МНОГОПРОЦЕССОРНЫЕ ВЫЧИСЛИТЕЛЬНЫЕ И УПРАВЛЯЮЩИЕ СИСТЕМЫ

МВУС-2022

Конференция посвящена 100-летию со дня рождения выдающегося российского ученого, Героя социалистического труда, академика РАН А.В. Каляева и развитию его научных идей

29.06.1922 - 10.03.2004

Материалы Всероссийской научно-технической конференции

27 - 30 июня 2022 г.
Таганрог, Ростовская область

Министерство науки и высшего образования Российской Федерации
Российская академия наук
Южный научный центр Российской академии наук
Южный федеральный университет
НИИ многопроцессорных вычислительных систем имени академика А.В. Каляева Южного федерального университета
Институт компьютерных технологий и информационной безопасности Южного федерального университета
ООО «НИЦ супер-ЭВМ и нейрокомпьютеров»
ООО «НИИ МВУС»

МНОГОПРОЦЕССОРНЫЕ ВЫЧИСЛИТЕЛЬНЫЕ И УПРАВЛЯЮЩИЕ СИСТЕМЫ

Материалы Всероссийской научно-технической конференции

Таганрог, Ростовская область, 27–30 июня 2022 г.

Ростов-на-Дону – Таганрог
Издательство Южного федерального университета
2022

УДК 004.4:681.3(063)
ББК 32.97+32.973.26 я431
М73

М73 Многопроцессорные вычислительные и управляющие системы (МВУС-2022) : Материалы Всероссийской научно-технической конференции (Таганрог, 27–30 июня 2022 г.) ; Южный федеральный университет. – Ростов-на-Дону ; Таганрог : Издательство Южного федерального университета, 2022. – 252 с.
ISBN 978-5-9275-4144-7

В материалах Всероссийской научно-технической конференции «Многопроцессорные вычислительные и управляющие системы» (МВУС-2022), посвященной 100-летию со дня рождения выдающегося российского ученого, Героя социалистического труда, академика РАН А.В. Каляева и развитию его научных идей, представлены доклады, посвященные вопросам построения архитектуры, разработке математического и программного обеспечения МВУС, реконфигурируемым вычислительным системам, проблемно-ориентированным вычислительным и управляющим системам, а также интеллектуальным и робототехническим системам.

Публикуется в авторской редакции.

ISBN 978-5-9275-4144-7

УДК 004.4:681.3(063)
ББК
32.97+32.973.26 я431

© Авторы докладов, 2022
© Научно-исследовательский институт многопроцессорных вычислительных систем им. академика А.В. Каляева, составление, оформление, 2022

ПРЕДИСЛОВИЕ

Уважаемые коллеги!

Всероссийская научно-техническая конференция «Многопроцессорные вычислительные и управляющие системы» (МВУС-2022) проводится с целью представления достижений российских ученых в фундаментальных исследованиях и прикладных разработках в области многопроцессорных вычислительных и управляющих систем и посвящена 100-летию со дня рождения выдающегося российского ученого, Героя социалистического труда, академика РАН Анатолия Васильевича Каляева.

Академик А.В. Каляев (1922–2004 гг.) – выдающийся ученый в области многопроцессорных вычислительных и управляющих систем, организатор, научный руководитель и признанный лидер ведущей научной школы, пользующейся известностью и авторитетом в широких кругах ученых Российской Федерации и за рубежом.

А.В. Каляев пятьдесят лет, начиная с 1954 года, работал в Таганрогском радиотехническом институте (позднее Таганрогском радиотехническом университете) вначале в качестве доцента, затем профессора, заведующего кафедрой вычислительной техники, проректора. Восемнадцать лет с 1968 по 1986 гг. А.В. Каляев являлся ректором ТРТИ. В этот период по его инициативе и под его руководством был сформирован Учебно-научно-производственный комплекс Таганрогского радиотехнического института, который завоевал высокий авторитет среди вузов и научных организаций в СССР и на международной арене. А.В. Каляевым были основаны и организованы Особое конструкторское бюро «МИУС», научным руководителем которого он был 15 лет, и НИИ многопроцессорных вычислительных систем, директором которого он являлся более 20 лет. В ОКБ «МИУС» и НИИ МВС под научным руководством А.В. Каляева были выполнены десятки фундаментальных и прикладных научных исследований и опытно-конструкторских работ.

Основными направлениями научных исследований А.В. Каляева являлись суперкомпьютеры с массовым параллелизмом, нейрокомпьютеры и нейропроцессорные системы. В этих научных направлениях А.В. Каляев опубликовал более 380 научных работ, включая 14 моно-

графий, 80 изобретений и более 280 статей, в том числе десятки статей в зарубежных научных изданиях.

Научные результаты, полученные А.В. Каляевым, широко использовались на предприятиях радиотехнической, электронной, авиационной и машиностроительной промышленности.

Под руководством А.В. Каляева были разработаны и созданы более 20 параллельных многопроцессорных вычислительных систем и параллельных нейрокомпьютеров, которые были внедрены у заказчиков. А.В. Каляевым и его учениками еще в 1964 году была создана (одна из первых в мире) многопроцессорная вычислительная система цифровая интегрирующая машина «Метеор-3» на 100 параллельно работающих процессоров с очень высокой для того времени производительностью.

Под научным руководством А.В. Каляева выполнен ряд крупных комплексных научно-технических программ, в которых, наряду с ОКБ «МИУС» и НИИ МВС, принимали участие десятки крупных организаций: вузы, научные организации, научно-производственные объединения и предприятия радиотехнической и электронной промышленности. В рамках этих программ были разработаны, созданы и поставлены заказчикам несколько типов проблемно-ориентированных многопроцессорных суперкомпьютеров с массовым параллелизмом и сверхвысокой производительностью, в том числе многопроцессорные вычислительные системы «Трасса», «Модуль-8», ЕС-2703, «Символ-2», «Символ-4» и другие.

Под руководством А.В. Каляева разработаны и серийно выпускались предприятиями электронной и радиотехнической промышленности два поколения микропроцессорных комплектов с программируемой структурой для проблемно-ориентированных суперкомпьютеров с массовым параллелизмом.

А.В. Каляев многократно выступал с научными докладами, в том числе и с заказными, на международных научных конгрессах, конференциях и симпозиумах в США, Англии, Германии, Италии, Ирландии, Японии, Южной Корее и во многих других странах. Научные достижения А.В. Каляева неоднократно отмечались зарубежными учеными в ряде ведущих иностранных научных журналов, демонстрировались на престижных международных выставках.

А.В. Каляев был организатором и в течение 20 лет председателем первого в ТРТУ совета по защитам докторских диссертаций. Состоял членом советов по защитам докторских диссертаций в ряде других научных организаций. Являлся членом Комитета по государственным премиям РСФСР в области науки и техники при Совмине РСФСР. Был членом редакционных советов и редакционных коллегий центральных научных журналов: «Микропроцессорные средства и системы» (Москва), «Электронное моделирование» (Киев), «Нейрокомпьютер» (Москва), «International Journal of Intelligent Control and Fuzzy Logic» (Нью-Йорк, США). Являлся членом Совета Северо-Кавказского научного центра высшей школы и председателем отделения радиоэлектроники, вычислительной техники и систем СКНЦ ВШ, членом Президиума Международной Академии наук высшей школы, членом Научного Совета Российской академии наук по суперЭВМ. А.В. Каляевым лично подготовлены 17 докторов и 56 кандидатов наук.

Академик А.В. Каляев – заслуженный деятель науки и техники России, действительный член (Senior Member) Американского общества инженеров электротехники и электроники (IEEE).

За большой личный вклад в развитие вычислительной техники и подготовку научных и инженерных кадров А.В. Каляеву присвоено звание Героя Социалистического труда.

А.В. Каляев – участник Великой Отечественной войны и героической обороны Ленинграда. Он награжден шестью орденами и шестнадцатью медалями.

Представляемый сборник включает доклады, посвященные развитию идей академика А.В. Каляева и их практическому воплощению и реализации в различных организациях и предприятиях страны.

Надеемся, что проведение конференции будет способствовать повышению уровня фундаментальных и прикладных исследований в области многопроцессорных вычислительных и управляющих систем и их практического применения в различных сферах человеческой деятельности.

Желаем всем участникам и гостям конференции плодотворной творческой работы и дружеского общения.

Оргкомитет МВУС-2022

СОДЕРЖАНИЕ

РАЗДЕЛ 1. АРХИТЕКТУРА И СПОСОБЫ ОРГАНИЗАЦИИ ВЫЧИСЛИТЕЛЬНЫХ ПРОЦЕССОВ В МВУС

Боев С.Ф., Петренко А.С., Петренко С.А., Ступин Д.Д.
Квантовый криптоанализ криптосхем блокчейн-платформ **12**

Боев С.Ф., Петренко С.А., Ступин Д.Д.
Возможности построения адаптивной защиты информации в масштабируемых вычислительных структурах информационных систем **21**

Левин И.И., Доронченко Ю.И., Федоров А.М., Раскладкин М.К.
Перспективный суперкомпьютер «Арктур» **25**

Непомнящий О.В., Легалов А.И., Рыженко И.Н.
Методы редукции параллелизма в процессе высокоуровневого синтеза СБИС **30**

Павский В.А., Павский К.В.
Математическая модель вычислительной системы с накопителем для расчета показателей надежности и осуществимости решения задач **34**

Смелянский Р.Л.
Организация вычислений: Quo Vadis? **38**

Сойфер В.А.
Достижения и перспективы аналоговых оптических вычислений **45**

Степаненко С.А.
Фотонная вычислительная машина. Оптические логические элементы **50**

Шагалиев Р.М.
Направления работ и актуальные задачи развития суперкомпьютерных технологий, включая высокопроизводительную вычислительную базу **61**

Щаников С.А., Борданов И.А., Букварев Е.А., Хранилов В.П., Грязнов Е.Г., Михайлов А.Н.
Масштабируемые аппаратные ускорители искусственного интеллекта на базе мемристивных устройств **64**

РАЗДЕЛ 2. МАТЕМАТИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ МВУС

Баркалов К.А., Козинов Е.А., Лебедев И.Г., Сысоев А.В., Усова М.А.
Параллельные методы решения задач многокритериальной многоэкстремальной оптимизации для многопроцессорных систем **68**

Вахлаков Д.В., Мельников С.Ю., Пересыпкин В.А.
Задача коррекции искаженных текстов на естественных языках. Актуальность и состояние дел. Проблемы и перспективы **73**

Данилов И.Г., Ерошенко И.Н.
Симулятор программного обеспечения системы мониторинга и тестирования РВБ **78**

Духнич Е.И., Чефранов А.Г., Шапель А.П.
Модифицированный шифр Хилла в пространстве кватернионов **81**

Крамаренко К.Е., Поляков А.Ю., Ефимов А.В.
Алгоритм синхронизации доступа к общей памяти на основе блокировки MCS **86**

Лебедев А.С.
Построение пространственных и временных преобразований программ для распараллеливания вложенностей циклов **90**

Мельник Э.В., Орда-Жигулина М.В., Орда-Жигулина Д.В.
Интеграция компонентов системы мониторинга и диагностики при решении задач различной направленности **95**

Мельников А.К.
Уточнение классификации типов дискретных распределений для организации эффективной обработки последовательностей на многопроцессорных вычислительных системах **98**

Подопригора А.В.
Объединение базовых БРН-матричных макроопераций **103**

Титенко Е.А.
Метод параллельно-конвейерного поиска по образцу **108**

Ченцов А.Г., Григорьев А.М., Ченцов А.А.
Маршрутизация перемещений при условиях предшествования: методы и алгоритмы **114**

РАЗДЕЛ 3. РЕКОНФИГУРИРУЕМЫЕ ВЫЧИСЛИТЕЛЬНЫЕ И УПРАВЛЯЮЩИЕ СИСТЕМЫ

Абросимов В.К.
Реконфигурируемое взаимодействие в киберфизической группе объектов управления **120**

Алексеев К.Н., Жуков А.Л., Касаркин А.В., Коваленко А.Г., Семенов И.С., Сорокин Д.А.
Конвейерная реализация задачи обучения сверточных нейросетей на реконфигурируемых вычислительных системах **124**

Алексеев К.Н., Сорокин Д.А., Леонтьев А.Л.
Синтезатор топологических ограничений ПЛИС **129**

Борзов Д.Б., Масюков И.И.
Метод расположения задач в реконфигурируемых вычислительных системах **134**

Буряков Д.С., Левин И.И.
Система передачи данных на основе ПЛИС в устройствах цифровой обработки сигналов с фазированными антенными решетками **138**

Гудков В.А., Дордопуло А.И., Михайлов Д.В., Кузнецова И.Ю.
Описание информационных графов с ассоциативными операциями в ресурсонезависимой форме на языке программирования Set@l **143**

Дудников Е.А.
Сжатие плотных потоков разнородных данных на РВС в темпе их поступления **146**

Мельник Э.В., Таранов А.Ю.
Улучшение надежностных характеристик бортовых вычислительных систем за счет применения реконфигурации **149**

Сластен Л.М., Кузнецова И.Ю., Механцев Б.Е., Гуленок А.А.
Архитектурно-независимое программирование параллельных вычислительных систем на языке Set@l **152**

Чекина М.Д.
Решение задачи супердиффузии на реконфигурируемых вычислительных системах **155**

РАЗДЕЛ 4. ПРОБЛЕМНО-ОРИЕНТИРОВАННЫЕ ВЫЧИСЛИТЕЛЬНЫЕ И УПРАВЛЯЮЩИЕ СИСТЕМЫ

Борзов Д.Б., Чернышев А.А.
О методике оценки времени распределения в распределенной вычислительной системе реального времени **161**

Бычков И.В., Феоктистов А.Г., Костромин Р.О.
Адаптивная среда моделирования работы инфраструктурных объектов **164**

Винтайкин И.Б., Табалин С.Е., Назолин А.Л., Фуфурин И.Л., Голяк Ил.С., Глушков В.Л., Морозов А.Н.
Встраиваемые вычислительные системы управления работой фурье-спектрометров для дистанционной локации химических соединений **168**

Витиска Н.И., Георгиева М.А.
ОДИН ИЗ ПОДХОДОВ К МАТЕМАТИЧЕСКОМУ МОДЕЛИРОВАНИЮ УПРАВЛЕНИЯ ЭРОЗИОННЫМИ ПРОЦЕССАМИ НА БАЗЕ МНОГОПРОЦЕССОРНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ С ОБЩЕЙ ШИНОЙ **171**

Витиска Н.И., Гуляев Н.А.
О ПРОБЛЕМАХ РЕАЛИЗАЦИИ ПРОГРАММНО-АППАРАТНЫХ СИСТЕМ ПРЯМОЙ ОБЪЁМНОЙ ВИЗУАЛИЗАЦИИ **175**

Грузликов А.М.
ВЫСОКОУРОВНЕВОЕ ПРОЕКТИРОВАНИЕ БОРТОВЫХ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ **181**

Дорохина Г.В., Гаркуша Д.А.
ВЫСОКОУРОВНЕВОЕ ПРОЕКТИРОВАНИЕ БОРТОВЫХ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ **185**

Сафроненкова И.Б., Клименко А.Б.
О ВОЗМОЖНОСТИ РЕШЕНИЯ ОПТИМИЗАЦИОННЫХ ЗАДАЧ В ТУМАННЫХ СРЕДАХ **190**

Семенистый С.А.
ПЛАНИРОВАНИЕ ЗАДАЧ В РАСПРЕДЕЛЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ ПРИ ОБРАБОТКЕ ПОТОКОВ ВИДЕОДАННЫХ **194**

Фуфурин И.Л., Голяк Иг.С., Карева Е.Р., Щербакова А.В., Небритова О.А., Демкин П.П., Бережанский П.В., Морозов А.Н.
ПРОБЛЕМНО-ОРИЕНТИРОВАННЫЕ СИСТЕМЫ ЭКСПРЕСС-ДИАГНОСТИКИ СОСТОЯНИЯ ЗДОРОВЬЯ ЧЕЛОВЕКА НА ОСНОВЕ АНАЛИЗА ВЫДЫХАЕМОГО ВОЗДУХА С ПРИМЕНЕНИЕМ СВЕРТОЧНЫХ НЕЙРОННЫХ СЕТЕЙ **198**

РАЗДЕЛ 5. ИНТЕЛЛЕКТУАЛЬНЫЕ И РОБОТОТЕХНИЧЕСКИЕ СИСТЕМЫ

Голяк Иг.С., Анфимов Д.Р., Винтайкин И.Б., Назолин А.Л., Глушков В.Л., Голяк Ил.С., Морозов А.Н.
МНОГОПРОЦЕССОРНЫЕ ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ КОНТРОЛЯ БЕЗОПАСНОСТИ ГОРОДСКОЙ СРЕДЫ С ЦЕЛЬЮ ДИСТАНЦИОННОГО ОБНАРУЖЕНИЯ БЕСПИЛОТНЫХ ЛЕТАТЕЛЬНЫХ АППАРАТОВ **202**

Гушанский С.М., Буглов В.Е.
Применение квантовых нейронных сетей для распознавания образов **206**

Гушанский С.М., Потапов В.С.
Исследование и разработка квантовых сверточных нейронных сетей Фурье **211**

Зеленский А.А.
Синтез систем управления движением реального времени промышленных роботов и мехатронных систем **217**

Зуев В.М., Близно М.В., Никитина А.А.
Метод уменьшения объема вычислений сверточной нейронной сети при распознавании изображений **221**

Колесников В.И., Пашков Д.М., Колесников И.В., Гуда А.А., Беляк О.А.
Обратное проектирование механических свойств покрытий методами машинного обучения **225**

Остроухов А.Ю., Пуха И.С., Мельник Э.В.
Программные средства агентного моделирования на базе мультиагентного диспетчера в распределенной среде с GPU **230**

Титенко Е.А., Щитов А.Н.
Подход к созданию схемы комплексирования данных для системы технического зрения **233**

Туровский Я.А., Мещеряков Р.В., Бирин Д.А.
Применение гибридного интеллекта для управляющей системы с искусственным интеллектом **238**

Фролов В.В., Пигуз В.Н.
Интеллектуальные системы видеонаблюдения со встроенной видеоаналитикой для решения задач охранной сигнализации **241**

Черских Е.О., Савельев А.И., Ронжин А.Л.
Событийно-ориентированная модель управления робототехническим средством с параллельными вычислениями на микроконтроллерах **245**

АВТОРСКИЙ УКАЗАТЕЛЬ **250**

РАЗДЕЛ 1

АРХИТЕКТУРА И СПОСОБЫ ОРГАНИЗАЦИИ ВЫЧИСЛИТЕЛЬНЫХ ПРОЦЕССОВ В МВУС

С.Ф. Боев[1], А.С. Петренко[2], С.А. Петренко[2], Д.Д. Ступин[1, 2]

КВАНТОВЫЙ КРИПТОАНАЛИЗ КРИПТОСХЕМ БЛОКЧЕЙН-ПЛАТФОРМ*

[1]МАК «Вымпел», г. Москва,
[2]Московский физико-технический институт, г. Москва,
ddstupin@yandex.ru

Введение

Современные блокчейн-платформы состоят из ряда независимых узлов, которые достигают консенсуса перед обновлением соответствующих реестров новыми транзакциями. Известно несколько механизмов достижения консенсуса, например, Proof-of-Work (PoW). Отметим, что применение механизмов консенсуса и криптографических примитивов (см. табл. 1) уже недостаточно для нейтрализации квантовой угрозы и обеспечения требуемой квантовой устойчивости блокчейн-платформ, если под квантовой устойчивостью понимать системное свойство платформы сохранять постоянство ключевых системных свойств в условиях криптографических атак с использованием квантовых компьютеров [2-5,10].

Оценка развития алгоритмической базы показывает, что появление эффективных квантовых вычислителей следует ожидать к 2025 году. В ноябре 2021 года IBM представила 127-кубитовый процессор Eagle, а к 2023 году прогнозирует преодоление 1000-кубитового предела (https://newsroom.ibm.com/2021-11-16-IBM-Unveils-Breakthrough-127-Qubit-Quantum-Processor). В свое время, исследователями компании Google было показано, что для эффективного криптоанализа RSA достаточно порядка 20 млн физических кубитов (https://arxiv.org/pdf/1905.09749.pdf). С учетом возможности эффективного распараллеливания вычислений между несколькими устройствами с существенно меньшим числом кубитов достижения IBM свидетельствуют о реалистичности квантовой угрозы. Поэтому в ряде стран, главным образом в США и Евросоюзе запланирован в течение 2022–2025 гг. переход к устойчивой квантовой (пост-квантовой) криптографии (https://nvlpubs.nist.gov/nistpubs/specialpublications/nist.sp.800-175b.pdf). Так, упомянутый NIST находится в процессе разработки стандартов

* Статья подготовлена по результатам исследований, выполненных при поддержке гранта РФФИ (№ 20–04–60080)

квантовой криптографии, а АНБ рекомендует своим поставщикам внедрить SHA-384 вместо SHA-256.

Таблица 1

Характеристика объекта исследования

Криптовалюта	Электронная подпись	Параметры	Алгоритм консенсуса	Регулировка сложности	Целевой интервал времени блокировки
Bitcoin	ECDSA	Secp256k1	Proof of Work	Отсутствует	10 мин.
Litecoin	ECDSA	Secp256k1	Proof of Work	Отсутствует	2,5 мин.
Namecoin	ECDSA	Secp256k1	Proof of Work	Детерминированное обязательство	10 мин.
Dogecoin	ECDSA	Secp256k1	Proof of Work	Отсутствует	1 мин.
Primecoin	ECDSA	Secp256k1	Proof of Work	Сложность регулируется в каждом блоке	1 мин.
Auroracoin	ECDSA	Secp256k1	Proof of Work	Каждые 8 блоков	1 мин.
Dash	ECDSA	Secp256k1	Proof of Work	Механизм DGW	2,5 мин.
Vertcoin	ECDSA	Secp256k1	Proof of Work	В каждом блоке	2,5 мин.
Ethereum	ECDSA	Secp256k1	Proof of Work	Отсутствует	15 сек.
Zcash	ECDSA EDDSA	Secp256k1 (по умолчанию) Ed25519 (защищённый вариант)	Proof of Work	Отсутствует	75 сек.
Bitcoin Cash	ECDSA	Secp256k1	Proof of Work	Экстренная регулировка сложности	10 мин.

Вербальная и математическая задача исследования

Вербальная постановка задачи.

Дано: Схемы асимметричного шифрования (RSA, Эль-Гамаля) и цифровой подписи (DSA, ECDSA или RSA-PSS) (см. табл. 2) исследуемых блокчейн-платформ: InnoChain (Innopolis University), Waves Enterprise (Waves, Vostok), Hyperledger Fabric (Linux, IBM), Corda Enterprise, Bitfury Exonum, Blockchain Industrial Alliance, Exonum (Bitfury CIS), NodesPlus (b41), Мастерчейн (Сбербанк), Microsoft Azure Blockchain, Enterprise Ethereum Alliance и др.

Необходимо: Повысить эффективность криптоанализа систем асимметричного шифрования и цифровой подписи исследуемых платформ с учетом стойкости дискретного алгоритма (DLP) и дискретного алгоритма с эллиптической кривой (ECDLP) в зависимости от:

– трудоемкости алгоритма криптоанализа систем асимметричного шифрования и цифровой подписи;

– имеющихся реальных вычислительных ресурсов (типа компьютера; количества логических и физических кубитов; уровней глубины моделируемой квантовой схемы; показателей стабильности квантового вычислителя; производительности компьютера в целом);

– вероятности взлома системы.

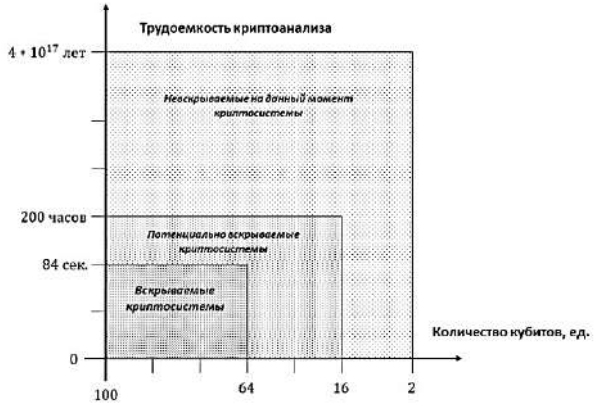


Рис. 1. Область гарантированного решения задачи квантового криптоанализа криптопримитивов блокчейн-платформ с требуемой результативностью.

Математическая постановка задачи

Исходные данные:

– Структура схемы асимметричного шифрования (RSA, Эль-Гамаля) и/или цифровой подписи (DSA, ECDSA или RSA-PSS) исследуемых блокчейн-платформ InnoChain (Innopolis University), Waves Enterprise (Waves, Vostok), Hyperledger Fabric (Linux, IBM), Corda Enterprise, Bitfury Exonum, Blockchain Industrial Alliance, Exonum (Bitfury CIS), NodesPlus (b41), Мастерчейн (Сбербанк), Microsoft Azure Blockchain, Enterprise Ethereum Alliance и др.

– Приложения и/или протоколы (TLS, SSH и IPSec).

