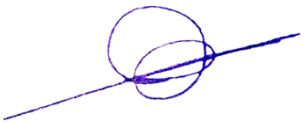
На правах рукописи



Алаасам Амир Басим Абдуламир

МОДЕЛИ, МЕТОДЫ И АЛГОРИТМЫ ОБРАБОТКИ ПОТОКОВ ДАННЫХ В ТУМАННЫХ ВЫЧИСЛИТЕЛЬНЫХ СРЕДАХ

Специальность 05.13.11 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

Автореферат
диссертации на соискание ученой степени
кандидата физико-математических наук

Челябинск — 2022

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

Научный руководитель: РАДЧЕНКО Глеб Игоревич
кандидат физико-математических наук, доцент,
ведущий научный сотрудник кафедры системного программирования ФГАОУ ВО «Южно-Уральский государственный университет (национальный исследовательский университет)» (г. Челябинск)

Официальные оппоненты: ИЛЬИН Вячеслав Анатольевич,
доктор физико-математических наук, профессор,
главный научный сотрудник ФГБУ Национальный исследовательский центр «Курчатовский институт» (г. Москва)

СУХОРОСЛОВ Олег Викторович,
кандидат технических наук,
старший научный сотрудник ФГБУН «Институт проблем передачи информации им. А.А. Харкевича Российской академии наук» (г. Москва)

Ведущая организация: ФГБОУ ВО «Московский государственный университет имени М.В. Ломоносова» (г. Москва)

Защита состоится «22» июня 2022 г. в 12:00 час. на заседании диссертационного совета Д 212.298.18 при Южно-Уральском государственном университете по адресу: 454080, г. Челябинск, пр. Ленина, 76, ауд. 1007.

С диссертацией можно ознакомиться в библиотеке Южно-Уральского государственного университета и на сайте: https://www.susu.ru/ru/dissertation/d-21229818/alaasam-amir-basim-abdulamir

Автореферат разослан «___» _____ 2022 г.

Ученый секретарь
диссертационного совета М.Л. Цымблер

Общая характеристика работы

Актуальность темы. Актуальность темы диссертационного исследования основывается на следующих основных факторах:

1. Экспоненциальный рост устройств и систем интернета вещей.
2. 4-я индустриальная революция и развитие технологий, связанных с Цифровыми двойниками.
3. Естественные ограничения применимости модели облачных вычислений, связанные с латентностью при обработке потоков данных.
4. Необходимость разработки и исследования новых подходов и моделей организации вычислительного процесса в гетерогенных распределенных вычислительных средах, обеспечивающих обработку потоков данных.

Рассмотрим эти факторы более подробно.

Технологии интернета вещей (Internet of Things – IoT) сегодня переживают стадию стремительного роста. По данным Allied Market Research, мировой рынок датчиков оценивался в $138 965 млн. в 2017 году и, по прогнозам, достигнет $287 002 млн. к 2025 году. Специалисты корпорации Ericsson подсчитали, что по состоянию на 2021 год, в мире насчитывается более 28 миллиардов подключенных к Интернету устройств, что составляет более 3 устройств на каждого жителя Земли. Из-за широкого распространения использования технологий IoT, в настоящее время физический и цифровой миры стали взаимосвязанными, что послужило мотивом для установления взаимосвязи между этими двумя мирами посредством телеметрии, поддерживаемой моделированием. Например, в автогонках Формулы-1 поток данных, собранный с сотен датчиков, установленных на автомобиле, и передаваемый на пульт технического обслуживания, служит источником данных для моделирования работы автомобиля в реальном времени. Используя эти модели, инженеры могут вносить корректировки в режим работы автомобиля удаленно, непосредственно в режиме гонки. Использование таких подходов в индустриальной сфере называется индустриальным интернетом вещей (Industrial Internet of Things – IIoT) целью которого является создание умной индустрии (Smart Industry) или Индустрии 4.0 (Industry 4.0), которая интегрирует IoT с производственными технологиями для создания взаимосвязанного производственного предприятия, которое анализирует информацию для осуществления интеллектуальных действий в физическом мире.

Важным приложением умной индустрии является так называемый цифровой двойник (Digital Twin – DT). DT представляет собой систему, состоящую из трех основных компонентов: физический объект в реальном мире, его виртуальное представление в виртуальном мире, а также потоки данных и управления,

которые объединяют реальные и виртуальные компоненты. В отличие от традиционного моделирования, виртуальное представление в DT постоянно обновляется с учетом состояния обслуживания и производительности на протяжении всего жизненного цикла физического объекта. Для создания DT технологических процессов и систем применяются системы математических моделей и методов, таких как интеллектуальный анализ данных, метод конечных элементов и др. Каждый из таких методов предъявляет особые требования к необходимым вычислительным ресурсам. Например, методы интеллектуального анализа данных требуют вычислительных ресурсов, предоставляющих существенные объемы хранения данных, в то время как модели, использующие метод конечных элементов, требуют высокопроизводительных вычислительных систем (или суперкомпьютеров).

Из-за необходимости сбора, передачи и анализа потоков данных от систем DT в режимах, близких к реальному времени, для настройки и актуализации их виртуального состояния, применение облачных технологий не позволяет обеспечить требуемые характеристики предоставляемых вычислительных ресурсов с точки зрения времени задержки и местоположению сервисов обработки данных. Возможным решением этой проблемы может служить применение модели туманных вычислений, которая расширяет концепцию облачных вычислений, предоставляя вычислительные ресурсы ближе к источникам данных. Туманные вычисления – это многоуровневая модель, обеспечивающая повсеместный доступ к общему континууму масштабируемых вычислительных ресурсов и поддерживающая развертывание распределенных приложений и сервисов с учетом латентности. Хотя промышленные данные часто являются неструктурированными, их можно уточнить и предварительно обработать локально на уровне туманных вычислений перед отправкой на облачный уровень для дальнейшей обработки. Концепция туманных вычислений позволяет перенести часть задач по обработке и хранению данных из облака на туманные узлы на границе сети для снижения задержки.

Для организации эффективной обработки данных в ограничениях, накладываемых, с одной стороны, особенностями предметной области IoT и DT, а с другой стороны, возможностями и особенностями архитектуры туманных вычислительных систем, может быть применен ряд существующих подходов. Событийно-управляемая архитектура (Event-Driven Architecture – EDA) наиболее адаптирована к этому типу приложений. EDA – это системная архитектура, состоящая из слабосвязанных, компонентов обработки событий, которые принимают и обрабатывают события одновременно. EDA по своей природе является экстремально слабосвязанной и высоко распределенной архитектурой программных систем. С другой стороны, для решения задачи обработки данных в DT применяется концепция научных потоков работ (Scientific Workflow – SWF).

Научным потоком работ называют набор взаимосвязанных вычислительных задач и задач по обработке данных, направленных на достижение конкретной цели, в частности на проведение вычислительного эксперимента. Тем не менее, сегодня можно выделить несколько ключевых проблем, связанных с использованием SWF для обработки потоков данных IoT и DT. Во-первых, SWF не ориентированы на обработку потоков данных. SWF исторически ориентированы на исполнение вычислительных задач в пакетном виде, где набор исходных данных собирается и подается в SWF в виде пакета, который и обрабатывается в рамках соответствующего потока работ. Кроме того, действия SWF могут генерировать большое количество промежуточных данных в течение жизненного цикла SWF. В такой сильносвязанной архитектуре, интенсивная передача данных между действиями SWF может вызвать значительное затруднение в процессе выполнения.

Анализ исследований, направленных на декомпозицию потоков работ на «под-потоки» показывает, что при решении этой задачи, авторы работ оставляют сильные связи между под-потоками; потоки работ реализуются в формате пакетной обработки данных, что не позволяет применить эти решения в контексте событийно-управляемой архитектуры для поддержки систем IoT в туманных вычислительных средах. Также большинство исследователей не фокусируются на потребностях туманных вычислений, которые включают необходимость географического распределения, а также необходимость слабосвязанной архитектуры не только между данными и обработкой, но и в самом слое обработки, где каждый вычислительный объект может быть реализован как независимый сервис. В связи этим, разработка моделей, методов и алгоритмов обработки потоков данных в туманных вычислительных средах является актуальной задачей.

