильно разрежена. На основе того факта, что для этой задачи алгебра многомерных матриц и реляционная алгебра изоморфны, предложено ее решение средствами языка Transact-SQL. Вычислительная сложность этих алгоритмов достаточно высока, однако использование предложенных методов распараллеливания умножения многомерных матриц и операции Join, позволяет получит результаты за приемлемое время. В разделе 6.2 показано, что использование предложенного в первом разделе метода возведения матрицы весов графа в (1, 0)-свернутую степень может быть использовано для решения задачи вывода ассоциативных правил. В разделе 6.3 рассмотрено использование предложенного метода для решения задачи поиска изображений в базах данных на основе перцептивного хеширования. Каждому изображению ставится в соответствие двоичное число, содержащее 64 и более двоичных цифр (хеш-код), которое служит идентификатором (ключом) изображения в БД. Два изображения считаются близкими, если

расстояние Хэмминга между их хэш-кодами не превосходит заданного значения. Введено понятия веса ключа, которое позволило использовать метод симметричного горизонтального распределения для параллельной обработки запросов к БД. Предложен и реализован метод параллельного сравнения ключей изображений с использованием SIMD-регистров процессора. Предложена архитектура программно-аппаратного комплекса для поиска изображений в базах данных. Раздел 6.4 посвящен реализации алгоритма шифрования Хилла на основе алгебры многомерных матриц. Приведены дополнительные элементы алгебры многомерных матриц: понятие трансверсали, детерминанта, обратной многомерной матрицы. Показано, что криптостойкость обобщенного алгоритма Хилла повышается за счет свойств многомерных матриц, таких как количество и размерности индексов, выбора скоттовы и кэлиевых индексов, возможности построения различных единичных и, соответственно, обратных матриц.

Заключение посвящено анализу основных положений диссертационной работы и перспективным направлениям внедрения результатов работы для решения таких практических задач, как создание многомерно-матричных моделей баз данных с использованием современных облачных средств конструирования виртуальных кластеров на основе тензорных и графических процессоров и современных программных средств, реализующих алгебру тензоров, а также разработки САПР с применением предложенных в диссертации процессов массовой обработки на основе многомерно-матричной и реляционных моделей.

ОСНОВНЫЕ ПУБЛИКАЦИИ ПО ТЕМЕ ДИССЕРТАЦИИ

1. Гендель Е. Г., Мунерман В. И. Применение алгебраических моделей для синтеза процессов обработки файлов // Управляющие системы и машины, 1984. № 4. С.69-72.

2. Гендель Е. Г., Мунерман В. И., Шкляр Б.Ш. Оптимизация процессов обработки данных на базе алгебраических моделей // Управляющие системы и машины, 1985. № 6. С. 91-95.

3. Мунерман В.И. Использование матричной модели для обоснования выбора структур вычислительных систем и оптимизации процессов обработки файлов: диссертация на соискание ученой степени кандидата технических наук: 05.13.13. - Москва, 1985. - 171 с.: ил.

4. Левин Н. А., Мунерман В. И. Метод логически последовательного доступа к данным // Системы высокой доступности, 2011. Т. 7. № 4. С. 65-67.

5. Мунерман В. И. Объектно-ориентированная модель массовой обработки данных // Системы высокой доступности, 2011. Т. 7. № 4. С. 72-74.

6. Комиссарова А. Н., Мунерман В. И. Мера вычислительной сложности массовой обработки данных // Системы высокой доступности, 2011. Т. 7. № 4. С. 68-71.

7. Мунерман В. И. Многомерно матричная модель массовой обработки данных // Системы высокой доступности, 2012. Т. 8. № 3. С. 019-022.

8. Мунерман В. И. Построение архитектур программно-аппаратных комплексов для повышения эффективности массовой обработки данных // Системы высокой доступности, 2014. Т. 10. № 4. С. 3-16.

9. Мунерман В. И. Опыт массовой обработки данных в облачных системах (на примере Windows Azure) // Системы высокой доступности, 2014. Т. 10. № 2. С. 3-8.

10. Мунерман В. И., Мунерман, Д. В., Синицын, И. Н., Чукляев И. И., Параллельная реализация задач интегрированной логистической поддержки (CALS) // Современные информационные технологии и ИТ-образование, 2014. № 10. С. 548-554.

11. Захаров В. Н., Мунерман В. И. Модели и методы параллельной обработки структурированных больших данных // Современные информационные технологии и ИТ-образование, 2014. № 10. С. 534-547.

12. Мунерман В. И. Архитектура программно-аппаратного комплекса для массовой обработки данных на базе многомерно-матричной модели // Системы высокой доступности, 2015. Т. 11. № 2. С. 13-18.