Набор ограничений:

– Ограничения на W – значение показателя трудоемкости алгоритма криптоанализа (*в сек.*) схемы асимметричного шифрования (*RSA, Эль-Гамаля*) и/или цифровой подписи (*DSA, ECDSA или RSA-PSS*);

– Ограничения на N – временные и вычислительные ресурсы для проведения криптоатаки (допустимый период времени на криптоатаку, тип компьютера – квантовый компьютер или симулятор на классическом компьютере, количество логических и физических кубитов, количество уровней глубины моделируемой квантовой схемы, значения показателей стабильности функционирования квантового вычислителя в условиях когеренции, значения показателей производительности компьютера в целом);

– Ограничения по значению P – вероятности успешно проведенной криптоатаки на схему асимметричного шифрования (*RSA, Эль-Гамаля*) и/или цифровой подписи (*DSA, ECDSA или RSA-PSS*).

В результате анализа требуется найти:

– Числовую комплексную оценку, характеризующую стойкость схемы асимметричного шифрования (*RSA, Эль-Гамаля*) и/или цифровой подписи (*DSA, ECDSA или RSA-PSS*) исследуемых блокчейн-платформ к квантовым алгоритмам криптоанализа (под успешным выполнением криптоанализа понимается восстановление секретного ключа шифрования или получение возможности дешифровать сообщение без изначального знания секретного ключа)

$$F_{\Sigma} = \sum_{i=1}^{s} f_i\left(W, N, P_{def}\right), F_{\Sigma} \geq F_{def},$$

где F_{def} – эталонное значение,

$$f_i\left(W, N, P_{def}\right) = \begin{cases} 0, \forall W, N: P_{W,N} < P_{def} \\ 1, P_{W,N} \geq P_{def} \\ \dfrac{P_{def}}{\left(1 + \dfrac{W'}{W}\right)\left(1 + \dfrac{N'}{N}\right)}, P_{W,N} < P_{def}, \exists W', N': P_{W',N'} \geq P_{def}. \end{cases}$$

– Развернутую характеристику стойкости схем асимметричного шифрования (RSA, Эль-Гамаля) и/или цифровой подписи (DSA, ECDSA или RSA-PSS) к квантовым алгоритмам криптоанализа в виде таблицы 2 содержащей нечеткие оценки из заданного множества.

Таблица 2

Характеристика объекта исследования

Для оценивания компонентов криптосхемы	**Для оценивания криптосхемы в целом**
Способствует	Возможен
Не влияет	Затруднен
Затрудняет	Существенно затруднен
Делает невозможным	Невозможен

Предлагаемое решение. Стойкость алгоритмов симметричной и асимметричной криптографии основывается на сложности решения определенных классов задач на классических компьютерах (*перебор, факторизация и дискретное логарифмирование* лежат в классе сложности *NPP* по длине ключа в битах) в противоположность квантовой криптографии, стойкость которой основывается на законах квантовой физики. Оценка временной сложности основных алгоритмов факторизации и дискретного логарифмирования представлены в Табл. 3.

Типовые схемы асимметричного шифрования (RSA, Эль-Гамаля) и цифровой подписи (DSA, ECDSA или RSA-PSS) блокчейн основываются на сложности вычисления дискретного логарифма в конечных группах, определенных над различными алгебраическими конструкциями, или на сложности разложения натурального числа на простые сомножители. Степень уязвимости упомянутых схем базируется на вычислительной сложности факторизации чисел [3,6-10].

Заметим, что задачи дискретного логарифмирования и факторизации имеют асимптотическую сложность при решении на квантовом компьютере $O(n^2 \, log \, n \, log \, log \, n)$ квантовых шагов. На классическом компьютере с использованием лучшего из известных алгоритмов – алгоритма решета числового поля – асимптотическая сложность составляет $O(e^{cn^{1/3} log^{2/3} n})$, где c – некоторая константа. Полиномиальное решение задач дискретного логарифмирования на квантовом компьютере находится также за $O(n^2 \, log \, n \, log \, (log \, n))$.

Таблица 3

Временная сложность алгоритмов, решающих задачи факторизации и дискретного логарифмирования

Название	Сложность
Решение задачи факторизации	
Метод Ферма	$T(N) = O(N^{1/3})$
Метод Ленстры	$T = O(\exp(\sqrt{2\ln p\ \ln\ln p}))$
Метод Диксона	$T = O(L(n)^2)$
Метод квадратичного решета	$T = O(\exp\left((1+o(1))\sqrt{\log n \log\log n}\right))$
Метод решета числового поля	$T(N) = O(n \log n \log N)$
Метод Шора	$T = O(\log_3 M)$
Решение задачи дискретного логарифмирования	
Метод Адлемана	$T = O(c^{\ln p^{\frac{1}{2}}})$
Метод COS	$T = O(\exp((\log p \log\log p)^{\frac{1}{2}}))$
Метод решета числового поля	$T(N) = O(n \log n \log N)$
Метод Шора	$T = O(\log_3 M)$

В результате, практическая значимость квантовых алгоритмов решения задач факторизации и дискретного логарифмирования заключается в том, что с их помощью при использовании квантового компьютера с несколькими сотнями логических кубитов становится возможным взлом криптографических систем с открытым ключом. Например, *RSA* использует открытый ключ M, являющийся произведением двух больших простых чисел. Один из способов взломать шифр *RSA* – найти его множители. При достаточно большом M это практически невозможно сделать, используя известные классические алгоритмы. Лучший из известных классических алгоритмов факторизации требует времени порядка $M^{1/3}$. *Алгоритм Шора*, используя возможности квантовых компьютеров, способен произвести факторизацию числа не просто за полиномиальное время, а за время, ненамного превосходящее время умножения целых чисел (то есть практически так же быстро, как происходит само шифрование) [1,3-7].

1.	$\lvert 0\rangle\lvert 0\rangle$	*-Инициирование состояния*
2.	$\rightarrow \frac{1}{\sqrt{2^t}}\sum_{x=0}^{2^t-1}\lvert x\rangle\lvert 0\rangle$	*-Создание суперпозиции*
3.	$\rightarrow \frac{1}{\sqrt{2^t}}\sum_{x=0}^{2^t-1}\lvert x\rangle\lvert f(x)\rangle$	*-Применить U* $\quad f(x) = ax(mod\ N)$
	$\approx \frac{1}{\sqrt{r2^t}}\sum_{\ell=0}^{r-1}\sum_{x=0}^{2^t-1} e^{2\pi i \ell x/r}\lvert x\rangle\lvert \hat{f}(\ell)\rangle$	
4.	$\rightarrow \frac{1}{\sqrt{r}}\sum_{\ell=0}^{r-1}\lvert \widetilde{\ell/r}\rangle\lvert \hat{f}(\ell)\rangle$	*-Применить QFT к первому регистру*
5.	$\rightarrow \widetilde{\ell/r}$	*-Измерить первый регистр*
6.	$\rightarrow r$	*-Применить алгоритм цепных дробей*

Рис. 2. Основные этапы квантового алгоритма факторизации Шора.

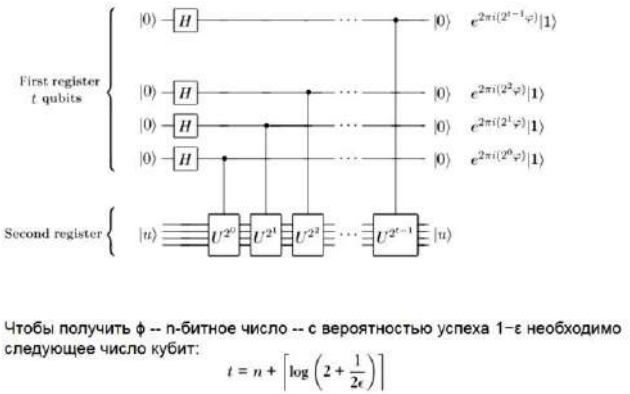


Рис. 3. Нахождение необходимого количества кубитов, как критерий успешности решения задачи криптоанализа.

Понятно, что здесь идет речь не только о *схеме RSA*, но и о других сходных схемах асимметричного шифрования (производные от RSA, Эль-Гамаля) и цифровой подписи (DSA, ECDSA или RSA-PSS), которые становятся уязвимыми.

$f(x) = a^x \bmod N.$

$7^4 \bmod 15 \equiv 1 \Rightarrow 7^4 - 1 \bmod 15 \equiv 0 \Rightarrow (7^2 - 1)(7^2 + 1) \bmod 15 \equiv 0 \bmod 15 \Rightarrow 15$ divides $48 \cdot 50$

$\gcd(48, 15) = 3$ $\gcd(50, 15) = 5$

$N = 15.$

$x = 7$

t=11 ε=1/4

$$\frac{1}{\sqrt{2^t}} \sum_{k=0}^{2^t-1} |k\rangle|0\rangle = \frac{1}{\sqrt{2^t}} \left[|0\rangle + |1\rangle + |2\rangle + \cdots + |2^t - 1\rangle\right] |0\rangle$$

$t = 11$ H

$$f(k) = x^k \bmod N$$

$$\frac{1}{\sqrt{2^t}} \sum_{k=0}^{2^t-1} |k\rangle|x^k \bmod N\rangle = \frac{1}{\sqrt{2^t}} \left[|0\rangle|1\rangle + |1\rangle|7\rangle + |2\rangle|4\rangle + |3\rangle|13\rangle + |4\rangle|1\rangle + |5\rangle|7\rangle + |6\rangle|4\rangle + \cdots\right]$$

$FT^\dagger$ $\widetilde{\ell/r}$

Рис. 4. Пример квантового решения задачи факторизации.

Заключение

В ходе работы был разработан специальный ПАК «Квант-К», адаптированный для работы в вычислительной гибридной среде IBM Q (16, 20, 100 и более кубитов) и РВС на основе ПЛИС [1-3].

Впервые для оценивания квантовой устойчивости известных блокчейн-платформ был предложен и обоснован новый вид квантового криптоанализа схем асимметричного шифрования (*RSA, Эль-Гамаля*) и цифровой подписи (*DSA, ECDSA или RSA-PSS*) на основе модифицированного алгоритма факторизации Шора.

Удалось повысить оперативность исследования криптостойкости схем асимметричного шифрования (*RSA, Эль-Гамаля*) и цифровой подписи (*DSA, ECDSA или RSA-PSS*) исследованных блокчейн платформ на **20–30%** за счет автоматизации первой и второй части модифицированного (квантового) алгоритма факторизации Шора в гибридной среде IBM *Q* (16, 20, 100 и более кубитов) и РВС на основе ПЛИС.

Произошло увеличение доли квантовых алгоритмов криптоанализа в арсенале инструментальных средств криптоанализа схем асимметричного шифрования (*RSA, Эль-Гамаля*) и цифровой подписи (*DSA, ECDSA или RSA-PSS*) блокчейн-платформ **до 17%**.

Удалось повысить результативность криптоанализа асимметричных шифров схем асимметричного шифрования (*RSA, Эль-Гамаля*) и цифровой подписи (*DSA, ECDSA или RSA-PSS*) блокчейн-платформ, исследованных в работе, на **10–15%** в целом**.**

Значимость полученных результатов состоит в выработке решения для вычислительно трудных задач факторизации и дискретного логарифмирования в РВС на основе ПЛИС, заданных над конечными коммутативными (и некоммутативными) ассоциативными алгебрами, в квантовой модели вычислений за полиномиальное время. Разработана методика применения программного комплекса «Квант-К» для оценивания квантовой устойчивости ряда блокчейн-платформ.

1. **Гузик, В.Ф.** Реконфигурируемые вычислительные системы / В.Ф. Гузик, И.А. Каляев, И.И. Левин; под общ. ред. И.А. Каляева. – Ростов-на-Дону: Изд-во ЮФУ. – 2016. – 472 с.
2. **Каляев, И.А.** Реконфигурируемые мультиконвейерные вычислительные структуры / И.А. Каляев, И.И. Левин, Е.А. Семерников, В.И. Шмойлов; под общ. ред. И.А. Каляева. – 2-е изд., переработанное и дополненное – Ростов-на-Дону: Изд-во ЮНЦ РАН. – 2009. – 344 с.
3. **Петренко, А.С.** Перспективный метод криптоанализа на основе алгоритма Шора / А.С. Петренко, А.М. Романченко // Защита информации. Inside № 2 2020. – СПб.: Изд. Афина, 2020. – С. 17–23.
4. **Холево, А.С.** Математические основы квантовой информатики / А.С. Холево. – М.: МИАН, 2018. – 118 с. – (Лекц. курсы НОЦ, ISSN 2226-8782; Вып. 30).
5. **Shor P.** Algorithms for quantum computation: discrete logarithms and factoring / P. Shor // Foundations of Computer Science. – 1994. – № 10. – 134 p.
6. **Deutsch, D.** Rapid solution of problems by quantum computation / D. Deutsch, R. Jozsa // Proceedings of the Royal Society of London A. – 1992. – Vol. 439, Is. 1907. – P. 553–558.
7. **Diffie, D.** New directions in cryptography / D. Diffie, M. Hellman // IEEE Transactions on Information Theory. – 1976. – Vol. 22, Is. 6.
8. **Grover, L.K.** A fast quantum mechanical algorithm for database search / L.K. Grover // Proceedings of the twenty-eighth, annual ACM symposium on Theory of computing. – 1996. – P. 212–219.
9. **Shor, P.W.** Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer / P.W. Shor // SIAM J. Computing. – 1997. – Vol. 26. – P. 1484–1509.
10. **Simon, D.R**. On the power of quantum computation / D.R. Simon // SFCS '94: Proceedings of the 35th Annual Symposium on Foundations of Computer Science. – 1994. – P. 116–123.

С.Ф. Боев[1], С.А. Петренко[2], Д.Д. Ступин[1,2]

ВОЗМОЖНОСТИ ПОСТРОЕНИЯ АДАПТИВНОЙ ЗАЩИТЫ ИНФОРМАЦИИ В МАСШТАБИРУЕМЫХ ВЫЧИСЛИТЕЛЬНЫХ СТРУКТУРАХ ИНФОРМАЦИОННЫХ СИСТЕМ

[1]МАК «Вымпел», г. Москва,
[2]Московский физико-технический институт, г. Москва,
ddstupin@yandex.ru

Введение

В информационных системах, связанных с обеспечением работы объектов критически важной инфраструктуры, объемы и потоки обрабатываемой информации постоянно увеличиваются. Из-за возрастания информационной нагрузки на систему возможно возникновение ситуации, когда ее собственных вычислительных ресурсов не хватает для решения всех возложенных на нее задач. Одним из способов преодоления этой проблемы является привлечение «внешних» вычислительных ресурсов, например, ресурсов «партнерских» (смежных по задачам и методам их решения) систем или ресурсов «общего назначения», например, предоставляемых существующими дата-центрами. При этом большую актуальность приобретает проблема обеспечения информационной безопасности, поскольку привлечение внешних ресурсов повышает риски «недружественных воздействий» из вне. В работе рассматриваются принципы обеспечения информационной безопасности масштабных информационных систем, в которых для обеспечения необходимого качества решения задач привлекаются внешние вычислительные ресурсы [1-3].

Проблема достаточности вычислительных ресурсов в современных информационных системах

Любая информационная система, имеющая отношение к критически важной инфраструктуре, строится, исходя из требований обеспечения работы в условиях максимальных информационных нагрузок, характерных для экстремальных ситуаций. В большинстве случаев этот подход является и оправданным, и результативным. Однако сегодня достаточно часто возникают ситуации, когда объект мониторинга информационной системы (особенно это касается техногенных сред) начинает меняться с темпом, на который не была запланирована система, и объем информации, требуемый для обработки, растет лавинообразно. В качестве примера можно назвать околоземное космическое пространство, количество объектов в котором вследствие высоких темпов роста группировок различных КА увеличивается в несколько раз по сравнению с ситуацией 5–7-летней давности.

Несмотря на то, что при разработке информационной системы параметры ее вычислительной среды задаются, как правило, с учетом прогнозируемого роста информационной нагрузки, в случае ошибочного прогноза система столкнется с дефицитом вычислительных ресурсов, что приведет к снижению качества ее работы. Наращивание вычислительных ресурсов в системе, связанной с критически важной инфраструктурой, не может происходить постоянно; как правило, оно происходит во время ее очередной модернизации, которая не происходит часто, хотя бы в силу экономических соображений. Поэтому разработчику системы необходимо иметь запасные варианты, чтобы в случае ужесточения внешних условий обеспечить ее работу с требуемыми характеристиками [1-3].

Привлечение «сторонних» вычислительных ресурсов: возможности и проблемы

Привлечение «сторонних» вычислительных ресурсов является достаточно нетрадиционным способом повышения возможностей системы в плане обработки больших массивов и потоков информации. Эта возможность появилась только в последние 10–15 лет, прежде всего, вследствие глобального распространения разнообразных сетевых технологий. Источниками дополнительных ресурсов могут являться другие информационные системы, которые могут рассматриваться по отношению к рассматриваемой системе в качестве «партнерских», схожих по назначению, решаемым задачам и, возможно, методам решения (мы не ставим целью строго определить термин «партнерский», оставляя это на интуитивном уровне восприятия). Другой вариант – привлечение ресурсов дата-центров специального или общего назначения, по сути дела это сводится к применению «облачных» технологий. Оба варианта имеют свои достоинства и недостатки, мы постараемся анализировать их только с точки зрения обеспечения информационной безопасности. Отметим основные особенности процесса привлечения «сторонних» ресурсов:

1. К решению задач обработки информации привлекаются, как правило, разнородные вычислительные средства; при этом информационная система не всегда «знает», какого типа средства привлекаются.
2. Привлекаемые вычислительные средства могут быть распределены по пространству или находиться на отдалении по отношению к вычислительным средствам системы, что чревато возникновением временных задержек информационных потоков; это необходимо учитывать при решении задач в реальном времени.
3. Необходимость управления организацией вычислительных процессов требует специальной управляющей программы – диспетчера, являющегося элементом ПАО информационной системы. Если речь идет о привлечении ресурсов «партнерской» системы, то с большой вероятностью у нее есть

«свой» диспетчер. Если же привлекаются ресурсы общего назначения, то необходимо формирование «виртуального» диспетчера, обеспечивающего управление вычислительным процессом (например, основной диспетчер на время привлечения «сторонних» средств может сформировать своего «двойника» в той среде, которая используется).

В любом случае, привлечение «сторонних» ресурсов снижает возможности системы с точки зрения информационной безопасности, поскольку появляются дополнительные информационные каналы для «недружественных» действий. Поэтому процесс привлечения вычислительных ресурсов должен иметь адекватную программную поддержку с точки зрения информационной безопасности. Рассмотрим основные принципы реализации такой поддержки для двух вариантов действий: когда конфигурация и параметры привлекаемых вычислительных ресурсов известны (например, в случае работы с «партнерской» системой) и когда эта информация отсутствует (например, для случая дата-центров) [2,3].

Дополнительные меры по защите информации при известной конфигурации вычислительных ресурсов

Как указывалось выше, подобная ситуация характерна для привлечения ресурсов «партнерских» систем, имеющих свою систему информационной защиты. Это обстоятельство, в принципе, снижает риск недружественных воздействий, хотя не исключает его полностью. При известной конфигурации и параметрах привлекаемых вычислительных ресурсов целесообразно проводить следующие операции:

1. Для вновь сформированной (с учетом имеющихся и привлеченных вычислительных средств) конфигурации вычислительных ресурсов проводится анализ потенциальных уязвимостей с точки зрения информационной безопасности.
2. Реализуются превентивные меры по «купированию» уязвимостей с учетом имеющегося опыта защиты информации в системе.
3. Программа-диспетчер формирует специальную подсистему мониторинга вычислительного процесса (на базе имеющейся штатной системы), ориентированную на обнаружение (в перспективе мы сможем говорить о прогнозировании и обнаружении) «недружественных» действий.

После завершения процесса обработки информации с использованием «сторонних» ресурсов система обеспечения информационной безопасности приводится в штатное состояние [1-3].

Дополнительные меры по защите информации при априори неизвестной конфигурации вычислительных ресурсов

В случае привлечения к обработке информации в системе вычислительных ресурсов общего назначения можно априорно предполагать, что

основным источником уязвимостей являются интерфейсы между вычислительными ресурсами системы и привлекаемыми ресурсами. Соответственно, меры по информационной защите должны предполагать минимизацию информационных потоков. Со стороны информационной системы в качестве информационного «посредника» предполагается использовать программу-диспетчер.

Предполагается следующая последовательность действий:

1. Программа-диспетчер на базе привлеченных ресурсов создает специальную управляющую программу (это может быть виртуальный «двойник» диспетчера или той его части, которая обеспечивает управление вычислениями), способную автономно управлять вычислительным процессом. При этом в процессе обработки информации взаимодействие сводится к индикации хода процесса («не начат», «идет», «завершен»).

2. После завершения процесса обработки информации созданный «двойник» пересылает программе-диспетчеру информацию о местонахождении результатов обработки и затем получает от диспетчера команду на самоликвидацию.

3. Перенос результатов обработки в память вычислительных средств информационной системы обеспечивает программа-диспетчер, формируя с этой целью еще одного виртуального «двойника», ориентированного на перенос информации из «облака» в память вычислительных средств системы. После окончания переноса информации и передачи соответствующего информационного признака диспетчеру этот «двойник» также получает команду на самоликвидацию.

Заметим, что в данном варианте не планируется специальных мер по защите информации; предполагается, что защита обеспечивается через минимизацию (по объему информации и по времени взаимодействия) контактов информационной системы с внешними вычислительными ресурсами [1,2].

Заключение

Представленная работа содержит подход к обеспечению защиты информации в информационной системе, которая для расширения возможностей обработки информации может привлекать «сторонние» вычислительные ресурсы. Тенденции развития информационных систем мониторинга пространственных и техногенных сред свидетельствуют о востребованности таких дополнительных информационных возможностей и, соответственно, об актуальности такого подхода.

1. **Петренко, С.А.** Национальная система раннего предупреждения о компьютерном нападении / С.А. Петренко, Д.Д. Ступин; научная монография под общей редакцией С.Ф. Боева // Издательский дом «Афина». – 2018. – 440 с. – ISBN 978-5-9909868-0-0.

2. **Гузик, В.Ф.** Реконфигурируемые вычислительные системы / В.Ф. Гузик, И.А. Каляев, И.И. Левин; под общ. ред. И.А. Каляева. – Ростов-на-Дону: Изд-во ЮФУ. – 2016. – 472 с.
3. **Каляев, И.А.** Реконфигурируемые мультиконвейерные вычислительные структуры / И.А. Каляев, И.И. Левин, Е.А. Семерников, В.И. Шмойлов; под общ. ред. И.А. Каляева. – 2-е изд., переработанное и дополненное – Ростов-на-Дону: Изд-во ЮНЦ РАН. – 2009. – 344 с.

И.И. Левин, Ю.И. Доронченко, А.М. Федоров, М.К. Раскладкин

ПЕРСПЕКТИВНЫЙ СУПЕРКОМПЬЮТЕР «АРКТУР»

НИЦ суперЭВМ и нейрокомпьютеров, г. Таганрог,
levin@superevm.ru, fedorov@superevm.ru,
doronchenko@superevm.ru, raskladkin@superevm.ru

Введение

Современные вычислительно трудоемкие задачи различных предметных областей зачастую характеризуются сильной связанностью, когда количество пересылок данных между функциональными устройствами больше, чем количество таких устройств. Решение таких сильносвязанных задач с помощью вычислителей на основе универсальных процессоров крайне неэффективно и требует создания вычислительных систем не только с высокой производительностью, но и со множеством каналов высокой пропускной способности для прямого информационного обмена между вычислительными устройствами.

Существенно повышает эффективность решения научно-технических задач применение реконфигурируемых вычислительных систем, которые позволяют адаптировать свою архитектуру под структуру решаемой задачи [1,2]. Однако существующие реконфигурируемые суперкомпьютеры характеризуются мощными информационными обменами только в пределах базовых функционально законченных узлов – вычислительных модулях. Пропускная способность между платами вычислительных модулей существенно ниже, что ограничивает возможности эффективного распараллеливания вычислений в прикладных программах.

Разрабатываемый в настоящее время в НИЦ СЭ и НК (г. Таганрог) перспективный реконфигурируемый вычислительный блок (РВБ) «Арктур» на основе программируемых логических интегральных схем (ПЛИС), являющийся развитием серийно выпускаемого РВБ «Неккар» [3], будет обеспечивать реализацию крупных графов сложных задач в едином вычислительном контуре вычислительной системы, позволяя без разрывов выполнять вычисления за счет мощной системы информационного обмена.

Конструкция РВБ «Арктур»

Рассмотрим основные особенности конструкции РВБ «Арктур».

Конструктив РВБ «Арктур» 3U 19” содержит 16 вертикально расположенных на кросс-плате вычислительных модулей (ВМ) по шесть ПЛИС XCVU37P фирмы Xilinx семейства UltraScale+ в каждом.

Для общего управления, конфигурирования, контроля над вычислительным процессом используется модуль загрузки и управления (МЗУ) на основе процессора Intel Skylake.

Для электропитания вычислительных модулей разработан модуль питания DC/DC, выполняющий преобразования из входного электропитания 400 В постоянного тока в 12 В. Модуль питания четырехканальный и обеспечивает электропитание четырех ВМ. Максимальная потребляемая мощность РВБ «Арктур» – 20 кВт.

Охлаждение нагруженных электронных компонентов РВБ обеспечивается иммерсионной системой охлаждения на основе диэлектрического хладагента [4]. Хладагент обладает высокой электрической прочностью и теплопроводностью, а также максимально возможной теплоемкостью при низкой вязкости. 3D-модель РВБ «Арктур» показана на рис. 1.