Степень разработанности темы. Фундаментом современных подходов к организации предоставления распределенных вычислительных сервисов является идея коммунальных вычислений (Utility computing), предложенная Дж. Маккарти (John McCarthy) и Т. Курцем (Thomas Kurtz). Развитие данной концепции привело к появлению подхода метавычислений (metacomputing) позднее трансформировавшегося в концепцию грид-вычислений (Grid computing). Важный вклад в развитие этих подходов внесли такие ученые, как Л. Смарр (Larry Smarr), М. Мутка (Matt Mutka) и М. Ливны (Miron W. Livny), Ян Фостер (Ian Foster), А. Штрайт (Achim Streit), Д. Андерсон (David P. Anderson). Развитие систем виртуализации и контейнеризации, привело к формированию концепции облачных вычислений, которая стала стандартом дефакто в организации предоставления вычислительных ресурсов по требованию. Развитием данной концепции стала архитектура туманных вычислений, призванная решить задачу минимизации латентности, вместе с предоставлением унифицированного континуума вычислительных возможностей. Существенный вклад в развитие данного подхода внесли такие ученые, как П. Беллависта (Paolo Bellavista), А. Дэви (Alan Davy), М. Аазам (Mohammad Aazam), Ш. Зеадалли (Sherali Zeadally) и Х.А. Харрас (Khaled A. Harras).

Важным аспектом исследований в этой области является решение задачи управления вычислительными задачами, и обработкой данных в распределенных вычислительных системах. Модель научных потоков работ (Scientific Workflow) сегодня представляет собой основную модель, ориентированную на решение подобных задач. Важнейшие работы в области, связанной с проектированием, планированием и выполнением потоков работ сегодня выполняются группами ученых под руководством Е. Дильман (Ewa Deelman), Ц. Ванга (Jianwu Wang), Т. Фахрингера (Thomas Fahringer), Р. Сакелляриу (Rizos Sakellariou), И. Алтинтаса (Ilkay Altintas), П. Корамбатха (Prakashan Korambath), Б. Людешера (Bertram Ludäscher), А.Н. Черных.

В области обработки потоков данных можно отметить результаты работы научных групп под руководством таких ученых, как А. Сундерраджан (Abhinav Sunderrajan), А. Антони (Aleksandar Antoni), С. Триллес (Sergio Trilles), О. Карвальо (Otávio Carvalho), С. Хааг (Sebastian Haag), Д. Шейбмайр (Jim Scheibmeir).

Среди российских ученых существенный вклад в решение задач разработки моделей распределенных вычислительных систем, и обработки потоков данных был сделан в работах А.В. Бухановского, С.В. Ковальчука, Д.А. Насонова, О.В. Сухорослова, В.А. Ильина, Вл.В. Воеводина, и некоторых других.

Цель и задачи исследования. *Целью* исследования является разработка новой концепции организации потоков работ, включая математическую модель, методы и алгоритмы, позволяющей организовать эффективную обработку потоков данных в туманных вычислительных средах. Для достижения этой цели необходимо решить следующие задачи:

1) проанализировать известные концепции и принципы обработки потоковых данных, используемые для реализации приложений в туманных вычислительных средах;
2) разработать новую математическую модель организации потоков работ, ориентированную на эффективную обработку потоков данных в туманных вычислительных средах;
3) разработать алгоритм преобразования монолитных приложений потоков работ в наборы независимых потоков работ, поддерживающих поточную обработку данных;
4) разработать комплекс программных компонентов и утилит для поддержки обработки потоков данных в туманных вычислительных средах посредством потоков работ;
5) провести вычислительные эксперименты для оценки эффективности предложенной концепции и разработанного программного обеспечения.

Соответствие специальности. Основными задачами, решавшимися в диссертации, являются разработка новой концепции организации потоков работ, включая математическую модель, методы и алгоритмы, позволяющей организо-

вать эффективную обработку потоков данных в туманных вычислительных средах, что соответствует специальности 05.13.11 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей.

Методология и методы исследования. Методологической основой диссертационного исследования являются теория множеств и теория графов. При разработке программных компонентов применялись методы объектно-ориентированного проектирования и язык UML. Для программной реализации разработанных подходов были использованы методы объектно-ориентированного проектирования, язык Java, платформа контейнеризации приложений Docker, система управления потоками работ Kepler и платформа обработки потоков данных Apache Kafka.

Научная новизна. Новизна работы заключается в том, что разработана новая концепция организации потоков работ, получившая название «концепция микро-потоков работ», которая включает в себя модель, методы и алгоритмы, обеспечивающие эффективную обработку потоков данных в туманных вычислительных средах, и позволяет на порядки уменьшить время задержки получения результата при обработке потоков данных.

Теоретическая и практическая значимость. Теоретическая значимость работы заключается в том, что разработанная концепция микро-потоков работ включает в себя формальную модель организации обработки данных и алгоритм организации рефакторинга монолитных потоков работ в наборы слабосвязанных микро-потоков работ.

Практическая значимость работы состоит в разработке набора программных акторов и набора утилит, обеспечивающих интеграцию системы управления потоками работ Kepler и платформы обработки потоков данных Apache Kafka для реализации обработки потоков данных в виде микро-потоков

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

1. Разработана новая концепция микро-потоков работ, ориентированная на организацию обработки данных в туманных вычислительных средах, позволяющая значительно уменьшить время задержки получения результата при обработке потоков данных.
2. Разработан алгоритм рефакторинга потоков работ, позволяющий преобразовать монолитный поток в набор микро-потоков, допускающих параллельное выполнение.
3. Выполнены проектирование и реализация комплекса вычислительных акторов и программных утилит, обеспечивающих функционирование микро-потоков работ на базе платформы управления потоками работ Kepler и платформы обработки потоков данных Apache Kafka.

4. С помощью разработанного комплекса созданы микро-потоки работ, обеспечивающие поддержку типовых задач обработки данных, на базе которых проведены вычислительные эксперименты, подтверждающие эффективность предложенных подходов.

Степень достоверности результатов. Результаты исследования подтверждаются данными вычислительных экспериментов, выполненных в соответствии с общепринятыми стандартами.

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

1) RuSCDays 2018: Международная конференция «Суперкомпьютерные дни в России» (24-25 сентября 2018 г., Москва);
2) UCC'2018: 2018 IEEE/ACM International Conference on Utility and Cloud Computing (17-20 декабря 2018, Цюрих, Швейцария);
3) SIBIRCON'2019: 2019 International Multi-Conference on Engineering, Computer and Information Sciences (21-27 октября 2019 г., Екатеринбург);
4) ПаВТ'2022: Международная конференция «Параллельные вычислительные технологии 2022» (29-31 марта 2022 г., Дубна).

Публикации. По теме диссертации опубликовано 11 работ. Из них 5 работ опубликовано в журналах индексируемых в Scopus и Web of Science, и 6 работ – в журналах, включенных ВАК в перечень изданий, в которых должны быть опубликованы основные результаты диссертаций на соискание ученой степени доктора и кандидата наук.