13. Захаров В. Н., Мунерман В. И. Параллельный алгоритм умножения многомерных матриц // Современные информационные технологии и ИТ-образование, 2015. Т. 2. № 11. С. 384-390.

14. Мунерман В. И., Самойлова Т. А. Параллельная реализация решения оптимизационных задач средствами баз данных // Системы высокой доступности, 2015. Т. 11. № 1. С. 18-22.

15. Мунерман В. И., Мунерман Д. В. Алгебраический подход к построению программно-аппаратных комплексов для повышения эффективности массовой обработки данных // Современные информационные технологии и ИТ-образование, 2015. Т. 2. № 11. С. 391-396.

16. Мунерман В. И., Самойлова Т. А. Обучение методам разработки информационно-аналитических систем на основе облачных технологий (на примере MICROSOFT AZURE) //Системы высокой доступности, 2016. Т. 12. № 4. С. 3-11.

17. Захаров В. Н., Мунерман В. И. Алгебраический подход к формализации параллелизма данных // Современные информационные технологии и ИТ-образование, 2016. Т. 12. № 1. С. 72-79.

18. Емельченков Е. П., Макаров А.И., Мунерман В. И. Объектно-ориентированный подход к представлению баз данных в не первой нормальной форме // Современные информационные технологии и ИТ-образование, 2015. Т. 2. № 11. С. 66-70.

19. Мунерман В. И., Мунерман Д. В. Параллельная реализация операций обработки файлов // Современные информационные технологии и ИТ-образование, 2016. Т. 12. № 2. С. 84-90.

20. Мунерман В. И. Реализация параллельной обработки данных в облачных системах // Современные информационные технологии и ИТ-образование, 2017. Т. 13. № 2. С. 57-63.

21. Мунерман В. И. Алгебраический подход к подготовке данных для вывода ассоциативных правил // Системы высокой доступности, 2017. Т. 13. – № 3. С. 34-37.

22. Мунерман В. И., Мунерман Д. В. Параллельная реализация симметричного горизонтального распределения данных на основе сетевых технологий // Современные информационные технологии и ИТ-образование, 2017. Т. 13. № 3.С. 38-43.

23. Мунерман В. И. Аксиоматический метод формализации массовой обработки данных в системах высокой доступности // Системы высокой доступности, 2017. Т. 13. № 2. С. 56-62.

24. Макаров А. И., Миронов А. И., Мунерман В. И. Реализация параллелизма на уровне задач в системах высокой доступности // Системы высокой доступности, 2018. Т. 14. № 5. С. 42-45.

25. Митенков В. Б., Митенков К. А., Мунерман В. И. Параллельная подготовка данных для расчета виброзащитных характеристик в ходе летных испытаний // Системы высокой доступности, 2018. Т. 14. № 5. С. 46-49.

26. Мунерман В. И., Самойлова Т. А. Алгебраический подход к алгоритмизации задач маршрутизации // Системы высокой доступности, 2018. Т. 14. № 5. С. 50-56.

27. Емельченков Е. П., Мунерман В. И. Подход к анализу систем высокой доступности // Системы высокой доступности, 2018. Т.14. № 5. С. 36-41.

28. Мунерман В. И., Мунерман Д. В. Анализ алгоритма оптимального распределения // Современные информационные технологии и ИТ-образование, 2019. Т. 15. № 3. С. 619-625.

29. Мунерман В. И., Самойлова Т. А. Реализация алгоритма шифрования Хилла на основе алгебры многомерных матриц //Системы высокой доступности, 2019. Т. 15. № 1. С. 21-27.

30. Емельченков Е. П., Мунерман, В. И., Мунерман, Д. В., Самойлова, Т. А. Объектно-ориентированный подход к разработке моделей данных //Современные информационные технологии и ИТ-образование, 2020. Т. 16. № 3. С. 564-574.

31. Гончаров Е. И., Ильин П. Л., Мунерман В. И., Самойлова Т. А. Подход к повышению эффективности алгоритмов свертки в современных системах высокой доступности // Системы высокой доступности, 2021. Т. 17. № 1. С. 15-24.

32. Емельченков Е. П., Мунерман В. И., Мунерман Д. В., Самойлова Т. А. Один метод построения циклов в графе // Современные информационные технологии и ИТ-образование, 2021. Т. 17. № 4. С. 814-823.

33. Морозов С. А., Мунерман В. И., Симаков В. А. Экспериментальный анализ многомерно-матричного подхода к построению маршрутов в графе // Известия высших учебных заведений. Электроника, 2022. Т. 27. № 5. С. 676-686.