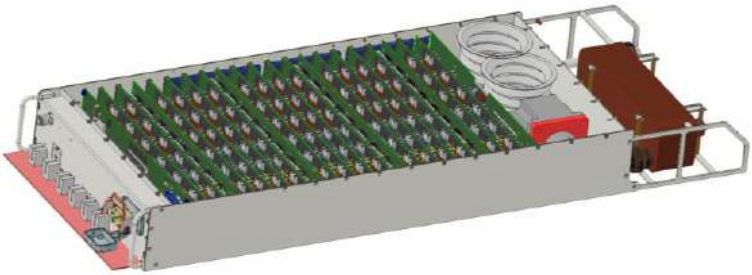


Рис. 1. 3D-модель РВБ «Арктур».

Подсистема информационного обмена

Характерной особенностью перспективного РВБ «Арктур» является подсистема информационного обмена, которая представляет собой множественные каналы связи между ПЛИС в пределах ВМ, а также между ПЛИС соседних ВМ. Связь между ПЛИС обеспечивается дифференциальными линиями с подключенными к ним мульти-гигабитными трансиверами (MGT), вспомогательными связями являются дифференциальные LVDS линии, подключенные к HP-банкам (рис. 2). Общая пропускная способность каналов связи ВМ – 15,6 Тбит/с, в том числе между ВМ – 9 Тбит/с.

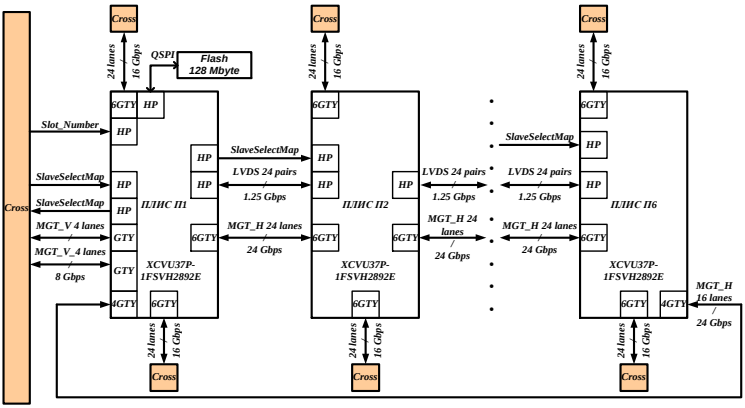


Рис. 2. Структура связей ВМ.

При построении вычислительных комплексов возможна организация информационного взаимодействия между РВБ, которое осуществляется через оптические каналы. Многоканальные оптические приемо-передатчики установлены на кросс-плате и подключаются к первому вычислительному модулю через разъемное соединение, обеспечивая пропускную способность до 4,5 Тбит/с (рис. 3).

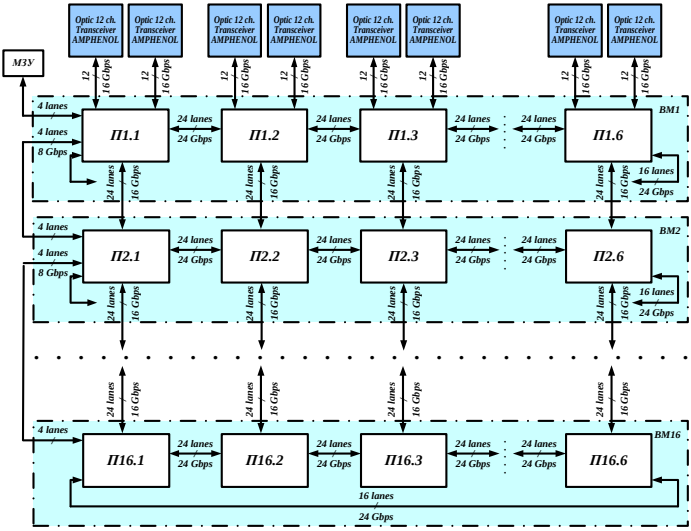


Рис. 3. Структура связей РВБ.

Работоспособность основных технических решений по организации подсистемы информационного обмена подтверждена комплексом работ по моделированию и экспериментальному исследованию на макете ВМ, в том числе с погружением макета в хладагент. Испытания, в рамках которых было передано петабайты информации по каждой дифференциальной линии, показали гарантированную передачу данных со скоростью до 24 Гбит/с между ПЛИС и до 16 Гбит/с между ВМ и между РВБ. Выработаны рекомендации по трассировке печатной платы для достижения заявленных скоростей передачи данных.

Подсистема охлаждения

Эффективность применения иммерсионной (погружной) системы жидкостного охлаждения обусловливается эффективностью технических решений при реализации каждого из компонентов: как применяемого хладагента, так и конструкции и параметров используемых радиаторов ПЛИС, насосного оборудования, теплообменников. Комплексная задача по реализации подсистемы охлаждения РВБ «Арктур», обеспечивающей отвод тепловой мощности до 20 кВт, успешно решена и апробирована на ряде макетов.

В рамках модернизации системы охлаждения решены следующие сложнейшие задачи:

– разработка новой конструкции радиатора, позволившей увеличить эффективную площадь поверхности теплообмена за счет уменьшения толщины ребер (увеличив тем самым количество каналов) и увеличения числа канавок в каналах;

– разработка погружного насоса с увеличенной производительностью до 60 л/мин, что позволит отказаться от принудительной циркуляции холодного воздуха в вычислительных стойках, значительно упростит техническое обслуживание, а также повысит надежность за счет уменьшения количества открытых гидравлических соединений;

– повышение эффективности охлаждения хладагента за счет использования нового теплообменника;

– разработка компенсатора объемного расширения хладагента в виде гофрированных резиновых рукавов, при этом полностью исключен контакт хладагента с атмосферным воздухом и минимизированы места потенциальных протечек.

Отличительной особенностью используемой ПЛИС является ее lidless-исполнение (без крышки). Предложена технология монтажа радиатора непосредственно на кристалл ПЛИС с применением современного графитового термоинтерфейса типа PSG фирмы Panasonic толщиной 0,1 мм. Исследования подтвердили преимущество данной технологии по сравнению с монтажом радиатора на крышку корпуса ПЛИС.

Также проведены значительные работы по изменению конструкции корпуса и крышки РВБ, что позволило:

– повысить жесткость конструкции и обеспечить технологичность изготовления изделия в целом;
– упростить сборку и ремонт изделия;
– минимизировать возможность протечек;
– уменьшить вес конструктива.

Использование HBM-памяти

Применяемая в изделии микросхема XCVU37P обладает новым для ПЛИС аппаратным ресурсом – встроенным модулем DDR памяти HBM (High Bandwidth Memory). Память типа HBM изготавливается в виде отдельных чипов внутри микросхемы. В ПЛИС размещается один или два чипа HBM объемом 4 или 8 ГБ каждый. В целевой ПЛИС объем HBM составляет 8 Гбайт. Пиковая пропускная способность – до 460 ГБ/с.

Подключение к контроллеру HBM осуществляется через 256-разрядные интерфейсы AXI, общее число которых достигает 32. Каждый интерфейс может работать либо со своим выделенным адресным пространством объемом 256 Мбайт, либо обращаться по любому адресу HBM через специальный коммутатор (32x32 AXI crossbar switch).

Использование ПЛИС с HBM-памятью позволяет исключить из компоновки платы отдельных микросхем памяти и упростить топологию печатной платы. Также применение данной технологии позволит существенно расширить возможности по рациональному использованию памяти в прикладных программах для решения задач обучения и математической физики.

Заключение

Перспективный РВБ «Арктур» с размещенными в его конструктиве (3U 19”) 96-ю ПЛИС высокой степени интеграции обладает уникальной плотностью компоновки вычислительного ресурса – всего более 270 млн. логических ячеек Logic Cells. При этом обеспечивается соответствующее электропитание и иммерсионное охлаждение элементов изделия при решении вычислительно трудоемких задач. Обеспечение требуемых характеристик РВБ в заданном конструктиве потребовало значительного усложнения топологии печатных плат и технологии изготовления его составных частей.

Особенностью разрабатываемого РВБ является широкие возможности информационного обмена внутри блока и между блоками для решения сильносвязанных задач. Объединение множества блоков в единый вычислительный контур позволит создавать вычислительные комплексы мирового уровня с производительностью до нескольких десятков петафлопс. Следует отметить, что такие комплексы требуют наличия соответствующей инженерной инфраструктуры.

1. **Каляев, И.А.** Реконфигурируемые вычислительные системы на основе ПЛИС / И.А. Каляев, И.И. Левин. – Ростов-на-Дону: Издательство ЮНЦ РАН, 2022. – 475 с. – ISBN 978-5-4358-0232-0.
2. **Kalyaev, I.A.** FPGA-based reconfigurable computing systems / I.A. Kalyaev, I.I. Levin, A.I. Dordopulo, L.M. Slasten // Proc. of 2013 Science and Information Conf., SAI 2013. – 2013. – P. 148–155.
3. **Каляев, И.А.** Развитие отечественных многокристальных реконфигурируемых вычислительных систем: от воздушного к жидкостному охлаждению / И.А. Каляев, А.И. Дордопуло, И.И. Левин, А.М. Федоров // Труды СПИИРАН. – 2017. – №1(50). – С. 5–31. DOI 10.15622/sp.50.1.
4. **Levin, I.I.** High-Performance Reconfigurable Computer Systems with Immersion Cooling / I.I. Levin, A.I. Dordopulo, A.M. Fedorov, Yu.I. Doronchenko // Parallel Computational Technologies (PCT 2018). 12th International Conference (Rostov-on-Don, Russia, April 2–6, 2018). – P. 62–76.

О.В. Непомнящий, А.И. Легалов, И.Н. Рыженко

МЕТОДЫ РЕДУКЦИИ ПАРАЛЛЕЛИЗМА В ПРОЦЕССЕ ВЫСОКОУРОВНЕВОГО СИНТЕЗА СБИС

Сибирский федеральный университет, г. Красноярск,
rodgi.krs@gmail.com

Введение

Рассмотрены проблемы и решения в области обеспечения архитектурной независимости и организации процесса сквозного проектирования цифровых интегральных схем. Представлен метод и язык параллельного программирования для функционально потокового синтеза проектных решений. При реализации метода выделены задачи редукции параллелизма. Предложен способ свертки (сжатия) параллелизма, базирующийся на введении дополнительного промежуточного слоя в процесс синтеза. Разработан принцип и алгоритмы редукции параллелизма.

Метод высокоуровневого синтеза

На основании ФПП – модели вычислений и поддерживающего ее языка ПИФАГОР [1] авторами предложен метод архитектурно – независимого высокоуровневого синтеза СБИС [2]. При реализации метода динамическая система типов была заменена на статическую, которая поддерживается в языках описания аппаратуры, исключены задержанные вычисления и предложен метод их преобразования при переходе на целевую платформу цифровых интегральных схем (ЦИС). Так же в модифицированный ПИФАГОР введен механизм преобразования рекурсии в итеративную схему с последующим переходом к конвейерной схеме [4].

Для обеспечения архитектурной независимости при разработке метода высокоуровневого синтеза используется промежуточное представление программы на языке ПИФАГОР в виде информационного графа (ИГ), задающего информационные связи и управляющего графа (УГ), использование которого в явном виде позволяет более детально описывать процесс управления вычислениями. ИГ формируется в ходе трансляции программы, а УГ может порождаться как динамически, как и статически. Последний способ используется для перехода от программы на ПИФАГОРе к программе описания ЦИС на HDL-языке.

При трансляции построение промежуточного представления ФПП программы осуществляется в два этапа. На первом этапе выполняется трансляция исходного текста в информационный граф. Затем по полученному информационному графу формируется управляющий граф. Управляющий граф может формироваться по информационному графу различными способами, например в соответствии с принципами управления потоками данных модели вычислений или сводиться к последовательному обходу информационного графа, либо обеспечивать другую стратегию управления вычислениями.

Для перехода на платформу ЦИС выполняется преобразование информационного графа к конвейерной схеме. Конвейерная схема вычислений представляет собой ярусно-параллельную форму информационного графа.

Для решения задачи преобразования параллелизма под конкретные ресурсные выполняются следующие шаги: определение границ изменения параллелизма, реализация алгоритма изменения параллелизма и оценка результата по ресурсным ограничениям.

Преобразование параллелизма

При переходе от неограниченного параллелизма ФПП модели на целевую платформу решается основная задача – редукция параллелизма. Это является ключевым моментом всех проводимых преобразований в архитектурно-независимом методе синтеза ЦИС. Для реализации механизма редукции был введен новый мета-слой, называемый «HDL–графом». Он представляет собой промежуточный слой для внесения изменений в модель ФПП. По мнению авторов, термин «HDL–граф» наиболее полно соответствует методу синтеза ЦИС. HDL–граф позволяет указывать связи между элементами списков связанных вершин, что в свою очередь позволяет выполнять оптимизационные преобразования при помощи вычисления операции выбора данных из списка. С помощью введенного мета-слоя при обработке типов в модели со статической типизацией вводится ограничение на динамическое изменение размерности списков во время вычислений.

Алгоритм редукции параллелизма

В процессе синтеза решаются следующие задачи:

– оценка ресурсов G_k получаемого архитектурного решения;

– оценка производительности получаемого решения;

– расчет коэффициента редукции по каждому классу ресурсов;

– редукция параллелизма схемы для достижения требуемых коэффициентов.

Для оценки ресурсов используется промежуточное представление программы – HDL–граф, в котором уже заданы архитектурно-зависимые данные. HDL–граф представляет собой ацикличный граф в ярусно-параллельной форме, в каждом узле которого заданы типы и разрядности данных [5]. В зависимости от целевой платформы определяются классы ресурсов N_k, по которым будет оцениваться схема.

Например, для платформы ПЛИС к основным классам ресурсов можно отнести:

– количество регистров N_r;

– количество логических ячеек N_{lc};

– количество блочной памяти N_m;

– количество арифметических и иных специализированных вычислительных блоков N_{dsp}.

В данном случае для свертки параллелизма ключевыми моментами являются оценка ресурсов памяти и оценка вычислительных ресурсов.

Пусть отношение ресурса, требуемого для реализации схемы в исходном максимально-параллельном виде, к доступному обозначается как R. Предполагается, что доступный ресурс – это наименьший из ресурсов. При решении задачи трансформации параллелизма в общем виде в зависимости от доступного ресурса целевой платформы можно выделить три варианта преобразования исходной максимально-параллельной схемы:

Первый вариант: если $\mathrm{R} > 1$, то необходима редукция параллелизма.

Здесь возможно увеличение производительности путем размещения нескольких схем параллельно: $\mathrm{S} = (\mathrm{Int}(1) / (\mathrm{R}))$.

Рассмотрим алгоритм редукции с использованием коэффициентов редукции. Алгоритм состоит из следующих шагов:

1. Рассчитываем коэффициенты редукции по вычислительным ресурсам $\mathrm{N_{calc}}$ и памяти $\mathrm{G_m}$;
2. Выбираем максимальный коэффициент $\mathrm{Nmax} = \max(\mathrm{Gm}, \mathrm{Ncalc})$;
3. Редуцируем параллелизм схемы на $\mathrm{N_{max}}$;
4. Для измененной схемы пересчитываем коэффициенты $\mathrm{G_m}$ и $\mathrm{N_{calc}}$, если они меньше 1 – завершение алгоритма;

5. Если какие-либо операции возможно реализовать с использованием другого типа ресурсов – изменяем данные операции на другой тип ресурсов без изменения коэффициента R и пересчитываем коэффициенты G_m и N_{calc};

6. Если какой-либо из коэффициентов больше 1: $\mathrm{Nmax} = 1 + \mathrm{Nmax}$ и возврат на шаг 3.

Последовательное увеличение коэффициента редукции позволяет подобрать минимально-возможный коэффициент для удовлетворения требований по ресурсам и достижения при этом максимальной производительности (минимально-возможной редукции по производительности относительно исходного максимально-параллельного варианта).

Второй вариант преобразования исходной максимально-параллельной схемы: при $R < 1/2$ возможно увеличение количества схем.

Третий вариант преобразования исходной максимально-параллельной схемы: при $1/2 < R < 1$, ресурса достаточно для размещения 1 максимально-параллельного варианта схемы;

Во втором и третьем случае никаких преобразований максимально-параллельной схемы не требуется.

Заключение

Алгоритм преобразования параллелизма включает оценку ресурсов, вычисление коэффициента редукции параллелизма по каждому классу ресурсов и редукцию параллелизма схемы для достижения требуемых характеристик конкретной целевой платформы.

Представленные методы редукции позволяют реализовать изменение параллелизма исходного описания алгоритма и обеспечить реализацию механизма переноса на различные архитектуры с различными ресурсными ограничениями.

В отличие от методов распараллеливания, предлагаемый в работе метод снижает сложность процесса переноса за счет уменьшения перебора количества вариантов реализации, получаемых в процессе синтеза под новые ресурсные ограничения. Вместе с тем методы оценки ресурса на высокоуровневом этапе требуют учета накладных расходов при изменении параллелизма к более последовательным схемам. Для данного случая необходимо увеличение точности оценки. Для этого могут использоваться нейронные сети и методы машинного обучения, которые на основе параметров оценки схемы на высокоуровневом этапе могут с необходимой точностью предсказать значения параметров схемы после реализации на низком уровне.

1. **Легалов, А.И.** Функциональный язык для создания архитектурно-независимых параллельных программ / А.И. Легалов // Вычислительные технологии. – 2005. – № 1 (10) – С. 71–89.

2. **Nepomnyashchy, O.V.** Methods and algorithms for a high-level synthesis of the very-large-scale integration / O.V. Nepomnyashchy, A.I. Legalov, V. Tyapkin, I.N. Ryzhenko, V.V. Shaydurov // WSEAS Transactions on Computers. – Vol. 15. – P. 239–247.
3. IEEE Std 1800-2012: IEEE Standard for SystemVerilog-Unified Hardware Design, Specification, and Verification Language, 2013.
4. **Легалов, А.И.** Методы преобразования параллелизма в процессе высокоуровневого синтеза СБИС / А.И. Легалов, О.В. Непомнящий, И.Н. Рыженко, В.В. Шайдуров // Моделирование и анализ информационных систем. – 2022. – Т. 29, № 1. – С. 60–72.
5. **Рыженко, И.Н.** Метод высокоуровневого синтеза и программный инструментарий для описания алгоритмов функционирования СБИС / О.В. Непомнящий, И.Н. Рыженко // Программная инженерия. – 2020. – № 1. – С. 34–39.

В.А. Павский[1], К.В. Павский[2,3]

МАТЕМАТИЧЕСКАЯ МОДЕЛЬ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ С НАКОПИТЕЛЕМ ДЛЯ РАСЧЕТА ПОКАЗАТЕЛЕЙ НАДЕЖНОСТИ И ОСУЩЕСТВИМОСТИ РЕШЕНИЯ ЗАДАЧ *

[1] Кемеровский государственный университет, г. Кемерово,
[2]Институт физики полупроводников им. А.В. Ржанова СО РАН,
[3]Сибирский государственный университет телекоммуникаций и информатики, г., Новосибирск,
pavva46@mail.ru, pkv@isp.nsc.ru

Введение

Рост производительности высокопроизводительных вычислительных систем связан как с развитием архитектуры вычислительных элементов, так и с масштабируемостью [1]. Например, кластерные ВС составляют 93% суперкомпьютеров в 58-й редакции (ноябрь 2021 г.) списка Top500. Число машин в подобных системах может измеряться десятками и сотнями тысяч. При таком количестве и современном уровне надежности элементной время между отказами в вычислительных системах может измеряться часами или даже минутами [2, 3], что снижает эффективность решения задач.

В работе предлагается математическая модель для расчета показателей надежности и осуществимости решения задач, в режиме обслуживания потока задач на ВС с накопителем. На рис. 1 представлена модель

*Исследования выполнены в рамках ГЗ 0242-2021-0011 и при поддержке Российского фонда фундаментальных исследований (грант № 20–07–00039).

поступающих задач в накопитель, где формируются пакеты задач определенного размера с последующей их обработкой на вычислительной системе. Необходимо оценить наполненность накопителя.

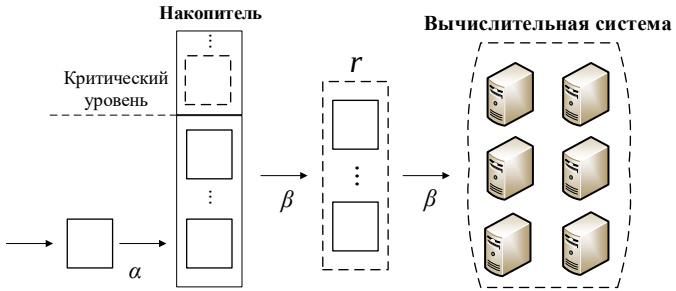


Рис. 1. Модель системы с накопителем.

Математическая модель построена в рамках теории массового обслуживания [4], где под потоком требований понимаем: в первом случае – поток поступающих задач на обслуживании; в другом случае – поток отказов элементарных машин (ЭМ) с последующим восстановлением. Рассматриваются простейшие потоки.

Математическая модель

На систему массового обслуживания (СМО) поступает пуассоновский поток требований с интенсивностью α. В каждый момент времени $t \in [0, \infty)$ СМО находится в одном из множества несовместных состояний C_k, где k – число требований в системе, включая необслуженное (см. рис.2).

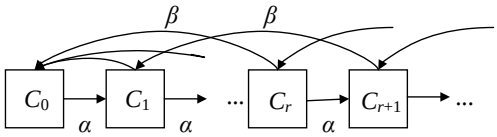


Рис. 2. Граф-схема состояний СМО.

Время обслуживания группы из r требований подчинено экспоненциальному распределению с параметром μ, $\alpha<\beta$.

Рассмотрим стационарный режим функционирования СМО. Пусть p_k – вероятность того, что СМО находится в состоянии C_k. Тогда система

алгебраических уравнений для вероятностей состояний в стационарном режиме представляется в виде

$$\begin{cases} \alpha p_0 = \beta(p_1 + p_2 + \ldots + p_r), \\ (\alpha + \beta)p_k = \beta p_{k+r} + \alpha p_{k-1}, \quad k \geq 1' \end{cases} \tag{1}$$

где $\sum_{k=0}^{\infty} p_k = 1$ – условие нормировки.

Для решения системы уравнений (1) воспользуемся аппаратом производящих функций [5]. Положим

$$F(z) = \sum_{k=0}^{\infty} p_k z^k. \tag{2}$$

Умножая уравнение k системы (1) на z^k , суммируя все уравнения и решая полученное уравнение для производящей функции (2) получаем

$$F(z) = \frac{1-1/z_0}{1-z/z_0}, \tag{3}$$

где $|z_0| > 1$ единственный корень, который является иррациональным числом, находится приближенно [4] при r>2.

Из свойств производящих функций $p_k = F^{(k)}(z)|z = 0 \ \forall k \in N$, тогда $p_0 = 1/z_0$ и общий вид для вероятностей

$$p_k = p_0(1 - p_0)^k, \quad k = 0,\ 1,\ 2,\ \ldots \tag{4}$$

На основе (3) и свойств производящих функций мат. ожидание и дисперсия числа требований в системе равно

$$(X) = \frac{1-p_0}{p_0} \quad , D(X) = \frac{1-p_0}{{p_0}^2}. \tag{5}$$

Зададим доверительную вероятность γ того, что накопитель заполнен ниже критического уровня. Тогда после определенных преобразований получаем выражение для расчета размера резерва

$$s_{\text{ср}} \geq \left[\frac{ln(1-\gamma)}{ln(1-p_0)}\right] + 1, \tag{6}$$

где $[x]$ – целая часть числа x.

Формулы (5), (6) позволяют оценить наполненность накопителя. Чтобы найти решения для (4)–(6) необходимо знать значение вероятности p_0, которое при $1 < r < \infty$ имеет иррациональное значение. Ниже

представлены оценка для p_0 и решения в частных случаях. Для нахождения оценки p_0 воспользуемся первым уравнением из (1). Тогда

$$\frac{\beta-\alpha}{\beta} \leq p_0 \leq \frac{\beta}{\alpha+\beta}. \quad (7)$$

При $r = 1$ имеем [1]

$$p_0 = \frac{\beta-\alpha}{\beta}. \quad (8)$$

При $r = 2$

$$p_0 = \frac{1}{2}\left(3 - \sqrt{1 + 4\frac{\alpha}{\beta}}\right). \quad (9)$$

При $r \to \infty$

$$p_0 = \frac{\beta}{\alpha+\beta}. \quad (10)$$

Модель вычислительной системы с резервом

На рис. 3 представлена модель вычислительной системы, состоящей из N ЭМ, с резервом. Каждая машина из основной системы отказывают с интенсивностью λ. В случае отказа ЭМ, она заменятся машиной из резерва и уходит на восстановление. Наполнение резерва выполняется группами по r ЭМ с интенсивностью μ. Оценим размер резерва, чтобы сохранить заданную производительность системы при заданных интенсивностях отказов и восстановлений элементарных машин ВС.

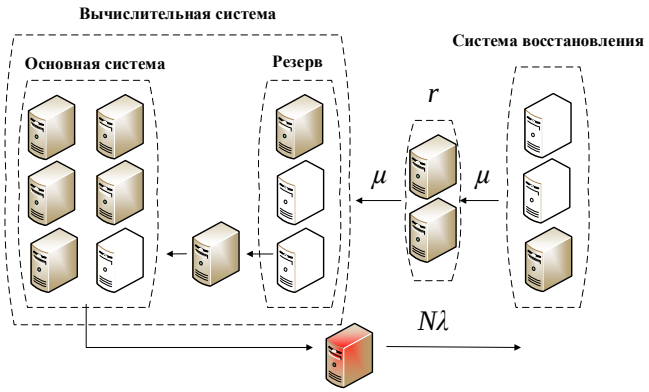


Рис. 3. Модель функционирования ВС с резервом при групповом обслуживании.