Личный вклад автора. Все результаты, представленные в диссертационной работе, получены автором лично. Содержание диссертации и основные положения, выносимые на защиту, соответствуют персональному вкладу автора в работах, опубликованных в соавторстве. В работе [1] Г.И. Радченко принадлежит раздел 1 (введение, стр. 591–592), А.Н. Черных принадлежит раздел 2 (обзор текущего состояния исследований, стр. 592), А.Б.А. Алаасаму принадлежат все остальные результаты и разделы (стр. 592–600). В работе [2] Г.И. Радченко принадлежит раздел 1 (введение, в части описания концепции цифрового двойника и микросервисных систем, стр. 511–512), А.Н. Черных принадлежит раздел 3 (обзор работ в части описания архитектур обработки данных интернета вещей, стр. 516), Х.Л. Гонсалес-Компеану принадлежит раздел 1 (введение, в части описания туманных вычислений и систем поточной обработки данных, стр. 512–513), А.Б.А. Алаасаму принадлежат все остальные результаты и разделы (стр. 513–525). В работе [3] Г.И. Радченко принадлежит введение (стр. 82–83), А.Н. Черных принадлежит раздел 1.2 (обзор облачных и туманных вычислений

стр. 84), А.Б.А. Алаасаму принадлежат все остальные результаты и разделы (стр. 84–98). В работе [4] А.Н. Черных принадлежит введение (стр. 48–49), Г.И. Радченко принадлежит раздел 1 (обзор технологий виртуализации, стр. 49–51), А.Б.А. Алаасаму принадлежат все остальные результаты и разделы (стр. 51–79). В работе [5] Г.И. Радченко принадлежит раздел 1 (введение, в части описания концепции цифрового двойника и микросервисных систем, стр. 66), А.Н. Черных принадлежит раздел 3 (обзор литературы в части описания архитектур обработки данных интернета вещей, стр. 71), Х.Л. Гонсалес-Компеану принадлежит раздел 1 (введение, в части описания туманных вычислений и систем поточной обработки данных, стр. 67), А.Б.А. Алаасаму принадлежат все остальные результаты и разделы (стр. 68–70, 72–80). В работе [6] Г.И. Радченко принадлежит введение, разделы 2-3 (описание концепции цифрового двойника и облачной платформы для поддержки цифровых двойников, стр. 100–102, 103–105), А.Н. Черных принадлежит раздел 1 (обзор смежных работ, стр. 102–103), А.Б.А. Алаасаму принадлежат все остальные результаты и разделы (стр. 105–116). В работе [7] Г.И. Радченко принадлежат разделы 1, 3, 4 (введение, описание концепции цифрового двойника, описание облачной платформы цифровых двойников, стр. 83–85), А.Н. Черных принадлежит раздел 2 (обзор близких по тематике работ, стр. 84), А.Б.А. Алаасаму принадлежат разделы 5-8 (описание подхода микро-потоков работ, реализация и развертывание экспериментального исследования, оценка производительности микро-потоков работ, заключение, стр. 86–88). В работе [9] Г.И. Радченко принадлежит введение (стр. 0804), А.Н. Черных принадлежит раздел 2 (обзор близких по тематике работ стр. 0805), А.Б.А. Алаасаму принадлежат все остальные результаты и разделы (стр. 0805–0809). В работе [10] Г.И. Радченко принадлежит введение и часть обзора литературы, посвященная описанию общей концепции цифрового двойника (стр. 489), А.Н. Черных принадлежит заключение (стр. 493), К.В. Бородулину принадлежит часть обзора литературы, посвященная поточной обработке данных (стр. 490), А.А. Подкорытову принадлежит часть обзора литературы, посвященная системам обработки потоков работ (стр. 490), А.Б.А. Алаасаму принадлежат разделы описания концепции микро-потоков работ, реализация и тестирование предложенного подхода (стр. 490–493).

Структура и объем работы. Диссертация состоит из введения, четырех глав, заключения, библиографии и двух приложений. В приложении 1 приведены основные аббревиатуры, используемые в диссертации. В приложении 2 приведены основные обозначения, используемые в диссертации. Объем диссертации составляет 147 страниц, объем библиографии – 150 наименований.

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, «Обработка потоков данных в туманных вычислительных средах», рассматриваются понятия модели туманных вычислений и обработки потоков данных в туманных вычислительных средах в контексте систем индустриального интернета вещей и цифровых двойников. Обсуждаются ключевые подходы к организации архитектуры программных систем и методы обработки потоков данных с учетом и без учета состояния, инструменты потоковой обработки данных, а также особенности применения платформ управления научными потоками работ для решения таких задач. Особое внимание уделено обзору методов декомпозиции потоков работ и подходов к проектированию программных систем, ориентированных на обработку потоков данных в туманных средах.

Во второй главе, «Микро-потоки работ», представлена математическая модель микро-потоков работ для обработки потоков данных в распределенных вычислительных средах, таких как туманные вычислительные системы. Модель включает в себя алгоритм рефакторинга зависимостей в монолитном потоке в набор автономных микро-потоков работ. Такое разделение поддерживает независимость реализации, исполнения, разработки, сопровождения и кроссплатформенного развертывания микро-потоков работ на независимых вычислительных узлах.

Назовем монолитным SWF научный поток работ, спроектированный для обработки данных в пакетном режиме, и состоящий из множества сильно-связанных между собой вычислительных задач (см. рис. 1a). Предлагается новая концепция, основанная на декомпозиции SWF на наборы слабосвязанных меньших потоков работ, называемых микро-потоками работ (Micro-Workflow – MWF) (см. рис. 1b). Данное название образовано из объединения названий двух концепций: микросервисов и потоков работ. Поток работ может быть представлен в виде ориентированного ациклического графа W:

$$W = (V, E),$$

где V – это множество вершин, представляющих вычислительные задачи, и E – это множество направленных ребер, представляющих собой зависимости по данным между задачами. Выходная степень $deg^{+}(v_i)$ это количество ребер, исходящих из v_i к другим вершинам. Входная степень $deg^{-}(v_i)$ это количество ребер, направленных к v_i от других вершин. Ребро $(v_i, v_j) \in E$ представляет собой зависимость по данным от v_i к v_j. Пусть n это общее количество вершин в V.

Поток работ W разделяется на множество, состоящее из k подпотоков работ $S = (S_1, \dots S_k)$, $S_i = (V_i, E_i)$, когда V_i это множество вершин, входящих в подпоток работ S_i, и E_i – это множество ребер между вершинами, входящими в V_i, при этом множество вершин W разделяется подпотоками работ на набор непересекающихся подмножеств:

1. $\forall i \in 1..k: V_i \subset V, E_i \subset E;$
2. $\forall v \in V, \exists i \in 1..k : v \in V_i;$
3. $E_i = \{(v_k, v_l) \in E: v_k, v_l \in V_i\};$
4. $\forall i, j : i \neq j \implies V_i \cap V_j = \emptyset.$