34. Morozov S. A., Munerman V. I., Simakov V. A. Experimental Analysis of the Multidimensional-Matrix Approach to Construct Routes in a Graph // Russian Microelectronics, 2023. Vol. 52. No. 7. P. 716-721.

35. Zakharov V., Kirikova A., Munerman V., Samoilova T.A. Architecture of Software-Hardware Complex for Searching Images in Database //2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), 2019. IEEE. 1735-1739.

36. Munerman V., Munerman D. Realization of Distributed Data Processing on the Basis of Container Technology //2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), 2019. IEEE. 1740-1744.

37. Kirikova A., Mironov A., Munerman V. The Method of Composition Hash-functions for Optimize a Task of Searching Images in Dataset //2020 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), 2020. IEEE. 1983-1986.

38. Grigoryeva G., Khodchenkov V., Mironov A., Munerman V. Creating a Vector Processor Based on Quantum Computing // 2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), 2019. IEEE. 1745-1748.

39. Goncharov E., Iljin P., Munerman V. Multidimensional Matrix Algebra Versus Tensor Algebra or $\mu > 0$ // 2020 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), 2020. IEEE. 1949-1954.

40. Goncharov E., Munerman V., Yakovlev G. Software and Hardware Complex for Calculating Convolutions by Methods Multidimensional Matrix Algebra // 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus), 2021. IEEE. 2176-2180.

41. Munerman V., Munerman D., Samoilova T. The Heuristic Algorithm For Symmetric Horizontal Data Distribution // 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus), 2021. IEEE. 2161-2165.

42. Iljin P., Munerman V. An Heuristic Algorithm of the Details Distribution by Machines // 2021 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (ElConRus), 2021. IEEE. 2085-2089.

43. Мунерман В. И., Мунерман Д. В. Обобщение одного алгоритма параллельного умножения матриц в алгебре многомерных матриц // Современные информационные технологии и ИТ-образование,2022.Т.18. №3.С.566-577.

44. Захаров В. Н., Мунерман В. И., Самойлова Т. А. Параллельные методы вывода ассоциативных правил в технологиях in-database и in-memory // II Международная научная конференция «Конвергентные когнитивно-информационные технологии»: сб. тр. науч. конф., 2017. С. 219-225.

45. Макаров А. И., Мунерман В. И. Использование (0, μ)-свернутого произведения многомерных матриц для решения задач теории графов // Известия высших учебных заведений. Электроника. Управляющие системы и машины, 2023. Т. 28. №. 5. С. 659-669.

46. Гончаров Е. И. Ильин П. Л., Мунерман В. И., Самойлова Т. А. Подход к повышению эффективности алгоритмов свертки в современных системах высокой доступности // Системы высокой доступности, 2021. Т. 17. №. 1. С. 15-24.

47. Мунерман В.И., Мунерман Д.В. Аксиоматический метод доказательства соответствия формализованных в различных моделях данных запросов // Современные информационные технологии и ИТ-образование, 2023. Т. 19. № 4.

48. Морозов С.А., Мунерман В.И. Параллельная реализация алгоритма построения всех маршрутов в графе средствами реляционной алгебры // Современные информационные технологии и ИТ-образование,2023.Т.19. № 4.

49. Гончаров Е.И., Мунерман В.И., Синицын И.Н. Современные технологические средства создания многомерно-матричных машин баз данных // Системы высокой доступности, 2024. Т. 17. С. 5-17.

50. Мунерман, В. И. Массовая обработка данных. Алгебраические модели и методы. – М.: ИНФРА-М, 2023. 229 с.

51. Свидетельство о регистрации программ для ЭВМ 2020613833 "Программа для обработки распределенных больших объемов данных для стоечных серверов и дата-центров".

52. Свидетельство о регистрации программ для ЭВМ 2020614270 "Программа для обработки распределенных больших объемов данных в вычислительных сетях рабочих станций".

53. Патент на полезную модель RUS 82355 12.08.2008/ Система представления данных в базе данных Сергеев В.П., Гайдаенко Т.И., Левин Н.А., Мунерман В.И., Оздемир С.М., Провоторова А.О., Ширай А.Е.

54. Патент № 2755568 Российская Федерация, МПК G06F 16/2455. Способ параллельного выполнения операции JOIN при обработке больших структурированных высокоактивных данных: №2020124733: заявл. 26.07.2020: опубл. 17.09.2021 / Мунерман В. И., Синявский Ю. В., Чукляев И. Л., Чукляев Е. И. – 10 с.