Состояние системы восстановления данной модели описывается процессом рис.2, где под требованием понимаются отказ ЭМ, а под восстановлением группы ЭМ обработанные требования пакета, при $\alpha=N\lambda$ и $\beta=\mu$. Тогда используя решения для (4)–(10), можем найти вероятности состояния системы восстановления и её наполненность и, следовательно, можем оценить состояние резерва.

Заключение

В работе рассмотрена математическая модель системы с накопителем при групповом обслуживании. Предложены аналитические решения для оценки размера накопителя, а также резерва ВС при групповом восстановлении отказавших машин. Полученные решения удобны для инженерных расчетов при экспресс-анализе функционирования масштабируемых вычислительных систем.

1. **Хорошевский, В.Г.** Архитектура вычислительных систем. / В.Г. Хорошевский. – Москва: МГТУ им. Баумана, 2008. – 520 с.
2. **Gupta, S.** Failures in large scale systems: long-term measurement, analysis, and implications / S. Gupta, T. Patel, C. Engelmann, D. Tiwari // SC '17: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. – 2017. – P. 1–12.
3. **Nikolic, S.** High Performance Computing Directions: The Drive to ExaScale Computing/ S. Nikolic // ПаВТ'2012 [электронный ресурс]. – URL: http://pavt.susu.ru/2012/talks/Nikolic.pdf
4. **Клейнрок, Л.** Теория массового обслуживания. / Л. Клейнрок. – М.: Машиностроение, 1979. – 432 с.
5. **Феллер, В.** Введение в теорию вероятностей и ее приложения: в 2-х т. – Т.1. / В. Феллер. – М: «ЛИБРОКОМ», 2010. – 528 с.

Р.Л. Смелянский

ОРГАНИЗАЦИЯ ВЫЧИСЛЕНИЙ: QUO VADIS?

Московский государственный университет,
Цент прикладных исследований компьютерных сетей, г. Москва

«Ты не можешь теперь идти за мной,
А после придёшь.»
Петр Симон Евангелие от Иоанна 13:36

Организация вычислений сегодня является одной из ключевых технологий современной цивилизации. Поэтому важно осознавать основные

тренды и перспективы ее развития, понимать проблемы, которые придётся решать. Ниже представлена краткая ретроспектива развития парадигм организации вычислений:

- конец 60-х – вычислитель с пакетной организацией вычислений;
- 70-е – вычислительный центр с mainframe и терминальной сетью;
- 80-е – клиент-серверная инфраструктура с сетевым доступом;
- 90-е – серверные фермы с Frontend сервером и локальной сетью доступа;
- 2000-е – ЦОД с высокоскоростной сетью доступа;
- н/в – периферийные вычисления (Edge Computing) – сеть мини-ЦОДов.

Что же дальше? На рубеже 70-х–80-х годов было хорошо осознано, что универсальных вычислителей не бывает. Каждый алгоритм, каждый класс методов для надлежащего уровня производительности требует определенной архитектуры вычислителя. Вычислитель должен иметь перестраиваемую архитектуру. В нашей стране первый кто это не только осознал, но и предложил решения в этом направлении был Анатолий Васильевич Каляев.

Основной движущей силой развития архитектуры вычислителя, операционной системы, средств программирования являются потребности приложений. Потребности современных приложений можно кратко охарактеризовать так:

- **распределенность** – приложение – совокупность взаимодействующих функций/сервисов, которые работают параллельно на нескольких вычислителях, интегрируя и агрегируя распределенные данные;
- **serverless** – программист/пользователь не обременен проблемами размещения, начальной загрузки и конфигурирования ресурсов и компонентов приложения;
- **эластичность** – производительность приложения можно изменять без прерывания его работы;
- **реальное время** – чувствительность к задержкам и времени отклика;
- **кросс-платформеность** – приложение не зависит от программного и аппаратного окружения;
- **взаимодействие и синхронизация** – объединение результатов разных этапов вычислений вне зависимости от места их проведения, агрегация цепочек сервисов;
- **обновления** – обновление приложения/сервисов не должно затрагивать пользователей;
- **доступность ресурсов** – все виды ресурсов доступны всегда, включая данные и код, при любых отказах физической инфраструктуры;
- **ресурсы терминала пользователя не используются** – вся обработка данных производится на удаленном оборудовании.

Организация вычислений или инфраструктура вычислений для таких приложений должны удовлетворять следующим требованиям:

- **детерминированность поведения** – предсказуемость и предопределенность;
- **безопасность** – не создает неприемлемых рисков для приложения и его данных, таких как конфиденциальность, целостность и доступность;
- **живучесть** – сервисы инфраструктуры должны быть достаточно надежными, чтобы противостоять авариям и атакам;
- **поддержка** – оборудование и услуги не должны быть сложными или чрезмерно дорогими;
- **эффективность и справедливость** – инфраструктура должна обеспечивать доставку и обработку данных приложения эффективно, надежно, не ущемляя другие приложения;
- **виртуализация** – виртуализация всех видов ресурсов (вычислительных, сетевых, хранения);
- **устойчивость** – инфраструктура должна реагировать в режиме реального времени на любые изменения в соответствии с SLA приложения;
- **доступность, надежность и отказоустойчивость** – инфраструктура обеспечивает высокий уровень доступности ее сервисов и отстраняют работоспособность в случае сбоев, с восстановлением потерянных данных;
- **масштабируемость** – производительность должна быть эффективно масштабируемой в зависимости от размещения данных, сервисов и интенсивности потоков данных;
- **serverless** – инфраструктура обеспечивает размещение компонентов приложения, гарантируя их коммуникацию и соответствие SLA.

Организация инфраструктуры вычислений с такими свойствами – это Сеть, которая представляет собой тесную интеграцию разнообразных вычислителей с высокоскоростной сетью передачи данных. Такая Сеть должна представлять собой полностью контролируемую, программируемую, виртуализированную инфраструктуру. Другими словами, Сеть становится компьютером!

Вот кратко основные составляющие организации такой Сети:

- система мониторинга и прогнозирования качества сетевого сервиса;
- системы управления опорной сетью на базе технологии программного конфигурирования;
- система управления наложенной сетью;
- система оптимального распределение ресурсов опорной сети;
- системы согласованного управления ПКС доменами транспортной сети на базе технологий ПКС/ВСС, включая базовые сервисы и функции;
- системы управления жизненным циклом сервисов в распределённых ЦОД на базе технологии ВСС;
- система мониторинга и управления функционированием сервиса;

- система оптимального размещения и миграции сервисов.

Учитывая, что диапазон масштабирования сетевого сервиса огромен и работает в режиме реальном времени, в Сети предъявляются высокие требования к временной сложности алгоритмов оптимизации. А с учетом скорости работы, как контура передачи данных, так и вычислителей, становится ясно, что в современных условиях доступны лишь субоптимальные решения методами невыпуклой оптимизации.

Для того, чтобы так организованная Сеть могла служить инфраструктурой для вычислений для приложений, в сформулированном выше смысле, ее поведение, функционирование должно удовлетворять требованиям эффективности и детерминированности, а именно:

- прогнозирование времени выполнения программы;
- возможность формировать вычислитель под приложение;
- операционная среда должна удовлетворять требованиям приложений;
- наличие разнообразных метода AI для распределения, балансировки, shaping, filtering и других способов TE на каждом hop в СПД;
- надежная изоляция контура управления и контура передачи данных от ошибок в коммутационном оборудовании, равно как изоляция разных потоков данных в этих контурах;
- Multi-Agent методы в TE и распределении заданий.
- для детерминированности потока пакетов необходимы:

– возможность устанавливать и гарантированность максимальную и минимальную сквозную задержку в сети передачи данных (СПД) и минимальные колебания джитера;

– гарантировать вероятность потери пакетов в среде передачи данных на уровне не ниже, чем в Ethernet, в беспроводке или оптике.

– максимально эффективно использовать имеющуюся пропускную способность (массовое избыточное использование ресурсов запрещено, например, как flooding);

– недопустимы задержки пакетов из-за нарушения порядка их следования, retransmission, обратная связь по перегрузке

– любая задержка передачи, вызванная СПД.

Для удовлетворения сформулированных выше требований необходимо решить широкий спектр задач. В контуре управления (КУ): определить способ организации – распределенный *vs* централизованный, выбор частота изменений и количество данных о состоянии контура данных (КД), распознавание вида потока данных, прогнозирование состояние канала, выбор оптимального канала, масштабирование КУ, масштабирование домена КД, оптимальная маршрутизация, справедливое распределение канальной емкости, балансировка потоков задач по вычислительным ресурсам, балансировка поток данных между сервисами, управление потоками данных к/от задач, выделение канальной емкости по требованию. В контуре данных – управление качеством

сервиса связи, управление перегрузками, минимизация сквозной задержки, мониторинг пропускной способности, планирование потоков по очередям.

Основной задачей управления сервисной и вычислительной сетью является обеспечение стабильного и детерминированного поведения Сети при максимальной утилизации имеющихся ресурсов. Математически эта проблема может быть сформулирована следующим образом.

Пусть $\{C_i\},\ i = 1, m$ – гетерогенная сеть высокопроизводительных вычислителей, $B = (b_{ij}),\ i = 1, m,\ j = 1, m$, где b_{ij} – пропускная способность между вычислителями C_i и C_j, $b_{ii} = 0,\ \ i = 1, m.$

$\{E_i\}, i = 1, k$ – «граничные» узлы, через которые в Сеть поступают задания P из заранее заданного множества $\{P_i\}, i = 1, n.$

Для каждого задания P известно: количество ресурсов, необходимых для выполнения задания P (ресурсы центрального процессора, оперативная память, объем дискового хранилища, количество процессов и потоков).

У каждого вычислителя C_i из множества$\{C_i\}, i = 1, m$ есть матрица M_{CP} размерности $m \times n$ оценок времени выполнения заданий P на каждом вычислителе из множества $\{C_i\}, i = 1, m.$

Необходимо предложить такой алгоритм распределения заданий P_i из множества $\{P_i\},\ i = 1, n$ по вычислителям ГСВВ, чтобы минимизировался следующий функционал:

$$\min \sum_{i}^{m} \left(\alpha \frac{c_i'}{c_i} + \beta \frac{r_i'}{r_i} \right) + \gamma \left[\left(\frac{c_i'}{c_i} - \mathbf{\Gamma} \right)^2 + \left(\frac{r_i'}{r_i} - \mathbf{\Delta} \right)^2 \right],$$

где α, β, γ – константы, c_i' – усредненное по времени количество использованного ресурса центрального процессора у вычислителя C_i, $c_i(t)$ – доступное количество ресурса центрального процессора у вычислителя C_i в момент времени t. r_i' – усредненное по времени количество использованной оперативной памяти у вычислителя C_i, $r_i(t)$ – доступное количество ресурса центрального процессора у вычислителя C_i в момент времени t, $\mathbf{\Gamma}$ – усредненное по времени количество использованного ресурса центрального процессора у всех вычислителей, $\mathbf{\Delta}$ – усредненное по времени количество использованной оперативной памяти у вычислителя.

Одной из важнейших здесь задач является задача прогнозирования времени выполнения задания на вычислителях Сети. Одним из новых подходов, на основе проведённых нами исследований является подход на основе разложения матриц. Этот подход широко применяется в рекомендательных системах и основан на технике embedding [1].

Приведенная выше формулировка проблемы распределения заданий и трафика предполагает использование техники мультиагентной оп-

тимизации (МА). Применение этой техники рассматривается в двух вариантах, которые сопоставляются с централизованным подходом. Централизованный подход предполагает, что у каждого агента знает свое локальное состояние, цент управления собирает статус каждого агента, принимает решение на основе политики оптимизации и направляет каждому агенту управляющее воздействие.

Важное значение при использовании МА-подхода в управлении имеет размер домена управления, под которым понимают подграф топологии Сети. Эксперименты применения МА-оптимизации для маршрутизатора показали, что, регулируя размер домена, можно добиться оптимального сочетания сходимости и качества оптимальности решения задачи маршрутизации.

С точки зрения организации вычислительного узла (ВУ) рассматривается, так называемый федеративный подход. Если кратко – это использование Мини-ЦОДа (mЦОД) или отдельно стоящего мощного вычислителя. Существующие подходы построения ЦОД предполагают: высокие требования к качеству сервиса каналов связи, для обеспечения доступности сервиса; очень большие затраты капитального строительства централизованного ЦОД. Существенные проблемы возникают при масштабировании ЦОД, низкий уровень утилизации ресурсов из-за отсутствия централизованной системы управления и оркестрирования.

Достоинства построения Сети на основе микро-ЦОД (mЦОД) над традиционным подходом подробно были рассмотрены в [2] характеризуется: снижением требований к транспорту за счет близости экземпляра сервиса к конечному потребителю; снижением затрат на организацию ЦОД за счет отсутствие необходимости строить централизованный ЦОД; эффективное масштабирование за счет использования централизованной облачной платформы; повышение оперативности работы сети за счет централизованной системы управления и оркестрирования и близости сервиса к клиенту. Проблемы организации контура управления и контура передачи данных для mЦОД во многом схожи с теми, что были уже перечисленных выше для КУ и КД Сети. Основная разница – скорость принятия решения должна быть на много выше.

В самом начале изложения видения того, какой должна быть инфраструктура вычислений, отмечалась необходимость динамической настройки архитектуры под алгоритм приложения. Для реализации этой потребности предлагается использовать подход дезагрегации ресурсов. Суть его заключается в следующем. В ЦОД более 20-ти лет используют модель монолитного сервера, где каждый сервер имеет материнскую плату, на которой размещены все типы аппаратных ресурсов, обычно включая процессор, микросхемы памяти, устройства хранения и сетевые карты.

Дезагрегация ресурсов центра обработки данных дезинтегрирует аппаратные ресурсы сервера на подключенные к сети автономные устройства, к

которым приложения получают удаленный доступ. Приложения должны иметь виртуализированный и защищенный доступ к аппаратным ресурсам, а центры обработки данных должны поддерживать эти приложения средствами, обеспечивающими их хорошей производительностью.

Сервер в стойке ЦОД более не рассматривается как неделимый ресурс. Стойка состоит из набора разнородных аппаратных средств обработки: CPU общего назначения, GPU, сопроцессоры на FPGA, твердотельные накопители, дисковые, ленточные подсистемы, тензорные процессоры, нейроморфные и другие типы процессоров. Все это оборудование соединено высокоскоростной сетью, например, фотонной. На сегодня коммутатор с пропускной способностью 26.5 Tbps – это реалия.

Дезагрегация ресурсов позволяет повысить эффективность использования ресурсов и производительность приложений за счет более гибкого распределения рабочих нагрузок и выбора средства обработки данных, наиболее соответствующего алгоритму обработки. Оптимальность распределения ресурсов в дезагрегированном центре обработки данных зависит от его топологии и размещения рабочих нагрузок.

В дезагрегированном ЦОД изменения оборудования в стойке должны быть прозрачными для виртуальных машин. В такой дезинтегрированной аппаратной инфраструктуре ЦОД работа гипервизора заключается в том, чтобы скрыть все нюансы работы с такой разнородной аппаратурой и обеспечить согласованную абстракцию ресурсов для виртуальных машин, используемых пользователями ЦОД. Ключевым разрешающим фактором для дезагрегации будет сеть – для поддержки хорошей производительности на уровне приложений становится критически важным, чтобы сеть обеспечивала связь с малой задержкой даже при увеличении нагрузки трафика, которую вызывает дезагрегация.

Итак, сказанное убеждает нас, что инфраструктура современного вычислителя – это Сеть, интегрирующая разнородные дезинтегрированные средства вычислителей, соединенных сетью передачи данных с программным управлением. Основными свойства которой уже были сформулированы выше.

Не рассмотренным остается вопрос об организации операционной среды – аналога традиционной ОС. Но это самостоятельная большая тема, требующая отдельной публикации.

1. **Смелянский, Р.Л.** MC2E – метаоблачная вычислительная среда для междисциплинарных исследований / Р.Л. Смелянский // Вестник Российской академии наук. – 2021. – Т. 92, № 1. – С. 57–67.
2. **Смелянский, Р.Л.** Иерархические периферийные вычисления / Р.Л. Смелянский // Моделирование и анализ информационных систем. – 2019. – Т. 26, № 1. – С. 146–169.

В.А. Сойфер

ДОСТИЖЕНИЯ И ПЕРСПЕКТИВЫ АНАЛОГОВЫХ ОПТИЧЕСКИХ ВЫЧИСЛЕНИЙ

Самарский национальный исследовательский университет имени академика С.П. Королева, г. Самара,
soifer@ssau.ru

Введение

Следует различать полностью оптический (фотонный) компьютер и гибридный электроннооптический компьютер.

Полностью оптический компьютер

Компьютер полностью на основе оптических логических элементов с преобразованием в электрический сигнал только для ввода и вывода информации.

Максимальный отказ от элементной базы на основе электронных компонент. Процессор, оперативная память, сетевые операции на оптической основе: адаптация оптических компонентов под двоичную логику, отказ от двоичной логики в пользу новой или ее подмена.

Гибридный электроннооптический компьютер

Замена части обычного компьютера на оптический аналог, выполняющий частную задачу быстрее и энергоэффективнее.

Использование оптических технологий для обмена данными.

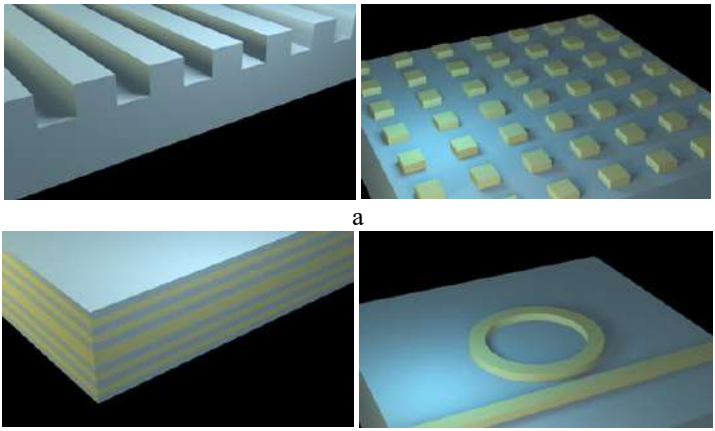
История аналоговых оптических вычислений насчитывает полвека. Достаточно вспомнить классическую работу Дж. Строука [1]. Созданные на протяжении первых 3-х десятилетий оптические процессоры не нашли широкого применения в силу их громоздкости и трудностей технической эксплуатации.

Новый импульс аналоговая оптическая обработка сигналов получила с развитием нанотехнологий и элементной базы оптоэлектроники, в т.ч. ПЗС-матриц, модуляторов света, КМОП структур и т.п.

В настоящее время существует три основных направления развития оптических аналоговых вычислений. Первый – использование планарных наноструктур для реализации интегро-дифференциальных операторов. Второй – использование пространственных модуляторов света с линзами. Третий – использование оптических нейронных сетей. Основное внимание в докладе будет уделено первому и второму направлениям. Третье направление в докладе не рассматривается.

Реализация линейных операторов на основе планарных структур нанофотоники

Базовые структуры нанофотоники для оптических вычислений показана на Рис. 1 (λ – длина волны):



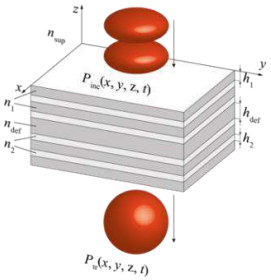
а

б в

Рис. 1. Элементная база нанофотоники: (а) одномерные и двумерные дифракционные решетки и метаповерхности (характерный период d = λ/2–λ), (б) многослойные брэгговские структуры (период d ~ λ/2), (в) микро и нанорезонаторы (радиус R ~ 50–100 λ).

Резонансные дифракционные решётки на определённых частотах могут иметь резкие изменения в спектрах пропускания и отражения (резонансы). Рассмотрим несколько примеров.

1. Временное интегрирование оптических импульсов [2]:



Геометрические параметры брегговской структуры:
h_1=153нм, h_2=260нм, h_{def}=306нм,
ε_{sup} =ε_{sub}=1, ε_1=6, ε_2=2.1, ε_{def}=6, N_L=29

Рис. 2. Огибающие падающего и прошедшего импульсов.

2. Решение обыкновенных дифференциальных уравнений с помощью кольцевых резонаторов [3]

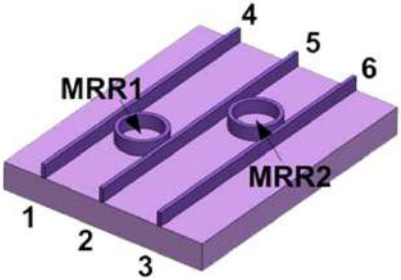


Рис. 3. Структура кольцевого резонатора.

Вход: порт 1, выход: порт 2.

$$\frac{dy_1(t)}{dt} + k_1 y_1(t) = x(t)$$

Вход: порт 5, выход: порт 6.

$$\frac{dy_2(t)}{dt} + k_2 y_2(t) = x(t)$$

Вход: порт 3, выход: порт 4.

$$\frac{d^2 y(t)}{dt^2} + a\frac{dy(t)}{dt} + b \cdot y(t) = x(t)$$

3. Пространственные преобразования трёхмерных пучков брэгговской решёткой с «дефектом» (БРД) [4].

Преобразование профиля пучка описывается дифференциальным уравнением:

$$(\Delta + p) A_{\text{refl}}(x, y) = r(\Delta + b) A_{\text{inc}}(x, y),$$

где Δ — оператор Лапласа, вычисление которого является базовой операцией при выделении контуров изображений.

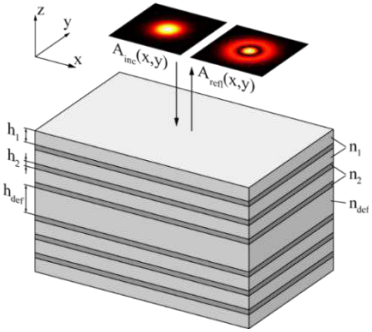


Рис. 4. Геометрия дифракции пучка на БРД.

Фотонный аналоговый сопроцессор

Предлагается схема оптического аналогового сопроцессора на основе Фурье-коррелятора с амплитудным пространственным модулятором света на входе и фазовым пространственным модулятором в частотной плоскости. В этой схеме, поступающий на камеру внешний видеопоток преобразуется амплитудным модулятором в когерентный видеопоток, а затем происходит его обработка в Фурье-корреляторе с фазовым пространственным модулятором в частотной плоскости, который и задает математическое ядро обработки. Обработанный видеопоток поступает в ЭВМ через скоростную PCIe шину [Рис. 5]. Вместо входного видеопотока сопроцессор сможет обрабатывать и последовательность искусственно сгенерированных распределений интенсивности, как матрицы с набором аналоговых сигналов. Важным достоинством предложенной схемы является относительная универсальность, т.е. широкий круг решаемых задач.

Такие системы в последние годы вновь появляются в большом количестве в публикациях, как системы быстрой обработки матричной аналоговой информации.

Основные практические решаемые задачи фотонным сопроцессором:

1. Оптическая обработка видеопотоков: выделение контуров, текстур, фильтрация шумов и т.д.
2. Распознавание объектов в видеопотоке.
3. Бортовые системы наведения.
4. Матричные операции.

Оценим производительность фотонного ускорителя.

В настоящее время существуют как модуляторы, так и камеры, позволяющие работать с большими массивами данных. Так наилучшее из существующих разрешение модулятора 4160×2464 пиксела при частоте

60Гц и относительно небольшом энергопотреблении. Наилучшее из существующих разрешение камеры 4096×4112 при той же частоте 60Гц.

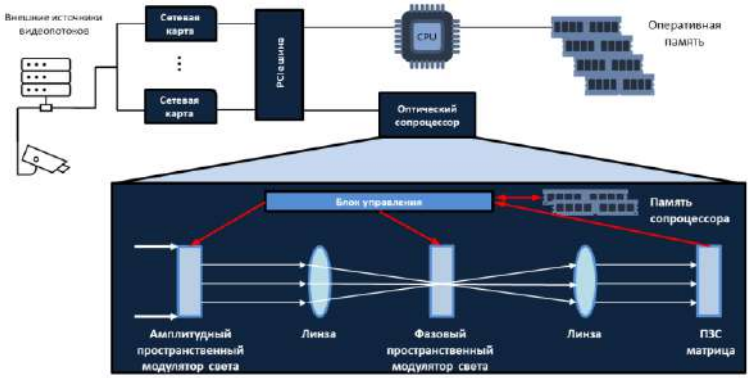


Рис. 5. Схема фотонного сопроцессора.

Эти параметры позволяют по довольно простой формуле оценить скорость обработки оптическим аналоговым сопроцессором видеопотоков. В Фурье-корреляторе происходит вычисление двух двумерных интегралов от массива размером $N \times N$, это дает нам N^4 вычислительной емкости, которая умножается на частоту и разрядность пиксела в битах. Такой расчет дает скорость вычислений 10^{17} бит/с. В перспективе, если считать, что размер матриц камер и модуляторов удвоится к 2025 году возможна скорость вычисления около 10^{19} бит/с.