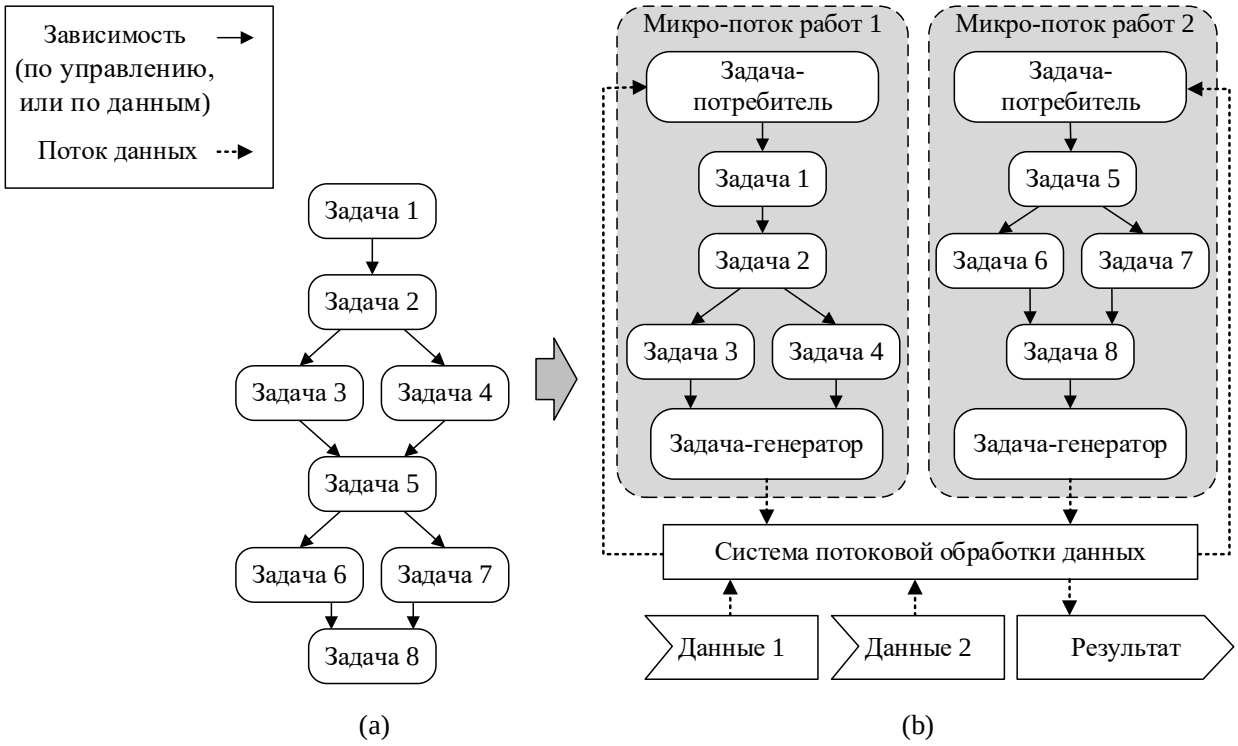


Рис. 1. Концепция микро-потоков работ: (a) монолитный поток работ, (b) микро-потоки работ в результате рефакторинга монолитного потока работ.

Определим следующие классы ребер и вершин, связанных с подпотоком работ S_i: EI_i – это набор входных ребер; EO_i – это набор выходных ребер; VI_i – это набор вершин в S_i, расположенных на головной части ребер EI_i, а также вершин, не имеющих входных ребер; VO_i – это набор вершин в S_i, расположенных на концах ребер EO_i, а также вершин, не имеющих выходных ребер:

$$EI_i = \{(v_k, v_l) \in E: v_k \notin S_i, v_l \in S_i\};$$
$$EO_i = \{(v_k, v_l) \in E: v_k \in S_i, v_l \notin S_i\};$$
$$VI_i = \{v_l \in S_i: (v_k, v_l) \in EI_i\} \cup \{v \in S_i: deg^-(v) = 0\};$$
$$VO_i = \{v_k \in S_i: (v_k, v_l) \in EO_i\} \cup \{v \in S_i: deg^+(v) = 0\}.$$

Чтобы преобразовать подпоток работ S_i в микропоток работ, нужно извлечь все вершины S_i из потока работ W и обеспечить коммуникационные механизмы, связывающие микро-поток работ с платформой потоковой передачи событий через выделенные «Вершину-потребитель» (cv_i) и «Вершину-генератор» (pv_i). Вершина cv_i обеспечивает потребление потока входных данных от платформы потоковой передачи данных и распределяет его между вершинами VI_i. Вершина pv_i собирает выходные данные из вершин, составляющих множество VO_i и передает их в виде сообщений на платформу потоковой передачи данных. Таким образом, микро-поток работ MWF_i может быть определен как ориентированный граф:

$$MWF_i = (MV_i, ME_i),$$

где:

– $ECV_i = \{(cv_i, v): v \in VI_i\}$, набор ребер, идущих от cv_i к вершинам в VI_i;

– $EPV_i = \{(v, cp_i) : v \in VO_i\}$, набор ребер, идущих от вершин в VO_i к cp_i.

– $MV_i = V_i \cup \{cv_i, pv_i\}$;

– $ME_i = E_i \cup ECV_i \cup EPV_i$.

В алгоритме рефакторинга монолитного потока работ в набор микро-потоков работ выделяются следующие ключевые шаги:

1. Инициализация матрицы Z для представления монолитного потока работ W и множества подпотоков S, на которые разбивается поток работ W.
2. Отображение в матрице Z внутренних и внешних ребер для подпотоков работ $S_x \in S$.
3. Создание матрицы M_x для представления каждого микро-потока работ MWF_x. Для этого, для каждого подпотока работ $S_x \in S$ выполняются следующие действия:
   - 3.1. Инициализация матрицы M_x.
   - 3.2. Отображение множества внутренних ребер подпотока E_x в соответствующую матрицу M_x.
   - 3.3. Формирование множества ребер, исходящих из вершины-потребителя cv_x и их отображение в M_x.
   - 3.4. Формирование множества ребер, входящих в вершину-генератор pv_x и их отображение в M_x.

Диагональные значения в матрице Z представляют собой индексы подпотоков работ S_i, в которые входят вершины v_i. Другие значения в матрице Z обозначают наличие ребра между вершиной v_i и вершиной v_j. Значение «-1» представляет собой ребро между различными под-потоками:

$$Z = \begin{bmatrix} z_{1,1} & \cdots & z_{1,n} \\ \vdots & \ddots & \vdots \\ z_{n,1} & \cdots & z_{n,n} \end{bmatrix}.$$

$$z_{i,j} = \begin{cases} 1, & i \neq j \ \wedge \ (v_i, v_j) \in E \\ 0, & i \neq j \ \wedge \ (v_i, v_j) \notin E, \\ x, & i = j \ \wedge \ v_i \in V_x \end{cases}$$

где $i, j \in 1..n$ – индексы элементов матрицы Z и $x \in 1..k$ – это индекс подпотока $S_x = (V_x, E_x)$.

$$z_{i,j} = \begin{cases} z_{i,i}, & i \neq j \ \wedge \ z_{i,j} = 1 \ \wedge \ z_{i,i} = z_{j,j} \\ -1, & i \neq j \ \wedge \ z_{i,j} = 1 \ \wedge \ z_{i,i} \neq z_{j,j}. \\ z_{i,j}, & \text{иначе} \end{cases}$$

На следующем шаге алгоритма происходит инициализация матриц M_x представления микро-потоков работ MWF_x на основе подпотоков S_x:

$$r_x = \left|\{a \in 1..n : z_{a,a} = x\}\right|.$$

$$M_x = \begin{bmatrix} m_{x_{1,1}} & \cdots & m_{x_{1,r_x+2}} \\ \vdots & \ddots & \vdots \\ m_{x_{r_x+2,1}} & \cdots & m_{x_{r_x+2,r_x+2}} \end{bmatrix}.$$

$$D_x = \{\forall i \in 1..n : z_{i,i} = x\}.$$

$$m_{x_{a,a}} = \begin{cases} d_{a-1}, & 1 < \ a < r_x + 2 \\ x, & a = 1 \vee a = r_x + 2 \end{cases},$$

где $x \in 1..k$ – это индекс подпотока S_x и $a \in 1..r_x + 2$ – это индекс строки в матрице M_x.

Диагональные значения в матрице M первой и последней строки будут равны x, где x – это индекс подпотока. Остальные диагональные элементы матрицы M принимают значение индексов строк матрицы Z, диагональные значения которых равны индексу подпотока. Все внутренние ребра S_x будут представлены в M_x следующим образом:

$$m_{x_{a,b}} = \begin{cases} 1, & z_{m_{x_{a,a}}, m_{x_{b,b}}} = m_{x_{1,1}} \\ 0, & \text{иначе} \end{cases},$$

$$\forall m_{x_{a,b}} \in M_x : a, b \in 2..r_x + 1 \wedge a \neq b.$$

Существует два типа вершин в M_x, которые должны получать входящее ребра от вершины cv_x: начальная вершина, не имеющая входящих ребер, и вершины, которые получают входящее ребра от вершин, из других подпотоков:

$$m_{x_{1,b}} = \begin{cases} 1, \begin{pmatrix} (\forall i \in 1..n \ \wedge i \neq m_{x_{b,b}} : z_{i, m_{x_{b,b}}} = 0) \\ \vee \ (\exists i \in 1..n : z_{i, m_{x_{b,b}}} = -1) \end{pmatrix}, \\ 0, \text{иначе} \end{cases}$$

$$\forall m_{x_{1,b}} \in M_x : b \in 2..r_x + 1.$$

Также существует два типа вершин, которые должны быть связаны ребрами с вершиной pv_x: вершина, не имеющая исходящих ребер, и вершины, исходящие ребра из которых идут к вершинам в других подпотоках:

$$m_{x_{a,r_x+2}} = \begin{cases} 1, \begin{pmatrix} (\forall b \in 2..r_x + 1 \wedge a \neq b: m_{x_{a,b}} = 0) \\ \vee \ (\exists j \in 1..n: z_{m_{x_{a,a'},j}} = -1) \end{pmatrix}, \\ 0, \text{иначе} \end{cases}$$
$$\forall m_{x_{a,r_x+2}} \in M_x: a \in 2..r_x + 1.$$