Реализационно аналоговый фотонный ускоритель представляется довольно компактным (порядка 140×80×80мм) и относительно низко энергоемким (не более 100Вт).

1. **Stroke, G.W.** Optical computing / G.W. Stroke // IEEE Spectrum. – 1972. – Vol. 9. – P. 24–41.
2. **Golovastikov, N.V.** Analytical description of 3D optical pulse diffraction by a phase-shifted Bragg grating / N.V. Golovastikov, D.A. Bykov, L.L. Doskolovich, V.A. Soifer, // Opt. Express. – 2016. – Vol. 24. – P. 18828–18842.
3. **Tan, S.** High-order all-optical differential equation solver based on microring resonators / S. Tan, L. Xiang, J. Zou [et al.] // Opt. Lett. – 2013. – Vol. 38. – P. 3735–3738.
4. **Bykov, D.A.** Optical computation of the Laplace operator using phase-shifted Bragg grating / D.A. Bykov, L.L. Doskolovich, E.A. Bezus, V.A. Soifer // Opt. Express. – 2014. – Vol. 22. – P. 25084–25092.

С.А. Степаненко

ФОТОННАЯ ВЫЧИСЛИТЕЛЬНАЯ МАШИНА. ОПТИЧЕСКИЕ ЛОГИЧЕСКИЕ ЭЛЕМЕНТЫ

Российский федеральный ядерный центр –
Всероссийский НИИ экспериментальной физики, г. Саров

Введение

Возможность и целесообразность реализации фотонной вычислительной машины [1] определяются, в частности, параметрами оптических логических элементов (ОЛЭ).

Активные ОЛЭ из известных материалов, задействующие нелинейные эффекты, уступают по производительности и энергоэффективности современным электронным элементам [2].

Пассивные ОЛЭ [3-7] имеют малые длительности выполнения операции, однако характеризуются большими потерями энергии и трудностями сохранения параметров сигнала.

Предлагаемые ниже интерференционные ОЛЭ задействуют различие значений интенсивности света, возникающее в результате отсутствия или наличия интерференции световых волн и их взаимодействия с волноводами. Показано применение этих ОЛЭ для создания фотонной вычислительной машины и других, конкурентных с электронными аналогами цифровых устройств.

1. Идентификация логических констант

ОЛЭ выполняют логические операции над световыми импульсами, образованными электромагнитными волнами. В результате операций получаются логические константы – логический нуль («0») и логическая единица («1»). Они идентифицируются интенсивностью I электромагнитной волны [8], распространяемой в оптическом волноводе [9].

Электромагнитная волна – линейно поляризованная монохроматическая [8], далее – бегущая волна, представляется в форме светового импульса длительностью $\tau = m\lambda/\upsilon$, где m – целое число, λ – длина волны [8], υ – скорость света в волноводе. Величина $d = m\lambda$ называется размером импульса.

Значение интенсивности выходного сигнала ОЛЭ вырабатывается либо при отсутствии входных импульсов – в ОЛЭ не задействуется никаких эффектов, либо в случае одного входного импульса – задействуются эффекты, порожденные бегущей волной, либо в случае двух входных встречных импульсов – задействуются эффекты, порожденные стоячей волной [8].

Чтобы выполнять логические операции необходимо перераспределять интенсивность в волноводах. Для этого применяются щели. Щелью [8] называется часть поверхности волновода, через которую излучение передается в другой волновод или в окружающую среду. Для отвода энергии из щели используется однонаправленный типа (1×2) ответвитель [9] с заданным коэффициентом ответвления $0 < q < 1$, означающим долю интенсивности, отводимую через щель в ответвительный волновод.

Остаток энергии в волноводе, имеющем m щелей, после прохождения бегущей волны пропорционален величине $I_m = I_0(1-q)^m$, где I_0 – интенсивность волны на входе в волновод.

Согласно закону сохранения, количество энергии, выделенной из волновода в результате взаимодействия бегущей волны с m щелями, пропорционально величине $\overset{\Delta}{I}_m = I_0 - I_0(1-q)^m$.

В результате столкновения когерентных импульсов длительностью τ образуется стоячая волна с $2m$ пучностями на расстоянии $\lambda/2$ одна от другой и $2m$ узлами, также удаленными на $\lambda/2$ один от другого [8]. Если щели волновода расположены над пучностями, то суммарная интенсивность на обоих выходах волновода составит $I_\Sigma = 4I_0(1-q)^{2m}$; при этом, суммарная интенсивность, выделяемая в 2m щелей волновода, равна $\overset{\Delta}{I}_{2m} = 2I_0 - 4I_0(1-q)^{2m}$. Если щели волновода расположены над узлами, то $I_\Sigma = 2I_0(1-q)^m$; при этом, $\overset{\Delta}{I}_{2m} = 2I_0 - 2I_0(1-q)^m$.

Элементы, в волноводах которых щели расположены над пучностями либо над узлами стоячей волны, называем ОЛЭ первого типа и соответственно ОЛЭ второго типа.

2. Функционально полный базис

Применим рассмотренные эффекты для создания функционально полного базиса ОЛЭ из элементов «И», «исключающее ИЛИ» и «НЕ», реализующих функции &, $\oplus$, $\overline{}$ [11].

В рассматриваемых далее ОЛЭ используется волноводная структура, показанная на рисунке 1.

Назначение ее элементов охарактеризуем согласно [12].

Поступающий в волновод на вход 1 импульс с поляризацией 0° изменяет после левостороннего вращателя $R\overrightarrow{45}(-)$ плоскость поляризации на 45° по правилу левого винта и через поляризационное зеркало $M1_1$ поступает в волновод. После фарадеевского вращателя $F\overrightarrow{45}(+)$ он восстанавливает плоскость поляризации и взаимодействует со щелями и (при наличии) со встречным импульсом, поступившим после аналогичных операций со входа 2. Оба импульса когерентны, имеют одинаковую поляризацию. В результате их столкновения образуются пучности и узлы.

Над пучностями (либо над узлами) в той же плоскости расположены щели и двунаправленные ответвительные волноводы. Устройство такого волновода показано на вставке к рис.1.

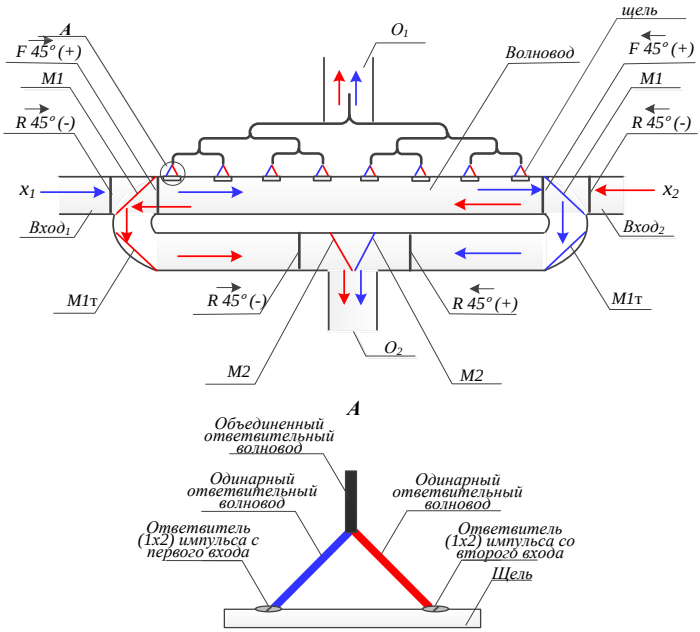


Рис. 1. Волноводная структура. Элементы конструкции.

После взаимодействия со щелями оставшаяся часть импульса со входа 1 поступает на фарадеевский вращатель $F\overleftarrow{45}(-)$, меняет плоскость поляризации и отражается поляризационными зеркалами $M1_1$, $M1_2$ на вращатель $R\overleftarrow{45}(+)$, восстанавливающий исходную поляризацию. Далее отраженный зеркалом M2 импульс поступает на выход O_1, где объединяется (при наличии) с импульсом со входа 2, над которым выполнялись такие же операции.

На выход O_2 поступает энергия из ответвительных волноводов, получаемая взаимодействием импульсов со щелями волновода.

2.1. Элементы первого типа

Элемент «И». Пусть в одном из направлений, обозначенных на рис.1 символами x_1 и x_2, распространяется импульс размером $m\lambda$ (это означает $x_2 = 1,\ x_1 = 0$ или $x_2 = 0,\ x_1 = 1$). На выходе O_1 элемента имеем $I_{\&} = I_0(1 - q_1)^{2m}$; это соответствует $f_{\&}(x_2, x_1) = 0$.

Если импульсы распространяются в обоих направлениях (это означает $x_2 = 1,\ x_1 = 1$), то образуется стоячая волна. На выходе O_1 имеем $I_{\&} = 4I_0(1-q_1)^{2m}$; это соответствует $f_{\&}(x_2, x_1) = 1$.

Элемент «ИСКЛЮЧАЮЩЕЕ ИЛИ». Выход элемента обозначен O_2, он объединяет волноводы от 2m щелей над пучностями стоячей волны. В случае одного импульса на O_2 имеем $I_{\oplus} = I_0 - I_0(1-q_2)^{2m}$, что соответствует $f_{\oplus}(x_2, x_1) = 1$. Для двух встречных импульсов на O_2 имеем, $I_{\oplus} = 2I_0 - 4I_0(1-q_2)^{2m}$, что соответствует $f_{\oplus}(x_2, x_1) = 0$.

Элемент «НЕ». Реализация функции $\overline{}$ («НЕ») достигается подачей на один из входов элемента $\oplus$ единичного сигнала – логической единицы. Значения функций $f_{\&}(x_2, x_1) = x_2 \& x_1$, $f_{\oplus}(x_2, x_1) = x_2 \oplus x_1$ и соответствующие значения $I_{\&}$, $I_{\oplus}$ представлены в таблице 1.

Таблица 1

Значения $f_{\&}(x_2, x_1)$, $f_{\oplus}(x_2, x_1)$ и $I_{\&}$, $I_{\oplus}$. Элементы первого типа

x_2	x_1	$f_{\&}$	$I_{\&}$	$f_{\oplus}$	$I_{\oplus}$
0	0	0	0	0	0
0	1	0	$I_0(1-q_1)^{2m}$	1	$I_0 - I_0(1-q_2)^{2m}$
1	0	0	$I_0(1-q_1)^{2m}$	1	$I_0 - I_0(1-q_2)^{2m}$
1	1	1	$4I_0(1-q_1)^{2m}$	0	$2I_0 - 4I_0(1-q_2)^{2m}$

Идентичность значений интенсивности. Для корректного функционирования устройств из ОЛЭ необходимо, чтобы одинаковым логическим значениям, получаемым через одинаковые временные интервалы, на выходах ОЛЭ, реализующих &, $\oplus$ и $\overline{}$, соответствовали одинаковые интенсивности $\boldsymbol{I}_{\&}$, $\boldsymbol{I}_{\oplus}$. Необходимо выполнение:

$$\begin{cases} 4I_0(1-q_1)^{2m} = I_0 - I_0(1-q_2)^{2m}, \\ I_0(1-q_1)^{2m} = 2I_0 - 4I_0(1-q_2)^{2m}. \end{cases}$$

Решением этой системы уравнений является:

$(1-q_1)^{2m} = 0{,}133$ и $(1-q_2)^{2m} = 0{,}466$.

Значения m задаются требуемой длительностью операции.

Получен функционально полный базис интерференционных ОЛЭ первого типа.

2.2. Элементы второго типа

Структура ОЛЭ второго типа показана на рис. 2, выходы обозначены O_1 и O_2. Перед каждым входным вращателем устроены $(2m - \frac{k_1}{2})$ щелей. Между вращателями устроены $k_1 < 2m$ щелей, расположенные над узлами. Остальные конструкционные элементы совпадают с указанными на рис. 1.

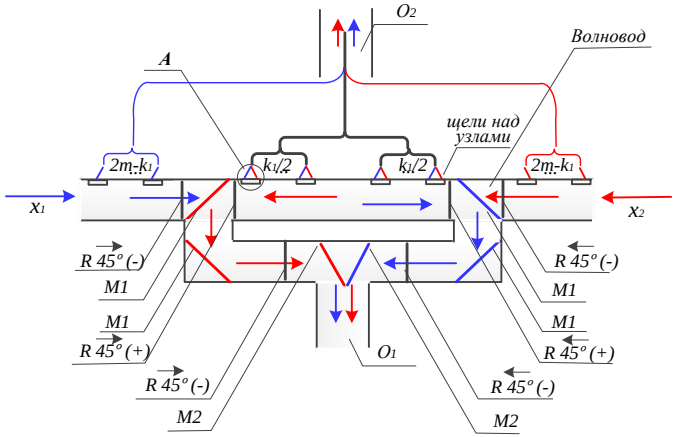


Рис. 2. Структура ОЛЭ второго типа.

Элемент «И». Одиночный импульс, означающий $x_2 = 1$, $x_1 = 0$ или $x_2 = 0$, $x_1 = 1$, взаимодействует с $2m$ щелями. На выходе O_1 имеем $I_\& = I_0(1 - q_1)^{2m}$.

Если импульсы распространяются в обоих направлениях (т.е. $x_2 = 1$, $x_1 = 1$), то перед столкновением импульсов и образованием стоячей волны интенсивность в каждом направлении равна $I_0(1 - q_1)^{2m - k_1/2}$. После столкновения интенсивность в щели не выделяется, т.к. щели расположены над узлами. На выходе O_1 имеем $I_\& = 2I_0(1 - q_1)^{2m - k_1/2}$.

Элемент «ИСКЛЮЧАЮЩЕЕ ИЛИ». Выход элемента обозначен O_2, он объединяет ответвительные волноводы. В случае одного импульса имеем $I_\oplus = I_0 - I_0(1 - q_2)^{2m}$, что соответствует $f_\oplus(x_2, x_1) = 1$. Для двух встречных импульсов имеем, $I_\oplus = 2I_0 - 2I_0(1 - q_2)^{2m}$, что соответствует $f_\oplus(x_2, x_1) = 0$.

Элемент «НЕ». Реализация функции $\bar{\ }$ («НЕ») достигается подачей на один из входов элемента $\oplus$ постоянного единичного сигнала – логической единицы.

Значения функций $f_\&(x_2, x_1) = x_2 \& x_1$, $f_\oplus(x_2, x_1) = x_2 \oplus x_1$ и соответствующие значения $I_\&$, $I_\oplus$ представлены в таблице 2.

Таблица 2

Значения $f_{\&}(x_2, x_1)$, $f_{\oplus}(x_2, x_1)$ и $I_{\&}$, $I_{\oplus}$. Элементы второго типа

x_2	x_1	$f_{\&}$	$I_{\&}$	$f_{\oplus}$	$I_{\oplus}$
0	0	0	0	0	0
0	1	0	$I_0(1-q_1)^{2m}$	1	$I_0 - I_0(1-q_2)^{2m}$
1	0	0	$I_0(1-q_1)^{2m}$	1	$I_0 - I_0(1-q_2)^{2m}$
1	1	1	$2I_0(1-q_1)^{2m-k_1/2}$	0	$2I_0 - 2I_0(1-q_2)^{2m-k_2/2}$

Идентичность значений интенсивности. Потребуем равенства значений $I_{\&}$ и $I_{\oplus}$, соответствующих одинаковым логическим константам:

$$\begin{cases} 2I_0(1-q_1)^{2m-\frac{k_1}{2}} = I_0 - I_0(1-q_2)^{2m}; \\ I_0(1-q_1)^{2m} = 2I_0 - 2I_0(1-q_2)^{2m-\frac{k_2}{2}}. \end{cases}$$

Решения этой системы уравнений получаются численными методами. В частности, решением являются $2m = 6$, $k_1 = k_2 = 4$, $q_1 \approx 0{,}15$ и $q_2 = 0{,}1$. Значения m определяются требуемой длительностью операции.

Получен функционально полный базис интерференционных ОЛЭ второго типа.

2.3. Y-волновые элементы

Импульсы, имеющие различные λ_i, не взаимодействуют.

Структуры ОЛЭ, одновременно и независимо выполняющие операции над данными, представленными на различных λ, получаются соответствующим размещением щелей и узлов. Параметры ОЛЭ первого типа и ОЛЭ второго типа, обрабатывающих длины λ_i, $i = 1,2,\dots,y$ вычисляются соответственно из:

$$\begin{cases} 4I_0(1-q_1)^{my+m} = I_0 - I_0(1-q_2)^{2my}; \\ I_0(1-q_1)^{2my} = 2I_0 - 4I_0(1-q_2)^{my+m}; \end{cases}$$

и

$$\begin{cases} 2I_0(1-q_1)^{2m-\frac{k_1}{2}y} = I_0 - I_0(1-q_2)^{2m}; \\ I_0(1-q_1)^{2m} = 2I_0 - 2I_0(1-q_2)^{2m-\frac{k_2}{2}y}. \end{cases}$$

3. Масштабирование интенсивности

Оценим возможности увеличения различия значений интенсивности, соответствующих логическим «0» и логическим «1».

J-уровневые структуры элементов. Волновод, в который входные импульсы поступают извне элемента, называем волноводом первого уровня. Волновод, в который входные импульсы поступают из волновода k-го уровня, где $k = 1,2, \ldots$, называем волноводом $(k+1)$ уровня.

Введем, как показано на рис. 3, в ОЛЭ первого типа еще один волновод – волновод второго уровня. Эту структуру назовем двухуровневым ОЛЭ первого типа.

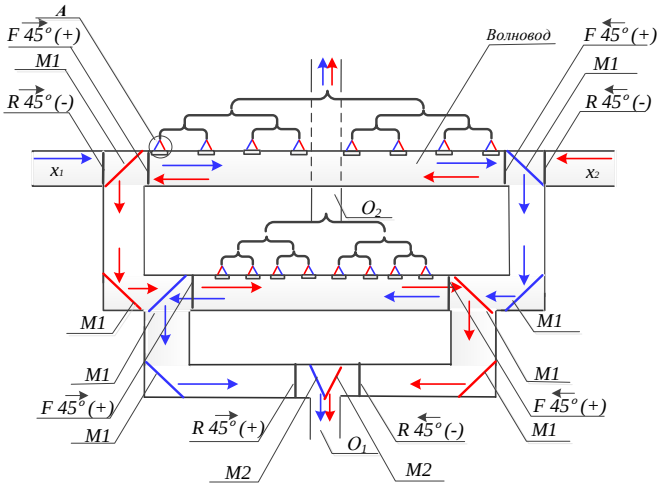


Рис. 3. Структура двухуровневого ОЛЭ первого типа.

Назначения и обозначения конструкционных элементов, совпадают с указанными в 2.1.

В волноводе второго уровня ОЛЭ, реализующего &, устроены $2m$ щелей в местах пучностей, образуемых импульсами, поступающими из волновода первого уровня. Выход элемента обозначен O_1.

В случае $x_1 = 1$, $x_2 = 0$ или $x_1 = 0$, $x_2 = 1$ имеем одиночный импульс, интенсивность которого после волновода первого уровня равна $I_{2m} = I_0(1-q_1)^{2m}$. В результате взаимодействия с $2m$ щелями волновода второго уровня его интенсивность составит $I_{\&} = I_{4m} = I_{2m}(1-q_1)^{2m} = I_0(1-q_1)^{4m}$. Общее энерговыделение в щели волноводов первого и второго уровней пропорционально $\overset{\Delta}{I}_{4m} = I_0 - I_0(1-q_1)^{4m}$.

В случае $x_1 = 1$, $x_2 = 1$ имеем на входах волновода второго уровня два встречных импульса; интенсивность каждого $I_{2m} = 2I_0(1-q_1)^{2m}$. В результате их столкновения суммарное значение интенсивности на выходе O_1, соответствующее логической единице равно $I_{\&} = 8I_0(1-q_1)^{4m}$. Общее энерговыделение в щели пропорционально $\overset{\Delta}{I}_{4m} = 2I_0 - 8I_0(1-q_1)^{4m}$.

Структура двухуровневого ОЛЭ, реализующего $\oplus$, представлена на рис. 3. Выход обозначен O_2. Значения $I_{\oplus}$ получаются аналогично вышеизложенному.

Элемент «НЕ» получается из ОЛЭ, реализующего $\oplus$, подачей на один из входов сигнала логической «1». Длительности выполнения операций «$\bar{\ }$» и $\oplus$ совпадают.

Значения функций $f_{\&}(x_2, x_1) = x_2 \& x_1$, $f_{\oplus}(x_2, x_1) = x_2 \oplus x_1$ и соответствующие им значения $I_{\&}$, $I_{\oplus}$, получаемые на выходах двухуровневых ОЛЭ первого типа, представлены в табл. 3.

Таблица 3

Значения $f_{\&}(x_2, x_1)$, $f_{\oplus}(x_2, x_1)$ и $I_{\&}$, $I_{\oplus}$. Двухуровневые ОЛЭ первого типа

x_2	x_1	$f_{\&}$	$I_{\&}$	$f_{\oplus}$	$I_{\oplus}$
0	0	0	0	0	0
0	1	0	$I_0(1-q_1)^{4m}$	1	$I_0(1-q_2)^{2m} - I_0(1-q_2)^{4m}$
1	0	0	$I_0(1-q_1)^{4m}$	1	$I_0(1-q_2)^{2m} - I_0(1-q_2)^{4m}$
1	1	1	$8I_0(1-q_1)^{4m}$	0	$2I_0(1-q_2)^{2m} - 8I_0(1-q_2)^{4m}$

Одинаковым логическим константам должны соответствовать одинаковые интенсивности на выходах ОЛЭ. Необходимо:

$$\begin{cases} I_0(1-q_1)^{4m} = 2I_0(1-q_2)^{2m} - 8I_0(1-q_2)^{4m}, \\ 8I_0(1-q_1)^{4m} = I_0(1-q_2)^{2m} - I_0(1-q_2)^{4m}. \end{cases}$$

Находим: $(1-q_1)^{2m} \approx 0{,}15$ и $(1-q_2)^{2m} = 0{,}238$.

Здесь различие интенсивностей логической «1» и логического «0» в 2 раза больше, чем в одноуровневом ОЛЭ.

Приведенные соотношения обобщаются на J уровней волноводов, где $J > 1$, целое. Получаем:

$$\begin{cases} 2 \cdot 2^J I_0(1-q_1)^{2mJ} = I_0(1-q_2)^{2m(J-1)} - I_0(1-q_2)^{2mJ}, \\ I_0(1-q_1)^{2mJ} = 2I_0(1-q_2)^{2m(J-1)} - 2 \cdot 2^J I_0(1-q_2)^{2mJ}. \end{cases}$$

При однократном срабатывании, соответствующие логическому «0» и логической «1», значения интенсивности в J-уровневых ОЛЭ отличаются в $2 \cdot 2^J$ раз. Длительность выполнения операции составляет не менее τJ, где τ – длительность выполнения операции одноуровневым элементом.

Динамика значений интенсивности. В результате последовательных операций значения интенсивностей, соответствующие различным логическим константам, могут сближаться. Это обусловлено различием значений выходной интенсивности при подаче на оба входа «1» и подачей на один вход «1», а на другой – «0». Оценим границы значений интенсивности.

Пусть с выхода i-го элемента импульс поступает на вход $(i+1)$ элемента, где $i = 1,2,\dots,(h-1)$. Обозначим значения интенсивностей «1» и «0» после h операций на выходе h-го элемента символами $\widetilde{I_1^{(h)}}$ и $\widetilde{I_0^{(h)}}$. Начальное входное значение, соответствующее логической «1», обозначим $\widetilde{I_1^{(0)}}$. Для простоты изложения используем J уровневый элемент «И» первого типа.

В общем виде имеем:

$$\widetilde{I_1^{(h)}} = (2 \cdot 2^J (1-q_1)^{2mJ})^h \cdot \widetilde{I_1^{(0)}},$$
$$\widetilde{I_0^{(h)}} = (1-q_1)^{2mJ} \left(\sqrt{\tilde{I}_0^{(h-1)}} + \sqrt{\tilde{I}_1^{(h-1)}} \right)^2.$$

Подставляя значения интенсивностей, получаем:

$$2^{J-1} < \frac{\widetilde{I_1^{(h)}}}{\widetilde{I_0^{(h)}}} < 2 \cdot 2^J.$$

Это справедливо и для ОЛЭ «исключающее ИЛИ» и «НЕ».

Для J-уровневых ОЛЭ второго типа имеем:

$$\begin{cases} 2 \cdot I_0(1-q_1)^{(2m-\frac{k_1}{2})J} = I_0(1-q_2)^{2m(J-1)} - I_0(1-q_2)^{2mJ}, \\ I_0(1-q_1)^{2mJ} = 2I_0(1-q_2)^{(2m-\frac{k_2}{2})(J-1)} - 2 \cdot I_0(1-q_2)^{(2m-\frac{k_2}{2})J}. \end{cases}$$

Динамические значения интенсивностей удовлетворяют условию:

$$\frac{1}{2(1-q_1)^{\frac{k_1}{2}J}} < \frac{\widetilde{I_1^{(h)}}}{\widetilde{I_0^{(h)}}} < \frac{2}{(1-q_1)^{\frac{k_1}{2}J}}.$$

4. Пример реализации и применения

Для реализации ОЛЭ могут использоваться оптоволоконные [10], либо планарно-интегральные [9] волноводы, фотонные кристаллы (ФК) [12], графеновые структуры [5]. Для локализации интенсивности в пучностях и узлах необходимо, чтобы размер щели составлял $\sim 0{,}1\lambda$.

Для $\lambda = 1{,}55$ мкм этому условию удовлетворяет ФК из кремния [4]. Оценим параметры ОЛЭ из этого ФК.

Значения $m = 4$ и $J = 2$ задаем из требований к быстродействию и различию интенсивностей. Из условий идентичности вычисляем $q_1 = 0{,}22$ и $q_2 = 0{,}15$.

Щели реализуются линейными дефектами в ФК [4]. Из графика относительных значений интенсивности [4] находим, что $q_1 = 0{,}22$ достигается при $r_d/p = 0{,}38$, где r_d – радиус стержня дефекта; $p = 0{,}418$ мкм – постоянная решетки кремния. Получаем $r_d \approx 160$ нм. Аналогично находим, что $q_2 = 0{,}15$ достигается при $r_d/p = 0{,}35$. Получаем $r_d \approx 120$ нм. Изготовление таких дефектов освоено [13].

Чтобы импульсы в ответвительных волноводах не взаимодействовали между собой, расстояние между центрами щелей выберем $8p = 3{,}3$ мкм [4]. Размещение 8 щелей требует 26 мкм. Фарадеевские вращатели в ФК требуют 1,5 мкм, другие конструкционные элементы около 1,0 мкм [13]. Длину волновода принимаем L=75 мкм. Полагая скорость света в волноводе $\upsilon = 2 \cdot 10^8$ м/с, находим $\tau = L/\upsilon = 375 \cdot 10^{-15}$ с длительность выполнения операции.

Периодичность регенерации. В процессе выполнения логических операций интенсивность уменьшается. Критичным является уровень интенсивности, соответствующий логической «1», определяющий наличие сигнала. Логический «0» может быть как угодно близок к физическому нулю.

Количество энергии в ОЛЭ из кремния при $\lambda = 1{,}55$ мкм и длительности выполнения операции $\tau = 375 \cdot 10^{-15}$ с составит $\mathrm{E} = IS\tau \approx 1{,}23 \cdot 10^{-11}$ Дж, где $I = \frac{\varepsilon_{\text{разр}}}{\tau} = 0{,}67 \cdot 10^{13}$ Вт/см^2 – пороговое значение интенсивности, превышение которого разрушает волновод, $\varepsilon_{\text{разр}} = 2{,}5$Дж/см^2 – лучевая прочность кремния при длительности импульса $\tau = 300 - 400 \cdot 10^{-15}$ с [14], $S \approx 4{,}9 \cdot 10^{-12}$ м^2 – площадь светового пятна.

Энергия фотона при $\lambda = 1{,}5$ мкм составляет: $\mathrm{E_ф} \approx 13{,}2 \cdot 10^{-20}$ Дж [8]. Количество фотонов в импульсе: $\mathrm{E/E_ф} \approx 0{,}09 \cdot 10^9$ штук. Для надежной идентификации импульса при $\mathrm{T} = 300\mathrm{K}$ достаточно 10^3 фотонов [12].

Мощность импульса $13{,}2 \cdot 10^{-20}$Дж $\cdot 10^{3}/375 \cdot 10^{-15} \approx 0{,}34$ мВт. Бюджет мощности 38 дБ.

Выполнение одной логической операции уменьшает мощность сигнала не более, чем на 5 дБ.

Без регенерации можно выполнить ≈ 7 шт. операций, в течение $2{,}5 \cdot 10^{-12}$ с. Для регенерации применяются нелинейные оптические ключи пикосекундного диапазона [15]. Их задействование не скажется на производительности и энергопотреблении устройства.

В процессе выполнения операций импульсы передаются только в волноводах, когерентность сохраняется. Энергия остается в ОЛЭ и задействуется для усиления и регенерации сигнала.

Общие потери энергии определяются потерями в волноводах, эта величина составляет 0,1 дБ/см, и эффективностью ввода излучения в волновод $\mu \geq 0{,}9$ [2].

Фотонная вычислительная машина, реализованная из предложенных ОЛЭ, при $\lambda = 1{,}5$ мкм, длине волновода L=75 мкм и указанных значениях энергетических параметров имеет в 10^3–10^4 раз большую производительность, по сравнению с электронными аналогами при одинаковых энергозатратах [1].

Заключение

Предложены структуры интерференционных логических элементов, образующих полный функциональный базис. Выполняются требования идентичности значений интенсивности, соответствующих логическим константам «0» и «1», вырабатываемых различными элементами в заданные фиксированные интервалы времени.

Масштабирование значений интенсивности логически констант достигается увеличением количества уровней волноводов и сопровождается пропорциональным увеличением длительности выполнения операций.

На примере цифровой фотонной вычислительной машины показаны целесообразность и перспективность применения предложенных элементов.

1. **Степаненко, С.А.** Фотонная вычислительная машина. Принципы реализации. Оценки параметров / С.А. Степаненко // Доклады академии наук. – 2017. – Т. 476, № 4. – С. 389–394.
2. **Белов, П.А.** Оптические процессоры: достижения и новые идеи / П.А. Белов, В.Г. Беспалов, В.Н. Васильев [и др.] [электронный ресурс]. – URL: http://pandia.ru/77/192/21190.php
3. **Полуэктов, А.О.** Оптический логический элемент. Патент РФ 20751061. Опубликован 10.03.1997 г.
4. **Сяо-Вэнь, С.** Разработка и анализ логических элементов «ИЛИ-НЕ», «И-НЕ» и «исключающее ИЛИ-НЕ» на основе эффекта интерференции / С. Сяо-Вэнь, Я. Сю-Лунь, М. Сян-Фэн [и др.] // Квантовая электроника. – 2018. – Т. 48, № 2 – С. 178–183.

5. **Wu, X.** A type of all-optical logic gate base on graphene surface plasmon polaritons / X. Wu, J. Tian, R. Yang // Optics Communications. –2017. – Vol. 403. – P. 185–192. DOI: 10.1016/j.optcom.2017.07.041.
6. **Papaioannou, M.** All-optical multichannel logic based on coherent perfect absorption in a plasmonic metamaterial / M. Papaioannou, E. Plum, J. Valente [et al.] // APL PHOTONICS. – 2016. – Vol. 1. – 090801. DOI: 10.1063/1.4966269.
7. **Hussein, M.E.** New design of a complete set of Photonic Crystals logic gates / M.E. Hussein, A.Ali Tamer, N.H. Rafab // Optics Communications. – 2018. – Vol. 411. – P. 175–181.
8. **Яворский, Б.М.** Справочник по физике для инженеров и студентов вузов / Б.М. Яворский, А.А. Детлаф, А.К. Лебедев. – М.: ООО «Издательство Оникс»: ООО «Издательство «Мир и Образование», 2008. – 1056 с.
9. **Никаноров, Н.В.** Волноводная фотоника / Н.В. Никаноров, С.М. Шандаров. – Санкт-Петербург: СПбГУ ИТМО, 2008. – 143 с.
10. **Вербовецкий, А.А.** Основы проектирования оптоэлектронных систем связи. / А.А. Вербовецкий. – М.: Радио и связь, 2000. – 160 с.
11. **Поспелов, Д.А.** Логические методы анализа и синтеза схем. / Д.А. Поспелов. – М.: Энергия, 1974. – 368 с.
12. **Салех, Б.** Оптика и фотоника. Принципы и применения. Т. 1, 2. / Б. Салех, М. Тейх. – Долгопрудный: «Интеллект», 2012.
13. **Котляр, В.В.** Нанофотоника – манипулирование светом с помощью наноструктур / В.В. Котляр // Компьютерная оптика. – 2008. – Т. 32, № 2. – С. 119–135.
14. **Gallais, L.** Wavelength dependence of femtosecond laser-induced damage threshold of optical materials / L. Gallais, D.-B. Douti, M. Commandre [et al.] // Journal of Applied Physics. – 2015. – Vol. 117. – 223103. DOI: 10.1063/1.4922353.
15. **Гончарова, О.В.** Оптический переключающий элемент. / О.В. Гончарова, А.В. Демин // Патент РФ 2096815, опубликован 20.11.1997.

Р.М. Шагалиев

НАПРАВЛЕНИЯ РАБОТ И АКТУАЛЬНЫЕ ЗАДАЧИ РАЗВИТИЯ СУПЕРКОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ, ВКЛЮЧАЯ ВЫСОКОПРОИЗВОДИТЕЛЬНУЮ ВЫЧИСЛИТЕЛЬНУЮ БАЗУ

Российский федеральный ядерный центр –
Всероссийский НИИ экспериментальной физики, г. Саров,
RMShagaliev@vniief.ru

В настоящее время роль суперкомпьютерного моделирования при проектировании изделий машиностроения сложно переоценить. Мировой опыт показывает, что высокопроизводительные вычисления применяются в различных масштабах: от моделирования отдельных узлов изделий до полномасштабного комплексного моделирования при производстве высокотехнологичной продукции.

Создание программного обеспечения для суперкомпьютерного моделирования является комплексной задачей и включает в себя: разработку базового программного обеспечения для моделирования основных физических процессов, специализированных программных модулей, предназначенных для моделирования специфичных физических процессов, характерных для функционирования конкретных классов изделий, единых баз данных для верификации и валидации прикладного программного обеспечения, модульной программной платформы для интеграции программных продуктов, разработанных разными коллективами в различных организациях.

Уровень внедрения суперкомпьютерных технологий напрямую зависит от обеспечения высокопроизводительный вычислительной базой.

По направлению программного обеспечения для суперкомпьютерного моделирования

– ФГУП «РФЯЦ-ВНИИЭФ» создан базовый многофункциональный конкурентоспособный с лучшими зарубежными мировыми аналогами полностью отечественный пакет программ для суперкомпьютерного моделирования и инженерного анализа Логос.

Передан более чем в 80 организаций промышленности, науки и образования, оснащено более 350 рабочих мест внешних организаций, обучено более 700 студентов и специалистов

Выполненные работы позволили обеспечить на конец 2020 года (данные приведены в скобках) охват задач промышленности в объемах:

- аэродинамика – до 95% (80%);
- гидродинамика – до 95% (70%);
- прочность и разрушение – до 75% (50%);
- теплоперенос – до 90% (60%).

Пакет программ Логос включен в единый реестр российского программного обеспечения для электронных вычислительных машин и баз данных и внедрен в работы в более чем 80 организаций ОПК, промышленности и образования.

– Создана программная платформа Логос Платформа для объединения в единый отечественный пакет программ специализированных модулей разработки других организаций, а также создания специализированных версий пакета программ Логос.

К настоящему времени на основе этой платформы созданы и успешно сданы специализированные версии комплекса Логос для работ соответственно в области авиастроения, судостроения, обычных вооружений, ведутся работы по разработке новых специализированных комплексов.

– Специализированное программное обеспечение для отраслей промышленности на базе программной платформы Логос. Используется

при проектировании более чем 40 высокотехнологичных изделий с учетом специфики их работы.

– С использованием базового пакета программ Логос ФГУП «РФЯЦ-ВНИИЭФ» совместно с большим кругом организаций ОПК, науки и образования начаты работы по созданию новой расчетной технологии для использования в проектировании и создании высокотехнологичных изделий. Получены первые результаты применения этой технологии.

По направлению развития высокопроизводительной вычислительной базы

В ФГУП «РФЯЦ-ВНИИЭФ» созданы и успешно развиваются:

– Центр коллективного пользования для предоставления высокопроизводительных вычислительных ресурсов предприятиям ОПК на основе технологии удаленного доступа, в том числе в защищенном исполнении (подключено более 70 организаций), оснащен современными супер-ЭВМ, системным и прикладным программным обеспечением собственной разработки, программно-аппаратными средствами защиты информации и т.д.

– Линейка компактных супер-ЭВМ терафлопсного класса – 12 модификаций (на предприятия промышленности поставлено более 170 КС-ЭВМ), в том числе на базе отечественных компонентов.

– Полностью отечественная система межпроцессорных обменов (СМПО) – ключевая компонента супер-ЭВМ.

– Отечественная защищенная операционная система «Арамид», сертифицированная во ФСТЭК России для обработки информации, содержащей государственную тайну. Не имеет отечественных аналогов.

В докладе представлены планы по дальнейшему развитию отечественных суперкомпьютерных технологий и высокопроизводительной вычислительной базы в интересах внедрения в работы организаций науки, промышленности и образования.

С.А. Щаников[1], И.А. Борданов[1], Е.А. Букварев[2], В.П. Хранилов[2], Е.Г. Грязнов[3], А.Н. Михайлов[3]

МАСШТАБИРУЕМЫЕ АППАРАТНЫЕ УСКОРИТЕЛИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА НА БАЗЕ МЕМРИСТИВНЫХ УСТРОЙСТВ*

[1]Муромский институт (филиал) Владимирского государственного университета имени Александра Григорьевича и Николая Григорьевича Столетовых, г. Муром
[2]Нижегородский Государственный Технический Университет им. Р.Е. Алексеева, г. Нижний Новгород
[3]Национальный исследовательский Нижегородский государственный университет имени Н.И. Лобачевского, г. Нижний Новгород,
seach@inbox.ru

Введение

В настоящее время электронно-вычислительная техника работает на двух фундаментальных технологиях: кремний, как основной материал для производства памяти и процессоров, и архитектура Джона фон Неймана, в основе которой лежит принцип физического разделения памяти и процессора на два отдельных устройства. Такой принцип создания электронно-вычислительной техники позволял долгое время эффективно решать задачи, возникающие во всех сферах человеческой деятельности: бытовая радиоэлектроника, промышленное оборудование, транспорт, военная техника и т.д. в связи с тем, что эти задачи являются формализуемыми, то есть для них можно четко описать последовательность шагов, которая приведет к решению. С появлением и развитием систем искусственного интеллекта (ИИ) человечество научилось решать сложно формализуемые и неформализуемые задачи. Глобальным вызовом на этом пути стала проблема жесткой зависимости между производительностью электронно-вычислительной техники и её размерами, и энергопотреблением, усугубляемая высокой вычислительной сложностью алгоритмов и моделей ИИ, что дало толчок к развитию нового класса систем – аппаратных ускорителей ИИ.

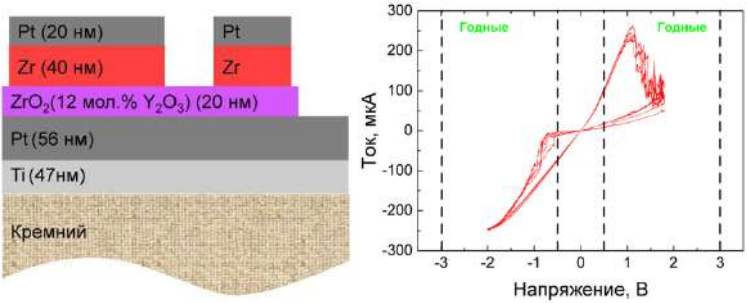
Методы и результаты

Доклад посвящен вопросам создания масштабируемых аппаратных ускорителей ИИ. Аппаратный ускоритель ИИ (с англ. AI accelerator) – это

*Работа выполнена при поддержке Субсидии Министерства науки и высшего образования РФ (проект №13.2251.21.0098, соглашение № 075–15–2021–1017).

специализированное устройство, архитектура которого позволяет выполнять алгоритмы искусственных нейронных сетей быстрее по сравнению с универсальным вычислителем. Наибольшими перспективами в данной области с точки зрения повышения производительности и снижения энергопотребления обладают нейроморфные системы на основе мемристивных устройств, что подтверждается анализом публикаций [1-6], а также собственными исследованиями авторов доклада [7-9]. Массив мемристивных устройств в топологии «кросс-бар» идеально подходит для аппаратной реализации операции матрично-векторного умножения (МВУ), реализуя её естественным образом на основе законов Ома и Кирхгофа. Поскольку МВУ является наиболее используемой операцией в нейросетевых алгоритмах, их аппаратная реализация в структуре «кросс-бар» позволяет на порядки величины повысить производительность и быстродействие нейроморфных вычислительных систем.

Для создания аппаратных ускорителей ИИ авторами разработаны мемристивные устройства типа «металл-диэлектрик-метал» (МДМ), где в качестве основного диэлектрика используется стабилизированный иттрием диоксид циркония, осаждаемый методом магнетронного распыления (рис. 1). Конструктивные варианты и технологические приемы изготовления таких мемристивных устройств совместимы с современным КМОП-процессом и предусматривают различные комбинации электродов для управления параметрами резистивного переключения. Для проведения исследований новых конструктивных вариантов мемристивных устройств авторами разработаны топологии тестовых кристаллов, содержащих как массивы отдельных мемристивных устройств, так и различные варианты «кросс-баров» (в том числе с селекторами), что позволяет отработать как простые, так и более сложные архитектуры аппаратных ускорителей.



а) б)

Рис. 1. Мемристивные устройства: а) Схематичное представление МДМ структуры Au/Ta/ZrO2(Y)/Pt/Ti; б) вольт-амперная характеристика.

Разработана масштабируемая архитектура вычислительного ядра аппаратного ускорителя (рис. 2), а также набор моделей различного уровня структурно-функциональной иерархии (от моделей уровня систем до моделей уровня электронных компонентов). Сигналы на все входы аппаратного ускорителя подаются одновременно с помощью ПЛИС (программируемой логической интегральной схемы). ПЛИС вместе с набором ЦАП (цифро-аналоговых преобразователей) представляет собой генератор сигналов произвольной формы. После обработки сигналов НСМ они регистрируются на АЦП (аналого-цифровых преобразователях) и передаются на ПЛИС для дальнейшего анализа. ПЛИС и набор АЦП в данном случае может быть рассмотрен как многоканальный осциллограф. Предусмотрено многоуровневое масштабирование системы.

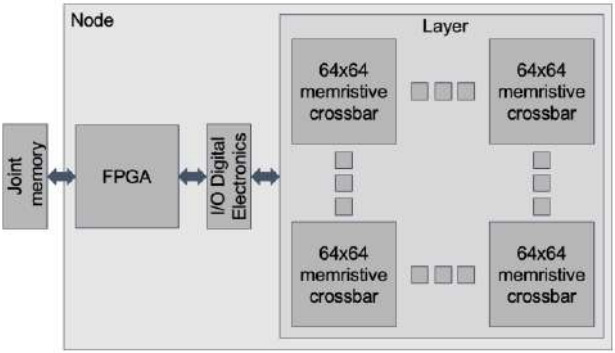


Рис. 2. Структура вычислительной ячейки аппаратного ускорителя ИИ на базе мемристивных устройств.

В докладе представлено описание текущих результатов и планы развития данного направления.

Заключение

Тренд развития систем ИИ должен быть направлен в сторону разработки новых концепций, архитектур и электронных компонентов. Как показывает анализ современных научных публикаций, а также собственный опыт авторов, мемристоры позволяют качественно и количественно улучшить характеристики современных систем искусственного интеллекта и создавать аппаратные ускорители ИИ, принципы работы которых максимально приближены к принципам работы биологических нейронных сетей. Научно-технические результаты, представленные в докладе, предназначены для применения в вычислительной технике, технологиях

обработки больших данных и ИИ и могут быть востребованы ведущими производителями интегральных микросхем и электронной техники, а также другими организациями, заинтересованными в выполнении научных исследований и опытно-конструкторских работ в данной области.

1. **Галушкин, А. И.** На пути к нейрокомпьютерам с использованием мемристоров / А.И. Галушкин //Информационные технологии. – 2014. – №. 4. – С. 2–19.
2. **Amirsoleimani, A.** In-Memory Vector-Matrix Multiplication in Monolithic Complementary Metal–Oxide–Semiconductor-Memristor Integrated Circuits: Design Choices, Challenges, and Perspectives / A. Amirsoleimani, F. Alibart, V. Yon [et al.] // Advanced Intelligent Systems. – 2020. – 2000115.
3. **Zhang, Y.** Brain-inspired computing with memristors: Challenges in devices, circuits, and systems / Y. Zhang, Z. Wang, J. Zhu [et al.] // Applied Physics Reviews. – 2020. – Vol. 7. – 011308.
4. **Mikhaylov, A.** Neurohybrid memristive CMOS-integrated systems for biosensors and neuroprosthetics / A. Mikhaylov // Frontiers in neuroscience. – 2020. – Vol. 14. – P. 358.
5. **Wang, Z.** Resistive switching materials for information processing / Z. Wang, H. Wu, G.W. Burr [et al.] // Nat. Rev. Mater. – 2020. – Vol. 5. – P. 173–195.
6. **Zhu, J.** A comprehensive review on emerging artificial neuromorphic devices / J. Zhu, T. Zhang, Y. Yang, R. Huang // Applied Physics Reviews. – 2020. – Vol. 7. – 011312.
7. **Shchanikov, S.** Designing a bidirectional, adaptive neural interface incorporating machine learning capabilities and memristor-enhanced hardware / S. Shchanikov // Chaos, solitons & fractals. – 2021. – Vol. 142. – 110504.
8. **Surazhevsky, I.A.** Noise-assisted persistence and recovery of memory state in a memristive spiking neuromorphic network / I.A. Surazhevsky // Chaos, solitons & fractals. – 2021. – Vol. 146. – 110890.
9. **Shchanikov, S.** Fault tolerance of memristor-based perceptron network for neural interface / S. Shchanikov // BioNanoScience. – 2021. – Vol. 11, Is. 1. – P. 84–90.

РАЗДЕЛ 2

МАТЕМАТИЧЕСКОЕ И ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ МВУС

К.А. Баркалов, Е.А. Козинов, И.Г. Лебедев, А.В. Сысоев, М.А. Усова

ПАРАЛЛЕЛЬНЫЕ МЕТОДЫ РЕШЕНИЯ ЗАДАЧ МНОГОКРИТЕРИАЛЬНОЙ МНОГОЭКСТРЕМАЛЬНОЙ ОПТИМИЗАЦИИ ДЛЯ МНОГОПРОЦЕССОРНЫХ СИСТЕМ*

Национальный исследовательский Нижегородский государственный университет им. Н.И. Лобачевского, г. Нижний Новгород, konstantin.barkalov@itmm.unn.ru, evgeny.kozinov@itmm.unn.ru

Введение

К числу вычислительно-трудоемких задач, для решения которых могут потребоваться суперкомпьютерные системы, относятся проблемы многокритериальной многоэкстремальной оптимизации в различных областях приложений. В задачах такого вида допускается, что критерии могут иметь несколько локальных оптимумов в области поиска, которые имеют различные значения. Наличие нескольких локальных экстремумов существенно усложняет поиск глобального оптимума, так как требует исследования всей допустимой области поиска. В случае нескольких критериев сложность задачи многократно возрастает, так как для ее решения необходимо найти целое множество компромиссных вариантов.

Постановки задач глобальной оптимизации используются, как правило, в наиболее трудных ситуациях оптимального выбора, когда проводится моделирование сложных технических объектов, изделий и систем. Показатели эффективности моделируемой системы могут являться нелинейными, области поиска могут быть несвязными, а вычислительная сложность функционалов, лежащих в основе оптимизируемых критериев, может быть значительной.

Многокритериальные постановки задач используются в случаях, когда объект оптимизации сложно описать одним критерием. Решение подобных задач, как правило, требует определения значимости каждого критерия, а затем решения серии задач глобального поиска. В некоторых случаях значимость критериев может меняться, что приводит к необходимости решения новых задач глобального поиска.

---

* Работа выполнена при финансовой поддержке Министерства науки и высшего образования РФ (проект № 0729–2020–0055) и научно-образовательного математического центра «Математика технологий будущего» (соглашение № 075–02–2021–1394).

Общее состояние исследований в области глобальной оптимизации достаточно полно представлено в ряде ключевых работах [1-5]. Основные методы решения многокритериальных задач оптимизации можно найти в работах [6-8].

Коллективом авторов разработана система Globalizer, сочетающая эффективные методы решения задач многокритериальной многоэкстремальной оптимизации [9-14]. В данной работе показаны ключевые особенности и продемонстрирована эффективностью методов, реализованных в Globalizer.

Постановка задачи

В рамках системы Globalizer задача многокритериального глобального поиска (МКП) состоит в решении задач вида:

$$f(y) = \big(f_1(y), f_2(y), \dots, f_s(y)\big) \to min, y \in D, \tag{1}$$

$$D = \{\, y \in R^N : a_i \le y_i \le\ b_i, 1 \le i \le N, a, b \in R\ \}. \tag{2}$$

где $f_i(y)$, $1 \le 1 \le s$, критерии эффективности, $y = (y_1, y_2, \dots, y_N)$ вектор варьируемых параметров, а N размерность решаемой задачи. Предполагается, что критерии $f_i(y)$, $1 \le i \le s$, являются многоэкстремальными, заданы в виде «черного ящика» и удовлетворяют условию Липщица

$$|f_i(y') - f_i\ (y'')| \le L_i\ \|y' - y''\ \|, y'\ , y'' \in D, 1 \le i \le s, \tag{3}$$

где L_i, $1 \le i \le s$, априори неизвестные константы Липщица.

В качестве частного решения задачи МКП может рассматриваться любой эффективный (Парето-оптимальный) вариант, в котором нельзя уменьшить значения сразу всех критериев $f_i(y)$, $1 \le 1 \le s$, путем изменения значений параметров $y \in D$. Как отмечалось ранее, чтобы решить задачу МКП, как правило, необходимо найти несколько эффективных вариантов.