В третьей главе, «Программная поддержка модели микро-потоков работ», представлено описание программных компонентов, реализованных для поддержки реализации и тестирования модели микро-потоков работ. На языке Java был разработан набор акторов, расширяющих возможности системы Kepler для обеспечения потоковой обработки данных с использованием платформы Apache Kafka. На их основе был разработан набор потоков работ для решения типовой задачи мониторинга состояния оборудования химического производства. В качестве исходных данных используется реальный набор данных, предоставленный DEBS GrandChallenge. Для проведения вычислительных экспериментов, разработан ряд программных утилит, поддерживающих симуляцию IoT потоков данных, репликацию данных между серверами Apache Kafka, а также рефакторинг монолитных потоков работ.

Актор ***KafkaConsumer*** представляет собой реализацию вершины-потребителя микро-потока работ. Актор KafkaConsumer подписывается на тему Kafka, подключаясь к серверу Apache Kafka по протоколу TCP, и получает поток данных (в виде последовательности сообщений) из этой темы. Актор обеспечивает десериализацию каждого полученного сообщения в набор токенов (тип данных в системе Kepler), которые затем отправляются на выходной порт. Актор ***KafkaProducer*** представляет собой реализацию вершины-генератора микро-потока работ. Данный актор получает токены Kepler в качестве входных данных на входной порт, сериализует каждую полученную запись токенов в виде сообщения в формате Apache Avro, а затем отправляет это сообщение соответствующей теме Apache Kafka на сервере Kafka по протоколу TCP. Актор ***DetectStateChange разработан*** для постоянного мониторинга изменений в последовательностях данных, получаемых от отдельного источника данных или от набора источников данных. Актор ***CorrelateStateChange*** решает задачу обнаружения зависимостей между изменениями, которые произошли в одной последовательности данных с изменениями, которые произошли в другой последовательности данных путем анализа корреляции между ними. Актор ***XYState*** обеспечивает подготовку значений на основе входящего потока данных, в формате, требуемом XYPlotter (стандартный актор в Kepler). Результатом работы актора является формирование пар значений (X, Y) в выходном порту. С использованием данных акторов

на базе платформы Kepler был разработан набор потоков работ, решающих типовую задачу обработки потоков данных интернета вещей, как в формате монолитного потока работ, так и в формате отдельных микро-потоков работ (см. рис. 2, рис. 3, и рис. 4).

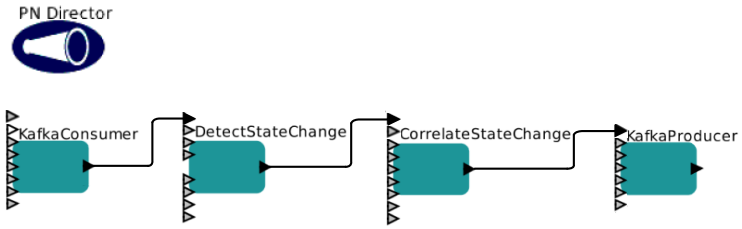


Рис. 2. Идентификация изменения состояния и корреляция изменения состояния в одном потоке работ.

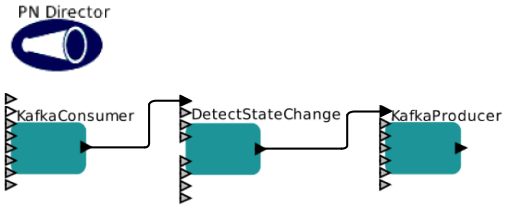


Рис. 3. Этап идентификации изменения состояния в отдельном микро-потоке работ.

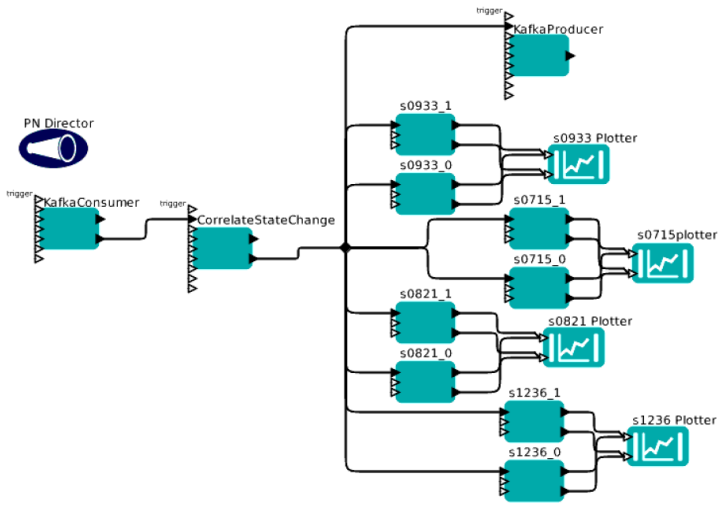


Рис. 4. Этап корреляции изменения состояния и построение результатов в отдельном микро-потоке работ.

Для организации синхронизации данных между географически распределенными центрами обработки данных в туманных средах был разработан механизм репликации данных между кластерами Apache Kafka на основе концепции микро-потоков работ (см. рис. 5).

Для реализации и испытания алгоритма рефакторинга монолитного потока работ на набор независимых микро-потоков работ, было разработано приложение. Листинг 1 представляет собой псевдокод разработанной утилиты. Основные исходные коды третьей главы свободно доступны в сети Интернет по следующему адресу: https://github.com/alaasamameer/Micro-Workflows.

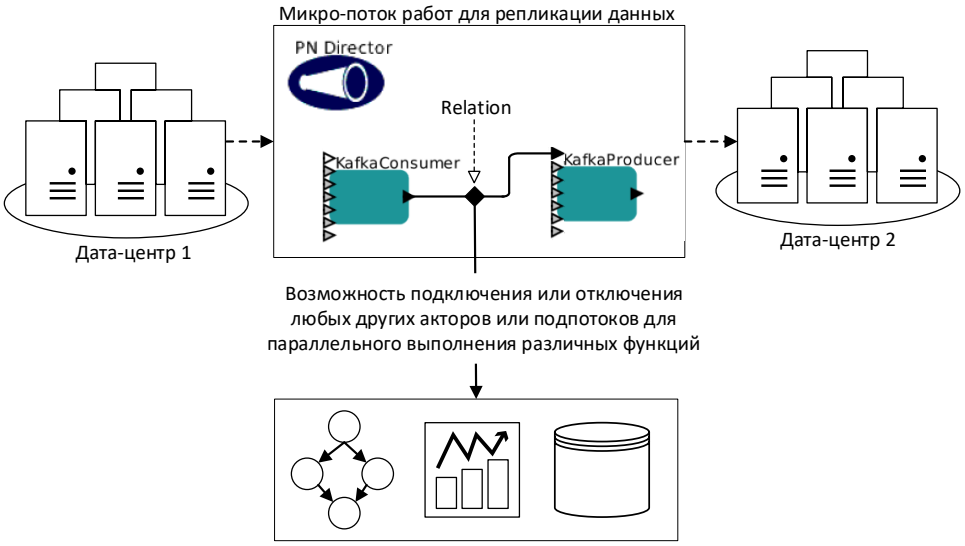


Рис. 5. Микро-поток работ для репликации данных.

Листинг 1. Рефакторинг микро-потоков работ

```
 1: procedure Refactor
 2: wf, subWfs ← ReadWorkflow(PARAMS.SOURCE)
 3: Z ←InitializeZ(wf, subWfs)
 4: Z ←IdentifyInternalExternalEdges(Z)
 5: for i from 1 to subWfs.Length():
 6:     MWF[i]←InitializeM(subWfs[i],Z)
 7:     MWF[i]←InternalEdges(MWF[i],Z)
 8:     MWF[i]←AddCv(MWF[i],Z)
 9:     MWF[i]←AddPv(MWF[i],Z)
10: WriteMWFs(MWF)
11: end procedure
```

В четвертой главе, «Вычислительные эксперименты», представлены результаты тестирования модели микро-потоков работ. Ключевой задачей вычислительных экспериментов является оценка применимости модели и алгоритмов микро-потоков работ для организации обработки потоков данных, в том числе данных IoT устройств, в распределенных вычислительных средах. Критериями для оценки эффективности данной модели выбраны метрики, отображающие среднюю времени ответа, объем данных, передаваемых через глобальные сети для обеспечения обработки данных, среднее время доставки сообщений.

В эксперименте по рефакторингу монолитного потока работ был протестирован алгоритм, обеспечивающий преобразование монолитного потока работ в набор микро-потоков работ. На рис. 6 приведена визуализация результата данного эксперимента. Алгоритм рефакторинга был выполнен над потоком работ «Montage 25» для выделения двух микро-потоков работ.

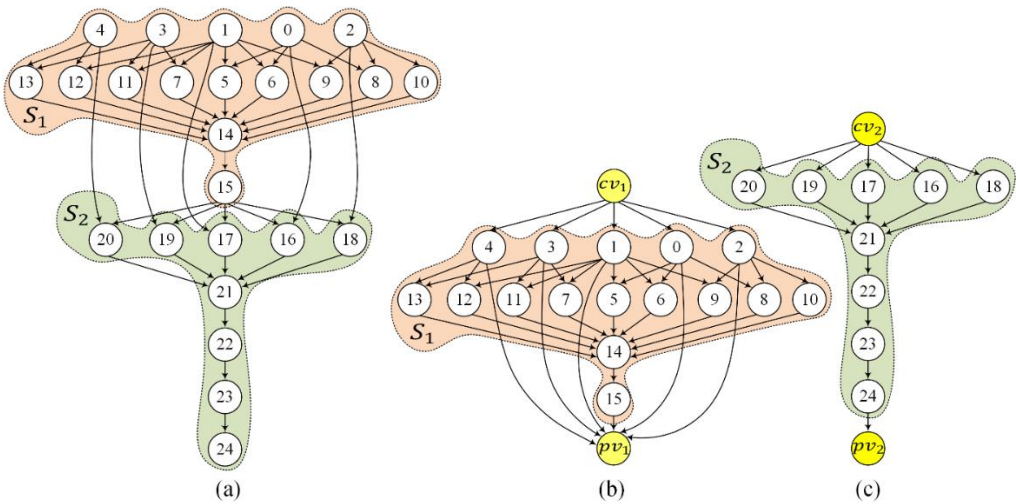


Рис. 6. Рефакторинг «Montage 25» на два микро-потока работ.

Второй эксперимент показывает преимущество разработанной концепции микро-потоков работ по сравнению с существующим подходом к организации вычислительного процесса в виде монолитного потока работ.

Табл. 1. Результаты испытания.

	Время ответа	**Объем входных данных**	**Объем результирующих данных**
Монолитный поток работ	670 732 мс	27.4 Мбайт	46 Кбайт
Микро-поток работ	1.4 мс	1 Кбайт	1 Кбайт

Результаты позволяют сделать следующие выводы:

1. При организации обработки потока данных посредством микро-потока работ среднее время ответа оказалось значительно меньше (1.4 мс в табл. 1), чем время ответа, обеспечиваемое монолитным потоком работ (11.1 мин. в табл. 1). Это достигается за счет того, что микро-поток работ получает данные и обрабатывает данные в потоковом режиме, обеспечивая предоставление результатов, как только они становятся доступными, в то время как монолитный поток ждет, пока завершится этап подготовки данных для запуска их обработки.
2. Среднее время ответа на события при обработке в формате микро-потока работ не зависит от общего периода выполнения, в то время как при реализации обработки данных в пакетном режиме монолитного потока работ, среднее время предоставления ответов возрастает при увеличении периода выполнения в связи с увеличением периода подготовки данных и увеличением размера входных данных.
3. Объем данных, необходимых для инициализации вычислительного процесса в формате микро-потока работ, очень мал по сравнению с монолитным потоком работ (1 Кбайт против 27.4 Мбайт в табл. 1). Аналогично,

средний размер сообщений с результатами обработки данных микро-потока работ также меньше по сравнению с монолитным потоком работ (1 Кбайт против 46 Кбайт в табл. 1). Это показывает, что микро-поток работ более подходит для развертывания в условиях ограниченных сетевых и вычислительных ресурсов, в том числе на узлах туманных вычислительных систем.

В эксперименте по локальному и распределенному развертыванию микропотоков работ оценивается объем накладных расходов, возникающих при распределенном развертывании микро-потоков работ. В первом варианте развертывания два микро-потока работ MWF_1 (см. рис. 3), MWF_2 (см. рис. 4) и симулятор датчика были развернуты на одном узле, оснащенным двухъядерным процессором Intel Core i7-4600U 2,1 ГГц с 8 ГБ оперативной памяти. Во втором варианте развертывания MWF_1, MWF_2 и симулятор датчика были развернуты на отдельных вычислительных узлах $VM1$, $VM2$, и $VM3$ в рамках одной локальной вычислительной сети. $VM1$ (включают симулятор датчика) и $VM3$ (включают MWF_2) находятся на одном физическом узле (4 ГБ оперативной памяти и 4 ядра процессора Intel Xeon X5680). $VM2$ (включают MWF_1) выполняется на отдельном физическом узле (12 ГБ оперативной памяти и 8 ядер процессора Intel Xeon X5680). По результатам проведенных испытаний были рассчитаны значения следующих метрик:

1. **Av_SM** (средний интервал между исходными сообщениями): средняя задержка по времени между двумя последовательными исходными сообщениями с данными, переданными симулятором датчиков на сервер Kafka.
2. **Av_TAT** (среднее время обработки): средний временной интервал, требуемый для создания одного итогового сообщения микро-потоком работ. Данная метрика позволяет оценить, какое время требуется системе обработки данных, состоящей из промежуточного ПО потоковой обработки данных и сервисов микро-потоков работ на обработку данных и генерацию результирующего сообщения.
3. **Av_L12** (средняя задержка): средняя сетевая задержка между узлами, на которых развернуты микро-потоки работ MWF_1 и MWF_2.

Анализ результатов эксперимента (см. табл. 2) показывает, что вычислительные процессы отдельных микро-потоков работ могут быть распределены по узлам вычислительной сети. При таком распределении время получения ответа увеличивается, по крайней мере, на величину задержки между узлами. Однако, результаты испытания 2 позволяют сделать вывод, что даже в этом случае время генерации ответа остается меньше скорости генерации данных с датчиков.

В рамках эксперимента по обработке данных с сохранением состояния средствами микропотоков работ обеспечивается анализ времени отклика при реализации в микро-потоках работ обработки данных с сохранением состояния.

В процессе обработки данных был интегрирован механизм синхронизации локального состояния путем реализации механизма поддержки вычислений с сохранением состояния в Kafka. Результаты эксперимента показывают, что внедрение обработки с сохранением состояния увеличивает среднюю метрику задержки обработки с 1.3 мс в режиме работы без сохранения состояния до 70.7 мс в режиме сохранения состояния.

Табл. 2. Сопоставление значений метрик, полученных при проведении эксперимента.