Подход к решению задач многокритериального поиска

На практике, для поиска эффективных вариантов часто применяются различные методы скаляризации [6-9]. В рамках настоящего исследования применяется минимаксная свертка частных критериев, обладающая хорошими теоретическими свойствами

$$\min f(y) \to \min_{y \in D}\ \max_{1 \le i \le s}(f_i(y)\lambda_i), \tag{4}$$

где λ_i, $1 \le i \le s$, есть показатели значимости каждого критерия.

В рамках предлагаемого подхода для решения скалярных задач оптимизации (4) используется информационно-статистическая теория глобального поиска. При решении задач с несколькими параметрами ($N > 1$), выполняется редукция размерности на основе кривых или разверток Пеано [4]. Для поиска оптимального решения выполняются испытания. Под испытанием понимается вычисление значений критериев $f_i\ (y), 1 \leq 1 \leq s$, из (4) при заданных значениях параметра y, а также выбранном наборе коэффициентов λ. Выбор значений оптимизируемых параметров y выполняется согласно следующим правилам.

1. Для каждого набора значений параметров y_i, $1 \leq i \leq k$, для которых были выполнены испытания, используя развертку Пеано, получить образ x_i, $1 \leq i \leq k$, на одномерной отрезке [0,1].
2. Разбить отрезок [0,1] на интервалы согласно образам проведенных испытаний.
3. Вычислить характеристику $R(i)$ для каждого интервала.
4. Выбрать интервал с максимальным значением характеристики $R(i)$.
5. В выбранном на одномерном отрезке интервале определить очередное значение x_{k+1} для проведения испытаний.
6. По выбранному значению x_{k+1} на одномерном отрезке определить значения параметров y_{k+1} проведения нового испытания.

Алгоритм заканчивает выполнение при достижении требуемой точности. В предлагаемом подходе, после остановки алгоритма глобального поиска, на основе накопленной в процессе оптимизации информации строится оценка области Парето. Если качество полученной оценки области Парето является не достаточным, то задаются новые предпочтения (набор коэффициентов λ) и процесс поиска продолжается.

Для эффективного параллельного поиска в рамках системы Globalizer реализованы несколько модификаций.

Первая модификация позволяет параллельно и совместно решать сразу несколько задач (4) при разных наборах коэффициентов λ. Каждая решаемая задача (4) может быть запущенна на отдельном вычислительном узле. Эффективность поиска повышается за счет обмена, хранения и повторного использования вычисленных значений критериев [13].

Вторая модификация позволяет задействовать параллельные свойства каждого отдельного вычислительного узла. При выполнении правил 4–6 алгоритма глобального поиска, выбирается не один, а несколько интервалов с максимальным значением характеристик. В каждом выбранном интервале вычисление очередной точки испытания и значений критериев может быть выполнено параллельно [4,13].

Эффективность разработанных параллельных алгоритмов показана на примере решения серии задач многокритериальной оптимизации.

Результаты вычислительных экспериментов

Вычислительные эксперименты проводились на суперкомпьютере «Лобачевский» Нижегородского государственного университета. Для демонстрации эффективности разработанных алгоритмов были выполнены эксперименты по решению 100 тестовых четырехмерных десятикритериальных задач с критериями, получаемыми при помощи генератора GKLS [4], т.е. $N = 4$, $s = 10$.

Условием остановки метода при решении очередной задачи оптимизации из (4) являлось нахождение набора параметров y в ε-окрестности минимального значения соответствующей свертки частных критериев. Точность поиска была выбрана $\varepsilon = 0.025$. Для решения каждой задачи МКП решалось $\tau = 50$ задач (4) при разных значениях коэффициентов свертки λ. Результаты вычислений были усреднены по всем решенным задачам МКП.

Результаты экспериментов представлены в таблице 1. Первые два столбца в таблице 1 обозначают количество используемых процессоров (P) и вычислительных ядер каждого процессора (Q). В третьем столбце (P*Q) содержится общее количество используемых вычислительных ядер при решении задачи МКП. В четвертом и пятом столбце приведено среднее количество итераций и испытаний, необходимых для нахождения решения задач. Последний столбец содержат информацию об ускорении параллельных вычислений.

Таблица 1

Эффективность параллельного метода решения задач многокритериальной оптимизации

P	Q	P*Q	Итераций	Испытаний	Ускорение
1	1	1	23 791.80	23 791.80	1
1	40	40	616.4	24 654.80	38.6
50	1	50	594.5	29 723.10	40.0
50	40	2000	21.0	41 967.70	1 134.0

Из таблицы 1 следует, что использование 40 вычислительных ядер или 50 процессоров позволяет получить ускорение около 40 раз. При использовании в общей сложности 2000 вычислительных ядер величина ускорения достигает величины 1134. С учетом сложности решаемой задачи получаемые ускорения вычислений является значительным.

Заключение

В рамках работы демонстрируются возможности системы Globalizer, реализующей эффективные методы решения сложных многокритериальных задач оптимизации, в которых критерии оптимальности могут быть многоэкстремальными, а вычисление значений критериев может требовать большого объема вычислений.

Эффективность реализованных методов основана на хранении и повторном использовании поисковой информации, а также на использовании нескольких уровней параллелизма в гетерогенных вычислительных системах. Представленные результаты вычислительных экспериментов подтверждают эффективность предлагаемых подходов.

1. **Floudas, C.A.** Recent advances in global optimization / C.A. Floudas, M.P. Pardalos. – Princeton University Press, 2016. – 644 p.
2. **Locatelli M.** Global optimization: theory, algorithms and applications / M. Locatelli, F. Schoen. – SIAM, 2013. – 432 p.
3. **Pintér, J.D.** Global optimization in Action (Continuous and Lipschitz optimization: algorithms, implementations and applications) / J.D. Pintér. – Kluwer Academic Publishers, Dordrecht, 1996. – 480 p.
4. **Strongin, R.G.** Global optimization with non-convex constraints. Sequential and parallel algorithms / R.G. Strongin, Y.D. Sergeyev. – Kluwer Academic Publishers, Dordrecht, 2000, 2nd ed. 2013, 3rd ed. 2014. – 704 p.
5. **Pardalos, M.P.** Advances in stochastic and deterministic global optimization / M.P. Pardalos, A.A. Zhigljavsky, J. Žilinskas. – Springer, 2016. – 296 p.
6. **Miettinen, K.** Nonlinear Multiobjective Optimization / K. Miettinen. – Springer, 1999. – 298 p.
7. **Ehrgott M.** Multicriteria Optimization / M. Ehrgott. – Springer, 2005. – 323 p.
8. **Pardalos, P.M.** Non-Convex Multi-Objective Optimization / P.M. Pardalos, A. Žilinskas, J. Žilinskas. – Springer, 2017. – 192 p.
9. **Gergel V.** Adaptive nested optimization scheme for multidimensional global search / V. Gergel, V. Grishagin, A. Gergel // Journal of Global Optimization. – 2016. – №66(1). – P. 35–51.
10. **Sysoyev A.** Globalizer – A parallel software system for solving global optimization problems / A.Sysoyev, K. Barkalov, V. Sovrasov [et al.] // Lecture Notes in Computer Science. – 2017. – № 10421. – P. 492–499.
11. **Barkalov, K.A.** Acceleration of global optimization algorithm by detecting local extrema based on machine learning / K.A. Barkalov, I.G. Lebedev, E.A. Kozinov // Entropy. – 2021. – № 23. – P. 1272.
12. **Gergel, V.P.** Efficient multicriterial optimization based on intensive re-use of search information / V.P. Gergel, E.A. Kozinov // J. Glob. Optim. – 2018. – № 71(1). – P. 73–90.
13. **Gergely, V.P.** Parallel solving of multiple information-coordinated global optimization problems / V.P. Gergely, E.A. Kozinov // Journal of Parallel and Distributed Computing. – 2021. – № 154. – P. 153–162.
14. **Barkalov, K.A.** An approach for simultaneous finding of multiple efficient decisions in multi-objective optimization problems / K.A. Barkalov, V.P. Gergel, V.A. Grishagin, E.A. Kozinov // Lecture Notes in Computer Science. – 2021. – № 12755. – P. 127–143.

Д.В. Вахлаков, С.Ю. Мельников, В.А. Пересыпкин

ЗАДАЧА КОРРЕКЦИИ ИСКАЖЕННЫХ ТЕКСТОВ НА ЕСТЕСТВЕННЫХ ЯЗЫКАХ. АКТУАЛЬНОСТЬ И СОСТОЯНИЕ ДЕЛ. ПРОБЛЕМЫ И ПЕРСПЕКТИВЫ

НТЦ «Орион», ООО «Линфо», г. Москва,
melnikov@linfotech.ru

Актуальность задачи

Большой вклад в развитие методов распознавания образов внесли отечественные ученые [1]. Характерным примером актуальной задачи распознавания образов является задача коррекции искаженных текстов, полученных теми или иными системами машинного распознавания (речи, изображений текстов и др.). В последние годы эта задача формируется как отдельное направление (пост-обработка) и привлекает значительное внимание исследователей. В 2017 и 2019 гг. в рамках конференции International Conference on Document Analysis and Recognition (ICDAR) проводились соревнования различных систем коррекции текстов, полученных в результате оптического распознавания [2, 3]. В соревнованиях принимало участие более 30 участников, и если в 2017 году рассматривались тексты на двух языках (английский и французский), то в соревнованиях 2019 года к ним добавились болгарский, чешский, нидерландский, финский, немецкий, польский, испанский и словацкий языки. Близкие к описанным постановки рассматриваются также в биоинформационных задачах секвенирования и сборки больших геномов [4] для коррекции так называемых чтений, получаемых с помощью машин-секвенаторов.

Основным фактором, существенно затрудняющим понимание и перевод текстов, полученных при машинном распознавании речи или изображений текстов, в том числе в перспективных системах дополненной (виртуальной) реальности [5, 6], являются содержащиеся в них искажения в виде ошибочных символов, слов и словосочетаний [7, 8]. При значительном количестве искажений такие тексты становятся практически нечитаемыми.

Состояние дел. Методы и средства коррекции

До недавнего времени не существовало эффективных программных средств коррекции текстов со значительными искажениями (или сильно искаженных текстов), хотя эта задача является актуальной как для русского, так и для других распространенных языков. В [9] показано, что распространенные программные средства коррекции, такие как Яндекс.Спеллер, Texterra, Bing Spell Check и другие, хорошо работающие при малых искаже-

ниях в тексте, в случае текстов с высоким уровнем искажений, вне зависимости от их происхождения (набранных с ошибками на клавиатуре, полученных в результате распознавания речи в условиях шумов и др.), показывают неудовлетворительные результаты. Это делает необходимым разработку новых подходов к коррекции сильно искаженных текстов.

В работе [10] авторами предложен новый многоэтапный метод коррекции искаженных текстов (в том числе, и сильно искаженных), значимо повышающий точность коррекции (количество правильно скорректированных слов в тексте) и основанный на последовательном определении ошибок и их исправлении. В качестве единицы искаженного текста рассматривается слово. Если в тексте имеется слово, не входящее в словарь словоформ языка, либо имеющее низкую вероятность в соответствии с выбранной вероятностно-лингвистической моделью языка (как правило, это N-граммная модель Маркова на словах), то оно помечается как искаженное, а в качестве возможных вариантов его исправления используются слова из словаря, характеризующиеся высокой степенью сходства с ошибочным словом по некоторой мере. Меру сходства корректируемого слова и слова из словаря вычисляется с использованием расстояния Левенштейна, которое равно минимальному количеству изменений (вставок, замен и удалений) символов алфавита, необходимых для преобразования одного слова в другое. После установки признака искаженности для отдельных слов происходит распространение этого признака на их сочетания, т.е. выделяются искаженные фрагменты текста. Скорректированный текст получается в результате обработки колонок слов с помощью алгоритма динамического программирования с заданной глубиной зависимости между словами и построения цепочек слов на основе вероятностно-лингвистической модели языка. Метод коррекции состоит из нескольких этапов, на каждом из которых используется определенное значение k расстояния Левенштейна (k=1,2…), а входными данными являются результаты предыдущего этапа. Количество этапов k является параметром и задается в зависимости от степени искаженности текста.

Оценена точность и трудоемкость (время/скорость работы на современных вычислительных средствах) предложенного метода коррекции искаженных текстов при различных уровнях искажений [11]. Мерой точности коррекции служила F1-мера, которая вычисляется как гармоническое среднее точности A и полноты R коррекций искажённого текста с одинаковым весом, т.е.

$$F1 = \frac{2AR}{(A + R)}.$$

Сделан вывод, что предложенный метод позволяет обеспечить достаточно высокую точность коррекции (в терминах F1-меры для средне и сильно искаженных текстов значение составило около 0.7, максимальное значение расстояния Левенштейна составило 4, т.е. алгоритм коррекции состоял из 4-х этапов) при высоком быстродействии (менее 2 секунд на страницу текста на вычислительном сервере с 44 ядрами и ОЗУ 250 ГБ).

Эффективность предложенного корректора оценивалась экспертами-лингвистами [12]. Установлено, что в случае очень хорошего знания языка эксперт корректирует искаженный текст точнее, чем автоматический корректор. В случае коррекции искаженного текста квалифицированным экспертом результаты его работы несколько ниже точности автоматического корректора. Однако если эксперту предоставить дополнительное время для работы, он заметно улучшает свои результаты. Для эксперта-лингвиста с квалификацией ниже средней точность коррекции искаженных текстов заметно хуже, чем точность работы автоматического корректора.

Проблемы и перспективы развития метода коррекции

Основным фактором, оказывающим влияние на эффективность коррекции искаженных текстов, является выбор и построение адекватной модели языка, требующее значительных лингвистических и вычислительных ресурсов. Разработанный метод коррекции используют сложные вычислительные процедуры, которые требуют многократных вычислений правдоподобий вариантов скорректированного текста по вероятностно-лингвистическим моделям языка текста. При выборе моделей учитываются такие обстоятельства, как достигаемая точность коррекции (косвенно ее можно характеризовать перплексией модели) и трудоемкость обучения модели на имеющихся тематических текстовых корпусах, в условиях использования доступных вычислительных ресурсов. В [10] в качестве моделей английского языка использовались 4-граммные Марковские модели на словах, построенные на корпусе текстов объемом 200 млн. слов, с модифицированным сглаживанием Кнессера-Нея [13], учитывающим отсутствие в обучающих текстах некоторых 4-х грамм слов. Особенно остро проблема набора обучающих текстов стоит для малоресурсных языков [14], не имеющих столь развитых электронных ресурсов как русский, английский и другие распространенные языки. Для программной реализации корректора (от создания корпусов текстов и построения вероятностно-лингвистических моделей слов до коррекции искаженных текстов) потребовалась следующая мощная вычислительная конфигурация: процессор Intel (R) Xeon (R) CPU E5-2699 v4 @ 2.20GHz, 44 ядра, ОЗУ 250 ГБ.

В последнее время все большую популярность приобретают нейросетевые модели языка. В работе [15] проведено сравнение *N*-граммной марковской модели с двумя нейросетевыми моделями языка текста с целью оценки

целесообразности использования последних для коррекции искаженных текстов. Были выбраны и исследованы два типа нейросетевых языковых моделей – QRNN и BiLSTM. Перплексия построенных нейросетевых моделей, равная 120,42, для английских корпусов текстов оказалась значимо выше, чем перплексия *N*-граммных моделей, равная 171,87, на тех же корпусах текстов, что дает основание для использования нейросетевых моделей языка для повышения точности коррекции искаженных текстов. Однако обучение нейросетевых моделей занимает значительное время даже на сравнительно небольшом использованном корпусе (около 100 млн слов). Для корпусов больших объемов без ускорителей вычислений обучение нейросетевых моделей по существующим алгоритмам провести за разумное время практически невозможно. Резервом повышения качества нейросетевых моделей является увеличение объема обучающего корпуса и числа эпох обучения, что целесообразно реализовывать на видеокартах Nvidia 3xxx поколения, с бóльшими объемом памяти и скоростью обработки.

В связи с возможным появлением новых типовых ошибок при распознавании речи и изображений текстов для поддержания высокой точности коррекции необходим регулярный анализ и последующий возможный учет этих ошибок для совершенствования метода коррекции искаженных текстов. Ранее ([9]) были систематизированы наиболее характерные типы ошибок, а именно:

– замена слова на похожее по звучанию или графическому написанию;
– замена нескольких слов на одно;
– замена одного слова несколькими;
– пропуск слов;
– вставка или удаление коротких слов (предлогов и союзов).

Установлено, что в результате машинного распознавания получается текст, имеющий искажения и состоящий, в основном, из словарных слов, в том числе и в местах искажений. Для коррекции таких искажений необходимы модели языка с достаточно глубокими зависимостями между словами (N>3).

Необходимо также учитывать существенное влияние на качество коррекции искаженных текстов специфики языка [12]. Например, для арабского языка – это особенности письменного оформления текстов, в частности: практически полное отсутствие в текстах огласовок, обозначающих краткие гласные, что усиливает смысловую неоднозначность слов и во многих случаях требует обращения к контексту; сложная морфология, т.е. преобладающее построение слов на основе трех корневых частей слова (харфов) с соответствующей аффиксацией по определенным моделям, характерным для какой-либо части речи и выражающим те или иные грамматические и/или лексические значения, и др. Поэтому значительную трудность при коррекции искаженных текстов на арабском языке представляют короткие слова (состоящие из 3–5 харфов), где при искажении заменяются, либо опускаются один

или несколько корневых харфов. Возможно правильно корректировать фрагменты даже сильно искаженного текста, если эти фрагменты представляют собой устойчивые, либо часто встречающиеся словосочетания, или они несколько раз встречаются в одном тексте, но с разными искажениями.

Выводы

Рассмотрена актуальная проблема коррекции текстов, полученных при машинном распознавании речи или изображений текстов. Отмечено, что до недавнего времени не существовало эффективных программных средств коррекции текстов со значительными искажениями Авторами предложен новый многоэтапный метод коррекции искаженных текстов, основанный на последовательном исправлении искаженных слов с использованием 4-х граммной Марковской модели текста. Сделан вывод (в том числе, на основе работы экспертов-лингвистов) о достаточно высокой эффективности предложенного метода коррекции. Обозначены имеющиеся проблемы при коррекции искаженных текстов, пути их решения и перспективы дальнейшего развития корректора.

1. **Каляев, А.В.** Распознавание образов на основе цифровых нейроноподобных структур / А.В. Каляев, Ю.В. Чернухин, Г.А. Галуев // Известия Сев. Кавказ. Научн. Центра Высш. Школы. Технические науки. – 1982. – С.44–49.
2. **Chiron, G.** ICDAR 2017 competition on post-OCR text correction / G. Chiron, A. Doucet, M. Coustaty, J.P. Moreux // 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). – 2017. – Vol. 1. – P. 1423–1428.
3. **Rigaud, C.** ICDAR 2019 Competition on Post-OCR Text Correction / C. Rigaud, A. Doucet, M. Coustaty, J.P. Moreux // International Conference on Document Analysis and Recognition. – 2019. – P. 1588–1593.
4. **Das, A.K.** A hybrid and scalable error correction algorithm for indel and substitution errors of long reads / A.K. Das, S. Goswami, K. Lee, S.J. Park // BMC Genomics. – 2019. – Vol. 20 (Suppl 11) – P.1–15.
5. Пехотная система дополненной реальности IVAS (США) [электронный ресурс]. – URL: https://topwar.ru/181316-pehotnaja-sistema-dopolnennoj-realnosti-ivas-ssha.html (дата обращения: 29.03.2021).
6. Компьютерное_зрение: технологии, рынок, перспективы [электронный ресурс]. – URL: https://www.tadviser.ru/index.php/Статья:Компьютерное_зрение:_технологии,_рынок,_перспективы (дата обращения: 26.06.2019).
7. **Мещеряков, Р.В.** Структура систем синтеза и распознавания речи / Р.В. Мещеряков // Известия Томского политехн. ун-та. – 2009. – Т. 315, № 5. – С. 127–132.
8. **Смирнов, С.В.** Корректировка ошибок оптического распознавания на основе рейтинго-ранговой модели текста / С.В. Мещеряков // Труды СПИИРАН. – 2014. – Вып. 4, № 35. – С. 64–82.
9. **Бирин, Д.А.** Об эффективности средств коррекции искаженных текстов в зависимости от характера искажений / Д.А. Бирин, С.Ю. Мельников, В.А. Пересыпкин, И.А. Писарев, Н.Н. Цопкало // Известия ЮФУ. Технические науки. – 2018. – № 8. – С.104–114. DOI: 10.23683/2311-3103-2018-8-104-114.

10. **Вахлаков, Д.В.** Многоэтапный метод автоматической коррекции искаженных текстов / Д.В. Вахлаков, С.Ю. Мельников, В.А. Пересыпкин // Известия ЮФУ. Технические науки. – 2020. – № 7. – С.35–45. DOI: 10.18522/2311-3103-2020-7-35-45.
11. **Вахлаков, Д.В.** О точности и трудоемкости многоэтапного метода коррекции искаженных текстов в зависимости от степени искажения / Д.В. Вахлаков, А.В. Германович, С.Ю. Мельников [и др.] // Известия ЮФУ. Технические науки. – 2021. – № 7 (224). – С. 130–142. DOI: 10.18522/2311-3103-2021-7-130-142.
12. **Германович, А.В.** Информационные измерения языка. Программная система оценки читаемости искаженных текстов / А.В. Германович, С.Ю. Мельников, В.А. Пересыпкин [и др.] // Известия ЮФУ. Технические науки. – 2019. – № 8. – С. 6–18. DOI: 10.23683/2311-3103-2019-7-6-17
13. **Ney, H.** On structuring probabilistic dependences in stochastic language modeling / H. Ney, U. Essen, R. Kneser // Computer Speech and Language. – 1994. – Vol. 8. – P. 1–38.
14. **Magueresse, A.** Low-resource Languages: A Review of Past Work and Future Challenges [электронный ресурс] / A. Magueresse, V. Carles, E. Heetderks // arXiv preprint. – 2020. –arXiv:2006.07264.
15. **Вахлаков, Д.В.** О сравнении вероятностно-статистических и нейросетевых моделей языка текста при коррекции ошибок распознавания в перспективных системах дополненной реальности / Д.В. Вахлаков, С.Ю. Мельников, В.А. Пересыпкин, В.М. Хвостенко // XIV Всероссийская мультиконференция по проблемам управления МКПУ-2021. Материалы XIV мультиконференции: в 4-х т. – 2021. – Т.2. – С.202–206.

И.Г. Данилов, И.Н. Ерошенко

СИМУЛЯТОР ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ СИСТЕМЫ МОНИТОРИНГА И ТЕСТИРОВАНИЯ РВБ

НИЦ супер-ЭВМ и нейрокомпьютеров, г. Таганрог,
danilov@superevm.ru, eroshenko@superevm.ru

Введение

В работе приведено описание ПО системы мониторинга и тестирования (СМТ) реконфигурируемых вычислительных блоков (РВБ) на основе ПЛИС, а также рассмотрен прототип симулятора ПО СМТ для отладки сложного клиентского программного обеспечения. Данные РВБ и СМТ разрабатываются в ООО «НИЦ СЭ и НК» (г. Таганрог).

Архитектура программного обеспечения СМТ

Для ПО СМТ характерна многоуровневая архитектура, состоящая из графической программы-клиента, HTTP-сервера, REST-сервиса, набора микросервисов мониторинга и тестов, слоя прямого доступа к данным

(рис. 1) [1, 2]. REST-сервис содержит каталог приложений в виде микросервисов. Каждый микросервис выполняет определенный набор подзадач (например, тест регистров, анализ скорости вращения вентиляторов или считывание температуры ядра ПЛИС). REST-сервис обменивается данными с микросервисами мониторинга и тестирования, благодаря технологии межпроцессного взаимодействия Qt Remote Objects (QtRO).

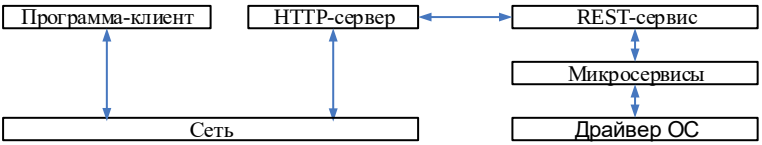


Рис. 1. Архитектура ПО системы мониторинга ресурсов.

На компьютере оператора программа-клиент выполняет HTTP-запросы к REST-сервисам мониторинга для формирования дерева ресурсов вычислительной системы. Для организации обратной связи с программой-клиентом используется протокол WebSocket.

В главном окне программы можно увидеть глобальный статус дерева ресурсов (наличие ошибок/предупреждений), а также статусы промежуточных узлов. Для детального изучения параметров какого-либо конечного узла дерева (РВБ) оператором используется окно мониторинга. Данное окно содержит множество дочерних элементов (таблицы, графики, индикаторы), которые подвергаются частым модификациям при адаптации программы к новому типу оборудования [3].

Для заполнения окна мониторинга данными программа-клиент осуществляет разбор ответов от REST-сервиса в формате JSON.

Описание прототипа симулятора

Симулятор СМТ состоит из HTTP-сервера, REST-сервиса, множества микросервисов мониторинга, графической программы управления параметрами симуляции.

Микросервис-симулятор представляет собой приложение, которое частично воспроизводит функционал реального микросервиса мониторинга. С помощью скрипта в подкаталоге REST-сервиса создаются копии исходного микросервиса-симулятора под разными именами. Специальный конфигурационный файл задает соответствие между именем файла микросервиса и его типом (системный мониторинг ПЛИС, мониторинг подсистемы питания и т.д.).