Метрики	**Испытание 1 (локальное развертывание микро-потоков работ)**	**Испытание 2 (распределенное развертывание микро-потоков работ)**
Время тестирования	24 часа	24 часа
Количество обработанных сообщений	9 180 056	7 328 844
Av_SM	9.4 мс	11.8 мс
Av_TAT	1.3 мс	3.2 мс
AV_L12	—	3.5 мс

В рамках эксперимента по живой миграции микро-потока работ проводится испытание по оценке возможности восстановления вычислительного процесса микро-потока работ, реализующего процесс обработки данных с сохранением состояния, после остановки или выхода из строя его контейнера. Кроме того, проверяется возможность использования функции синхронизации локального хранилища состояния с промежуточными темами Apache Kafka в качестве основы для переноса вычислительной задачи в новый контейнер, сохраняя при этом непрерывность результатов из предыдущей точки остановки.

Тест начинается с инициализации двух контейнеров (ksA1, ksB) одним и тем же микро-потоком работ. Каждый из запущенных микро-потоков работ получает уникальный идентификатор приложения и различные темы вывода в системе Apache Kafka (см. точки 1 и 2 на рис. 7). После этого производится запуск симулятора датчиков IoT, который генерирует поток данных в течение одного часа. Для симуляции экстренной остановки процесса обработки данных, через 20 минут после начала эксперимента завершается работа контейнера ksA1 (см. точку 3 на рис. 7), и запускается новый контейнер (ksA2) с тем же микро-потоком работ, той же темой ввода и вывода, и тем же идентификатором приложения, что и ksA1 (см. точку 4 на рис. 7). Таким образом, обеспечивается возможность ksA2 повторно использовать предыдущие промежуточные темы ksA1 для передачи состояния ksA1 в ksA2. Через 1 час генерация данных прекращается, и производится оценка результатов проведенного эксперимента: обработка данных (в случае ksA1 и ksA2) полностью соответствует непрерывной работе в случае штатного выполнения (в случае ksB).

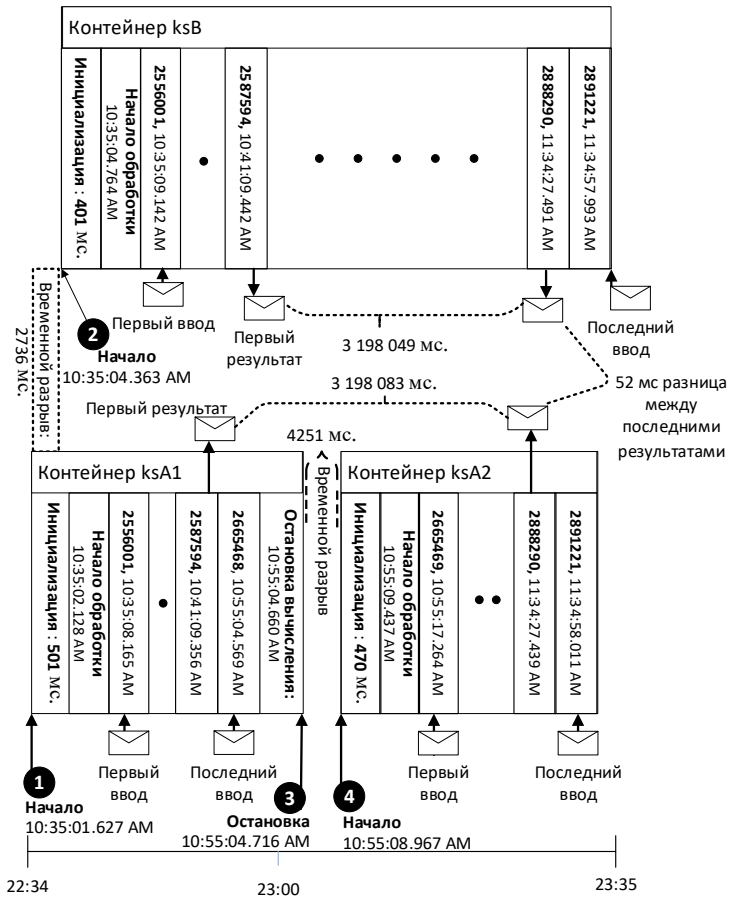


Рис. 7. Временная диаграмма проведения эксперимента по живой миграции.

В рамках эксперимента по распределению вычислительной нагрузки в модели туманной вычислительной среды исследуется возможность развертывания микро-потоков работ на базе географически распределенных гетерогенных вычислительных сред. В рамках эксперимента моделируется развертывание части микро-потоков работ на географически удаленных вычислительных узлах (например, на базе публичного облака), в то время как другая часть микро-потоков работ выполняется на узлах рядом с источником данных, чтобы обеспечить минимальное время ответа. В этом эксперименте сопоставляется два варианта обработки потока данных. Испытание 1 представляет собой обработку данных в виде единого микро-потока работ (ks_total), развернутого в частном облаке ЮУрГУ. Испытание 2 представляет собой обработку этого же потока данных посредством двух микро-потоков работ, один из которых (ks1_susu) размещен частном облаке ЮУрГУ, а второй (ks2_yandex) размещен в географически удаленном вычислительном центре – публичном облаке компании Яндекс. На рис. 8 представлена схема организации эксперимента.

Эксперимент состоит в организации обработки исходного потока данных в течение 6 часов. Как в рамках испытания 1, так и в рамках испытания 2, за это время было обработано 1 638 104 сообщения потока работ, что привело к гене-

рации 1 616 результирующих сообщений. Сравнение значений результатов обработки данных, полученных ks1_susu и ks2_yandex с результатами соответствующих частей в ks_total показало 100% соответствие результатов обработки данных с точки зрения полученных значений. Результаты эксперимента показывают, что микро-потоки работ могут быть успешно развернуты в географически-распределенных вычислительных средах. Более того, среднее время предоставления ответа при распределенном развертывании не увеличилось. Также стоит отметить тот факт, что в результате предварительной обработки, реализуемой в рамках «ks1_susu», за время испытания 2 было сформировано 1 616 сообщений, что составляет менее 0.1% от начального объема потока данных, составляющего 1 638 104 сообщений. Учитывая, что размер сообщений, формируемых «ks1_susu» соответствует размеру входящих сообщений, размещение «ks1_susu» на узлах, расположенных в непосредственной близости к источнику данных, позволило уменьшить объем данных, передаваемых в «ks2_yandex» по глобальной сети, в 1 014 раз по сравнению с вариантом, если бы вся обработка данных велась на узлах, размещенных в публичном облаке.

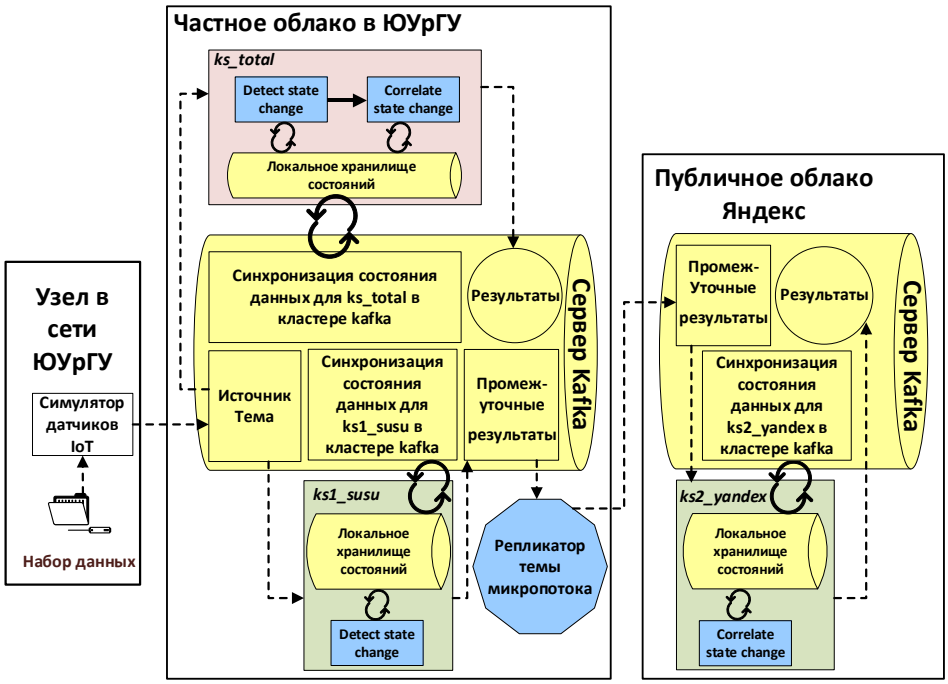


Рис. 8. Схема организации эксперимента по географическому распределению вычислительной нагрузки.

В заключении в краткой форме излагаются итоги выполненного диссертационного исследования, представляются отличия диссертационной работы от ранее выполненных родственных работ других авторов, даются рекомендации по использованию полученных результатов и рассматриваются перспективы дальнейшего развития темы.

Заключение

В диссертационной работе были рассмотрены вопросы разработки и исследования моделей обработки потоков данных, обеспечивающих поддержку для реализации приложений в туманных вычислительных средах. Основным научным результатом является создание концепции микро-потоков работ, позволяющей организовать эффективную обработку потоков данных в туманных вычислительных средах с применением потоков работ. Разработан алгоритм рефакторинга монолитных приложений потоков работ в наборы микро-потоков работ. Разработан на языке Java комплекс вычислительных акторов и программных утилит для поддержки функционирования микро-потоков работ на базе платформы управления потоками работ Kepler и платформы обработки потоков данных Apache Kafka. С использованием реализованных акторов и программных утилит были проведены вычислительные эксперименты для верификации предложенной модели и разработанного программного обеспечения.

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. Отметим основные отличия:

1) областью применения подели микро-потоков работ является интеграция поддержки поточной обработки данных в приложения потоков работ; ни одна из известных моделей декомпозиции потоков работ на под-потоки работ не ориентирована на реализацию такого формата обработки данных;
2) модель микро-потоков работ позволяет на порядки уменьшить время задержки получения результата при обработке потоков данных по сравнению с пакетными подходами обработки данных;
3) ни одна из рассмотренных работ не включала в себя алгоритм декомпозиции потока работ на набор независимых микро-потоков работ, взаимодействие между которыми осуществляется на основе событийно-ориентированной модели;
4) в отличие от типовых решений обработки потоков данных, применение концепции микро-потоков работ позволяет интегрировать в туманные вычислительные среды существующие вычислительные процессы, реализованные на основе потоков работ;
5) модель микро-потоков работ не применима в случае организации обработки больших объемов данных в пакетном режиме в связи с накладными расходами, связанными с необходимостью постоянного выполнения вычислительных сервисов микро-потоков работ на узлах вычислительной системы, а также поддержкой обмена данными с сохранением состояния.

Разработанная в ходе настоящего диссертационного исследования модель микро-потоков работ и алгоритм рефакторинга потоков работ на независимые микро-потоки работ могут применяться для организации вычислительного процесса обработки потоков данных в туманных вычислительных средах. Разработанный комплекс вычислительных акторов и программных утилит для поддержки функционирования микро-потоков работ может быть использован для создания микро-потоков работ на базе системы Kepler и платформы обработки потоков данных Apache Kafka.

В качестве направления дальнейших исследований можно выделить разработку методов автоматизированного планирования и масштабирования микро-потоков работ в туманных вычислительных средах.

Публикации автора по теме диссертации

Публикации в журналах из списка ВАК

1. Alaasam, A.B.A. Refactoring the Monolith Workflow into Independent Micro-Workflows to Support Stream Processing / A.B.A. Alaasam, G. Radchenko, A. Tchernykh // Programming and Computer Software. –2021. –Vol. 47, No. 8. –P. 591–600. DOI: 10.1134/S0361768821080077. (*Индексируется в Web of Science и Scopus*)
2. Alaasam, A.B.A. Analytic Study of Containerizing Stateful Stream Processing as Microservice to Support Digital Twins in Fog Computing / A.B.A. Alaasam, G. Radchenko, A. Tchernykh, J. L. González Compeán // Programming and Computer Software. –2020. –Vol. 46, No. 8. –P. 511–525. DOI:10.1134/S0361768820080083. *(Индексируется в Web of Science и Scopus*)
3. Alaasam, A.B.A. Micro-Workflows Data Stream Processing Model for Industrial Internet of Things / A.B.A. Alaasam, G. Radchenko, A. Tchernykh // Supercomputing Frontiers and Innovations. –2021. –Vol. 8, No. 1. –P. 82–98. DOI:10.14529/jsfi210106. *(Индексируется в Scopus*)
4. Radchenko, G. Comparative Analysis of Virtualization Methods in Big Data Processing / G. Radchenko, A.B.A. Alaasam, A. Tchernykh // Supercomputing Frontiers and Innovations. –2019. –Vol. 6, No. 1. –P. 48–79. DOI:10.14529/jsfi190107. *(Индексируется в Scopus*)
5. Алаасам, А.Б.А. Цифровые двойники в туманных вычислениях: организация обработки данных с сохранением состояния на базе микропотоков работ / А.Б.А. Алаасам, Г. И. Радченко, А. Н. Черных, Х.Л. Гонсалес-Компеан // Труды Института системного программирования РАН. –2021. –Т. 33, № 1. –С. 65–80. DOI:10.15514/ISPRAS-2021-33(1)-5.

6. Алаасам, А.Б.А. Микро-потоки работ: сочетание потоков работ и потоковой обработки данных для поддержки цифровых двойников технологических процессов / А.Б.А. Алаасам, Г. И. Радченко, А. Н. Черных // Вестник ЮУрГУ. Серия Вычислительная математика и информатика. –2019. –Т. 8, № 4. –С. 100–116. DOI:10.14529/cmse190407.

Публикация, индексируемая в Web of Science и Scopus

7. Radchenko, G. Micro-Workflows: Kafka and Kepler Fusion to Support Digital Twins of Industrial Processes / G. Radchenko, A.B.A. Alaasam, A. Tchernykh // 2018 IEEE/ACM International Conference on Utility and Cloud Computing Companion (UCC Companion). –Zurich, Switzerland: IEEE, 2018. No. 18. –P. 83–88. DOI:10.1109/UCC-Companion.2018.00039.

Публикации, индексируемые в Scopus

8. Alaasam, A.B.A. The Challenges and Prerequisites of Data Stream Processing in Fog Environment for Digital Twin in Smart Industry / A.B.A. Alaasam // International Journal of Interactive Mobile Technologies. –2021. –Vol. 15, No. 15. –P. 126–139. DOI:10.3991/ijim.v15i15.24181.

9. Alaasam, A.B.A. Stateful Stream Processing for Digital Twins: Micro-service Based Kafka Stream DSL / A.B.A. Alaasam, G. Radchenko, A. Tchernykh // 2019 International Multi-Conference on Engineering, Computer and Information Sciences (SIBIRCON). –IEEE, 2019. –P. 0804–0809. DOI:10.1109/SIBIRCON48586.2019.8958367.

Публикация, индексируемая в РИНЦ

10. Alaasam, A.B.A. Scientific Micro-Workflows: Where Event-Driven Approach Meets Workflows to Support Digital Twins / A.B.A. Alaasam, G. Radchenko, A. Tchernykh, K. Borodulin, A. Podkorytov // Суперкомпьютерные дни в Росси: труды международной конференции. –2018. –С. 489–495.

Свидетельство о регистрации программ для ЭВМ

11. Алаасам, А.Б.А., Радченко Г.И. Свидетельство Роспатента о государственной регистрации программы для ЭВМ «Комплекс акторов для поддержки концепции Micro-Workflow на платформе Kepler» № 2021661464 от 12.07.2021.

Работа выполнена при финансовой поддержке РФФИ в рамках научных проектов № 19-37-90073 и № 18-07-01224, а также при поддержке Министерства науки и высшего образования РФ в рамках государственного задания № FENU-2020-0022.

Формат 60 × 84 1/16. Бумага офсетная. Печать офсетная. Авт. л. 1,0. Уч.-изд. л. 1,0.
Подписано в печать __.__.2022. Тираж 100 экз.
Издательский центр ЮУрГУ. 454080, г. Челябинск, пр. Ленина, 76