В каталоге с микросервисами расположены файлы данных мониторинга в формате JSON. Они содержат снимки состояния РВБ на определенный момент времени (кэшированные данные). Файлы заполняются, благодаря выполнению соответствующих HTTP-запросов к REST-сервису РВБ.

При запуске микросервис-симулятор загружает содержимое релевантного файла, чтобы по запросу программы-клиента отдавать кэшированные данные мониторинга. Симулятор отслеживает изменения файла, автоматически считывая обновленные данные.

Для отладки окна с протоколом работы микросервиса симулятор добавляет записи в протокол с заданной частотой. Симулятор циклически изменяет статус работы операции, что приводит к изменению цвета индикатора РВБ в дереве ресурсов программы-клиента. Полученные команды от программы-клиента микросервис сохраняет в файл (протоколирование).

Графическая программа управления симулятором позволяет обновить снимки состояния РВБ за счет опроса массива микросервисов по заданному IP-адресу, а также упрощает перезапуск REST-сервиса.

В дальнейшем планируется расширить функционал симулятора СМТ:

- использовать множество конфигураций РВБ и данных мониторинга с возможностью быстрого переключения между ними;
- отображать данные в формате JSON, передаваемые от программы-клиента (команды), в окне в виде древовидного списка;
- циклически изменять значения физических величин в заданных диапазонах и статусов (флагов ошибок);
- использовать базы данных временных рядов (time series database), помимо обычных JSON-файлов. Это позволяет передавать программе-клиенту массив снимков состояния системы.

Заключение

Внедрение симулятора позволило заметно упростить отладку программы-клиента, т.к. отпадает необходимость в частом подключении к REST-сервису реального РВБ. Уменьшение времени монопольного доступа к РВБ при отладке клиентской программы позволяет другим разработчикам СМТ эффективнее заниматься тестированием микросервисов. Если РВБ собран частично, то недостающие данные можно принудительно добавить в файл мониторинга.

1. **Данилов, И.Г.** Особенности реализации прошивки микроконтроллера модуля загрузки и управления реконфигурируемой вычислительной системы с поддержкой SNMP-телеметрии / И.Г. Данилов // XIV Всероссийская мультиконференция по проблемам управления МКПУ-2021: материалы XIV мультиконференции: в 4 т. – 2021. – С. 243–245.
2. **Каляев, З.В.** Система мониторинга и управления реконфигурируемыми вычислительными системами на основе REST-архитектуры / З.В. Каляев,

И.Г. Данилов, И.Н. Ерошенко // Вестник компьютерных и информационных технологий. – 2019. – № 8(182). – С. 53–60.
3. **Ерошенко, И.Н.** Веб-клиент мониторинга ресурсов РВС на основе технологии webassembly / И.Н. Ерошенко // XIV Всероссийская мультиконференция по проблемам управления МКПУ-2021: материалы XIV мультиконференции: в 4 т. – 2021. – С. 216–218.

Е.И. Духнич[1], А.Г. Чефранов[2], А.П. Шапель[1]

МОДИФИЦИРОВАННЫЙ ШИФР ХИЛЛА В ПРОСТРАНСТВЕ КВАТЕРНИОНОВ

[1]Государственный морской университет имени адмирала Ф.Ф. Ушакова, г. Новороссийск,
[2]Восточно-Средиземноморский университет, г. Фамагуста, Северный Кипр
evgenydukhnich@gmail.com, alexander.chefranov@emu.edu.tr

Введение

В настоящее время развитие алгоритмов цифровой обработки сигналов и изображений происходит с использованием их кватернионных моделей [1,2]. В частности, для представления цветных изображений (например, в формате True Color [3]) применяются кватернионные матрицы с элементами в виде «чистого» кватерниона [4]:

$$S(x,y) = r(x,y)\cdot i + g(x,y)\cdot j + b(x,y)\cdot k. \quad (1)$$

где x, y – координаты пикселя изображения, r, g, b – интенсивности его основных цветов R,G,B, i, j, k – мнимые единицы.

Для обработки таких матриц разработан ряд важных алгоритмов (сингулярное разложение матриц, быстрое преобразование Фурье [4] и другие) в пространстве кватернионов, что приводит к необходимости разработки алгоритмов шифрования непосредственно кватернионных матриц для криптозащиты при их передаче.

В системах реального времени чаще используются симметричные блочные системы, так как они обладают преимуществом над асимметричными в скорости шифрования [5]. Один из обычных способов атаки шифров – использование статистических зависимостей между открытым текстом и результирующим шифротекстом, когда один и тот же ключ применяется к нескольким блокам открытого текста [6]. В отличие от этого варианта в шифроблокноте или одноразовом блокноте (one-time pad (OTP)) [7] длина ключа равна длине сообщения и ключ используется только один

раз, В этом случае атаки невозможны, и такой шифр считается безусловно безопасным. Поэтому практически может быть рассмотрена проблема разработки блочного шифра с достаточно большим переменным размером блока, так что один блок покрывает весь открытый текст, что исключает атаки, основанные на использовании известных пар «открытый текст-зашифрованный текст», полученных с помощью одного и того же ключа.

Известен шифр Хилла [6] как шифр с переменным (но фиксированным) размером блока и четко определенными операциями, однако его проблемы заключаются в квадратичной сложности матричных вычислений и памяти при шифровании / дешифровании, что не позволяет его использовать при ограничениях на время и память. Поэтому необходимы имеющие меньшую сложность блочные криптографические алгоритмы, адаптивные к меняющимся в реальном масштабе времени размерам блоков входных данных.

В шифрах НКР и I-НКР используется произведение Кронекера (КП) матриц [8-9] для формирования ключевых матриц большего размера из элементарных (ЭМ) матриц небольшого размера. Формирование может происходить «на лету» до размера T блока текущего сообщения [10], причем вычислительная сложность и расходы памяти будут соответственно равны $O(TlogT)$ и $O(T)$ в отличие от квадратичной сложности других модификаций шифра Хилла. Шифрование цветного изображения можно делать отдельно для каждого канала. Однако, если шифровать цветное изображение как единое целое (1) с помощью предлагаемых кватернионных ключевых матриц, то размер исходного текста будет в три раза больше, что усиливает криптостойкостью.

Таким образом, задачей данной работы является построение модифицированного шифра Хилла в режиме OTP на основе алгоритма I-НКР [9] в пространстве кватернионов.

Перевод алгоритма I-НКР в пространство кватернионов

Для построения криптосистемы используется шифр Хилла, в котором шифрование выполняется умножением ключевой матрицы M_k на вектор X исходного текста (в пространстве кватернионов):

$$C = M_k X mod N \tag{2}$$

Получаемый шифротекст состоит уже из 4-компонентных кватернионов. Дешифрование выполняется обратным преобразованием:

$$X = M_k^{-1} C mod N \tag{3}$$

Главными проблемами при таком подходе являются фиксированный размер T квадратичной ключевой матрицы и высокая вычислительная сложность $O(T^2)$ умножения матрицы на вектор.

Для построения ключевой матрицы *переменного размера*, равного размеру исходного текста, целесообразно использовать кронекерово произведение *переменного* числа ЭМ $\boldsymbol{A}(\boldsymbol{m},\boldsymbol{n})$ и $\boldsymbol{B}(\boldsymbol{p},\boldsymbol{r})$, которое представляет собой блочную матрицу ($\boldsymbol{m}\cdot\boldsymbol{p}$, $\boldsymbol{n}\cdot\boldsymbol{r}$) размера такую, что

$$(A\otimes B)_{ij} = A_{\left\lfloor\frac{i-1}{p}\right\rfloor+1,(\lfloor j-1)/r\rfloor+1} B_{(i-1)\,mod\,p+1,(j-1)\,mod\,r+1},$$
$$i=\overline{1,mp}, j=\overline{1,nr}, \quad (4)$$

где $\lfloor x\rfloor$ – целая часть x.

В работе [10] предложено построение алгоритма, ориентированного на аппаратную реализацию и ускоряющего процессы формирования КП и умножения вектора на него за счет совмещения их во времени. Таким образом матрица КП в явном виде фактически не рассчитывается и не хранится в памяти. Вместо этого матрицы-сомножители КП итеративно умножаются на компоненты вектора.

Для дополнительного ускорения алгоритма дешифрования (3) при формировании КП применяются инволютивные ключевые матрицы:

$$M_k = M_k^{-1} \quad (5)$$

Это можно сделать, основываясь на следующем свойстве КП [9]: КП инволютивных матриц является инволютивной матрицей.

Ключевая матрица M_k строится как КП порядка $K = log_2 T$, т.е. произведение K элементарных матриц, $\boldsymbol{P}_l$, $l=\overrightarrow{1,K}$:

$$\mathbf{P} = \mathbf{P_1}\otimes(\mathbf{P_2}\otimes\ldots(\mathbf{P_{K-1}}\otimes\mathbf{P_K})\ldots) = \bigotimes_{l=1}^{K}\boldsymbol{P}_l, \quad (6)$$

Для построения ключевой кватернионной матрицы необходимы кватернион-ключи: $f_l = a + b\cdot i + c\cdot j + d\cdot k$, $\{a,b,c,d \in Z_N\}$. Тогда элементарная матрица может иметь вид [8]:

$$P_l = \begin{pmatrix} -f_l & 1-f_l \\ 1+f_l & f_l \end{pmatrix}. \quad (7)$$

Умножение кватернионов в (2)–(3) выполняется не в матричной форме, а в явном виде по формуле:

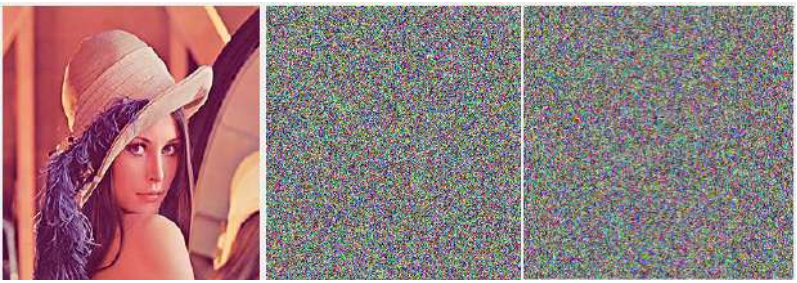
$$(a0+ia1+ja2+ka3)\times(b0+b1+b2+b3) =$$
$$= (a0b0-a1b1-a2b2-a3b3)+i(a0b1+a1b0+a2b3-a3b2)+$$
$$+j(a2b0+a0b2+a3b1-a1b3)+k(a3b0+a0b3+a1b2-a2b1) \quad (8)$$

Умножение кватернионов по формуле (8) требует 16 умножений. Однако, умножения и сложения могут быть выполнены параллельно. Таким образом, время умножения двух катернионов будет практически равно времени умножения двух чисел.

Криптостойкость шифра определяется мощностью множества ключевых кватернионов f_l, $l = \overrightarrow{1,K}$. При $log_2 N = 8$ мощность будет равна $E = 2^{32log_2 T}$. Например, при количестве пикселей изображения $T = 2^{20}$ значение $E = 2^{640}$, что значительно превышает криптостойкость шифров I-HKP ($2^{K(2\log_2 N-1)} = 2^{300}$) и AES ($2^{128}$) [9].

Результаты эксперимента

Ниже представлены результаты шифрования цветного изображения «Lena» (см. рис. 1) с разрешением 256×256, а также результаты подсчета характеристик качества шифрования (см. таблицу 1): среднеквадратическая ошибка (Mean Square Error (MSE)), пиковое отношение сигнала к шуму (Peak Signal to Noise Ratio (PSNR)) [12].



a) b) c)

Рис. 1. Шифрование изображения «Lena» размера 256×256:
a) оригинальное изображение, b) метод I-HKP, c) метод с использованием кватернионной матрицы.

Таблица 1

Результаты вычисления характеристик для цветного изображения «Lena» размера 256×256

Методы шифрования	MSE	PSNR	Вычислительная сложность (умн.)	Время шифрования, мс
Метод I-HKP	9170,79	8,5	3145728	0,1669
Метод с кватернионной матрицей	9186	8,5	16777216	0,6 494

Следует отметить, что существенное повышение вычислительной сложности происходит из-за применения в качестве основной операции умножения кватерниона на кватернион (8), требующего 16 умножений. Параллельный аппаратурно-ориентированный вариант алгоритма (аналогично [10]) разрабатывается.

Выводы

Приведены результаты перевода модифицированного шифра Хилла (I-HKP) в пространство кватернионов: значительно увеличивается длина шифра и мощность пространства ключей, но существенно повышается сложность алгоритма. Параллельная реализация метода позволит сократить время умножения кватернионов практически до времени умножения двух чисел.

1. **Фурман, Я.А.** Комплекснозначные и гиперкомплексные системы в задачах обработки многомерных сигналов / Я.А. Фурман, А.В. Кревецкий [и др.]; под ред. Я.А. Фурмана. – Москва: ФИЗМАТЛИТ, 2004. – 456 с.
2. **Schutte, H.D.** Hypercomplex numbers in digital signal processing / H.D. Schutte, J. Wessel // Proc. Int. Conf. on Circuits and Systems. – 1990. – P. 1557–1560.
3. **Гук, М.** Аппаратные средства IBM PC. Энциклопедия / М. Гук. – СПБ: Питер Пресс, 1999. – 816 с.
4. **Sangwine, S.J.** Fourier transforms of colour images using quaternions or hypercomplex numbers / S.J. Sanqwine // Electronics letters. – 1996. – Vol. 32, Is. 21. – P. 1979–1980.
5. **Алферов, А.П.** Основы криптографии: учебное пособие / А.П. Алферов, А.Ю. Зубов, А.С. Кузьмин, А.В. Черемушкин. – М.: Гелиос АРВ, 2002. – 480 с.
6. **Stallings, W.** Cryptography and network security / W. Stallings. – Principles and practice, 5th ed. – New York: Prentice Hall, 2011. – 412 p.
7. **Borowski, M.** Modern Usage of “Old” One-Time Pad / M. Borowski, M. Leśniewicz // Proceedings of the Military Communications and Information Systems Conference (MCC). – 2012. – P. 1–5.
8. **Chefranov, A.** One-Time Kronecker Product-Based Hill Cipher Modification / A. Chefranov, E. Dukhnich // Journal of Information Assurance and Security. – 2017. – Vol. 12. – P. 94–103.
9. **Chefranov, A**. One-Time Involutory Matrix-Based Hill Cipher Modification / A. Chefranov, E. Dukhnich, A. Shapel // International Journal of Information Assurance and Security (JIAS). – 2020. – Vol.15, Is. 4. – P.165–174.
10. **Духнич, Е.И.** Аппаратурно-ориентированный алгоритм быстрого умножения вектора на кронекерово произведение матриц / Е.И. Духнич, А.Г. Чефранов // Известия ЮФУ. Технические науки. – 2020. – № 7. – С. 45–52.
11. **Bibhudendra, A**. Novel methods of generating self-invertible matrix for Hill Cipher algorithm / A. Bibhudendra, S.R. Girija, K.P. Sarat, K.P. Saroj // International Journal of Security. – 2007. – Vol. 1, Is. 1. – P. 14–21.
12. **Сен, Н.Д**. Применение оценок на основе энтропии для сравнения криптостойкости алгоритмов шифрования / Н.Д. Сен, В.П. Котляров, Я.Ю. Григорьева // Современные наукоемкие технологии. – 2013. – № 2. – С. 105–106.

К.Е. Крамаренко[1,2], А.Ю. Поляков[3], А.В. Ефимов[1,2]

АЛГОРИТМ СИНХРОНИЗАЦИИ ДОСТУПА К ОБЩЕЙ ПАМЯТИ НА ОСНОВЕ БЛОКИРОВКИ MCS*

[1]Институт физики полупроводников СО РАН, г. Новосибирск,
[2]Сибирский государственный университет телекоммуникаций и информатики, г. Новосибирск,
[3]NVIDIA, Santa Clara, California, USA
kostya.kram@gmail.com, artemp@nvidia.com, alexandr.v.efimov@gmail.com

Введение

В архитектурном плане, распределенная вычислительная система (ВС) представляет собой композицию множества элементарных машин (ЭМ) и сети связей между ними [1]. Элементарная машина является основным структурным и функциональным элементом ВС. ЭМ может быть представлена как отдельными процессорными ядрами, так и конфигурациями вычислительных узлов (ВУ), оснащенных универсальными и специализированными сопроцессорами. Количество ЭМ в перспективных ВС экзафлопсного уровня производительности может достигать десятков миллионов.

Основным назначением ВС является обработка вычислительных задач. Такие задачи содержат метаданные с описанием ресурсов ВС, необходимых для их решения, и могут состоять из одной и более параллельных программ (ПП). Запуск задачи состоит из следующих этапов [2-5]:

1) развертывание сервисных процессов среды исполнения (СИ);
2) запуск прикладных процессов ПП формирующих вычислительную задачу;
3) определение параметров статического и динамического компонентов СИ, необходимых для коммуникации прикладных процессов ПП.

Библиотеки управления СИ влияют на эффективность всех этапов решения задач и создаются на основе интерфейса управления процессами (Process Management Interface, PMI). Версия PMI Exascale (PMIx) [6-7] разработана для ВС экзафлопсного уровня производительности и является стандартом де-факто для современных СИ [8-10].

Интерфейс PMI абстрагирует прикладные процессы, формирующие задачу, от деталей реализации СИ. Информация о параметрах СИ согласно PMI хранится в виде базы данных ключ–значение (БДКЗ), доступ к которой осуществляется через примитивы чтения (Get), записи (Put) и синхронизации (Commit, Fence). Размещение БДКЗ в общей памяти (ОП)

* Исследования выполнены в рамках ГЗ 0242–2021–0011 и при поддержке Российского фонда фундаментальных исследований (грант №20–07–00039).

ВУ позволяет исключить системные вызовы и повысить эффективность получения параметров СИ.

В данной работе рассмотрен алгоритм синхронизации доступа к ОП на основе блокировки MCS и приведены результаты экспериментального исследования эффективности доступа к БДКЗ с характерным для PMIx параллельным и интенсивным чтением данных, а также относительно редкими операциями записи.

Доступ к базе данных PMIx

Одним из узких мест существующих реализаций PMI является внутриузловой доступ к БДКЗ. Что приводит к значительному влиянию на производительность примитива Get [5].

Доступ к данным БДКЗ происходит с использованием операций глобальной и локальной (в рамках узла) барьерной синхронизации процессов, а также операций чтения или записи. В процессе примитива синхронизации «Fence», сервисный процесс СИ (поток «PMIx сервер») блокирует ОП для записи (write lock) и размещает в ней содержимое БДКЗ. По завершению синхронизации, прикладные процессы блокируют БДКЗ на чтение (read lock) и получают доступ к параметрам СИ. Если требуемые для выполнения коммуникации параметры отсутствуют в локальной БДКЗ, то блокировка на чтение снимается и выполняется обращение к сервисному процессу СИ для запроса данных, размещенных на других узлах ВС. Прикладные процессы не модифицируют БДКЗ, поэтому возможен параллельный доступ на чтение при условии отсутствия операций записи со стороны сервисного процесса СИ. Обеспечение такого доступа реализуется посредством блокировок чтения-записи [5].

Обзор существующих алгоритмов синхронизации доступа к общей памяти

Рассмотрим подробнее алгоритмы синхронизации, позволяющие обеспечить безопасный доступ к БДКЗ.

В пакете Open PMIx [11] используется схема блокировки 2N-mutex [5], в соответствии с которой для каждого прикладного процесса ПП (PMIx клиента) создаётся пара мьютексов (mutex). Первый мьютекс (LS*x*) используется как сигнальный, а второй (LP*x*) – для доступа к ОП. Заблокировав LP*x*, PMIx клиент освобождает LS*x*. При дступе для записи, сначала осуществляется блокировка всех сигнальных мьютексов (LS1 – LS*N*), которые освобождаются PMIx клиентами максимально быстро. После блокировки сигнальных мьютексов сервисный процесс СИ блокирует основные мьютексы (LP1 – LS*N*), при этом блокировка сигнального мьютекса гарантирует, что новые попытки блокировки на чтение предприниматься не будут.

Преимущество модификации описанного алгоритма под названием *N*(mutex +signal) обеспечивается заменой сигнального мьютекса обычной

переменной (VS*x*). Это позволяет сократить количество атомарных операций, требуемых для блокировки мьютекса на чтение, и снизить накладные расходы времени на выполнение блокировок на чтение и запись.

При доступе для записи выполняется установка всех сигнальных переменных (VS1 – VS*N*) в значение «1». При блокировке мьютекса на чтение, PMIx клиенты сначала проверяют состояние соответствующей сигнальной переменной (VS*x*). До тех пор, пока значение VS*i* равно «1», попытки заблокировать основной мьютекс LP*x* не производится.

Сократить время синхронизации для доступа на запись возможно с применением блокировок, гарантирующих очередность доступа. В данной работе предложена схема синхронизации *N*-MCS, основанная на блокировке MCS [13]. Аналогично схеме 1*N*-mutex в *N*-MCS для каждого PMIx-клиента создаётся индивидуальная блокировка MCS, расположенная в памяти, доступной как прикладному, так и сервисному процессам. Номер блокировки PMIx-клиента соответствует его локальному рангу в рамках ВУ. Как и в 1*N*-mutex, сервисному процессу СИ необходимо заблокировать все индивидуальные блокировки MCS, чтобы получить доступ для модификации БДКЗ.

Блокировка MCS имеет списочную структуру и гарантирует очередность доступа FIFO. В рамках реализации блокировки MCS в данной работе структура блокировки состоит из головного указателя (*head*) и двух списочных записей, соответствующих PMIx-клиенту (индекс *record_idx* = 1) и сервисному процессу (индекс *record_idx* = 2). В головном указателе размещается специальное значение «0», если доступ свободен или индекс записи процесса, заблокировавшего доступ. Каждая запись состоит из двух полей: состояние ожидания (*flag*) и индекс (*next_idx*) следующего процесса, ожидающего блокировки, или «0», если ожидающих блокировки процессов нет. В реализации MCS в данной работе не предполагается наличие более одного процесса, ожидающего блокировки.

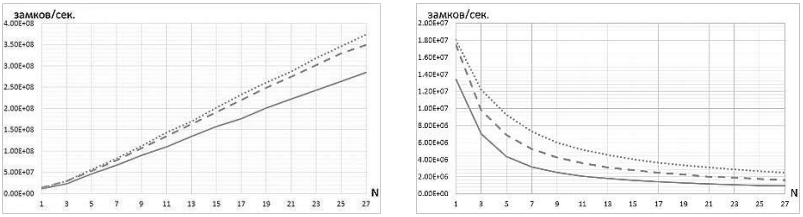
Блокировка MCS основана на атомарной операции fetch-and-store (atomic_swap). В head записывается *record_idx* претендента, а в качестве результата возвращается индекс процесса, уже заблокировавшего доступ, или «0» успешной блокировки. В случае неуспешной блокировки претендент переходит в фазу ожидания. При этом в записи претендента в поле *flag* записывается «1», а в записи «владельца» в поле *next_idx* сохраняется индекс претендента. Далее у претендента запускается цикл опроса флага ожидания, который будет изменен при разблокировке MCS.

Разблокировка MCS основана на атомарной операции *compare-and-swap* (CAS). Если в головном указателе обнаружен индекс записи текущего «владельца», то претендентов на владение нет и блокировка снята. При этом головной указатель уже установлен в специальное значение «0». Если

операция CAS не выполнена, то поле *next_idx* в записи текущего «владельца» содержит индекс «претендента». В этом случае блокировка передается путем записи «0» в поле *flag* в записи «претендента».

Результаты и заключение

Результаты экспериментального исследования оценки производительности алгоритмов синхронизации доступа к ОП с БДКЗ, представленные на рис. 1 получены при помощи микро-теста [13].



а б

Рис. 1. Микро-бенчмарк [14] производительности схем блокировки в режимах: (а) только запись; (б) только чтение
—— - 2N-mutex; – – – - N(mutex+signal); ····· - N-MCS

В работе рассмотрены алгоритмы синхронизации *N*(mutex+signal) и *N*-MCS, адаптированные для интенсивного доступа к общей памяти со значительным преобладанием операций чтения над операциями записи.

Уменьшение числа атомарных операций в предложенных схемах синхронизации позволило увеличить пропускную способность примерно на 20 % в режиме «только чтение». В режиме «только запись» наблюдается двух-трехкратное увеличение пропускной способности.

1. **Хорошевский, В.Г.** Архитектура вычислительных систем. / В.Г. Хорошевский. – М.: МГТУ им. Баумана, 2008. – 520 с.
2. **Yu, W.** Fast and Scalable Startup of MPI Programs in InfiniBand Clusters / W. Yu, J. Wu, D.K. Panda // High Performance Computing – HiPC 2004. 11th International Conference. Proceedings. – 2004. – P. 440–449.
3. **Balaji, P.** PMI: A Scalable Parallel Processmanagement Interface for Extreme-scale Systems / P. Balaji, D. Buntinas, D. Goodell [et al.] // Proceedings of the 17th European MPI Users' Group Meeting Conference on Recent Advances in the Message Passing Interface (EuroMPI'10). – 2010. – P. 31–41.
4. **Castain, R.H.** PMIx: Process Management for Exascale Environments / R.H. Castain, D. Solt, J. Hursey, A. Bouteiller // Proceedings of the 24th European MPI Users' Group Meeting (EuroMPI'17). – 2017. – Article 14.

5. **Polyakov, A.** A Performance Analysis and Optimization of PMIx-based HPC Software Stacks / A. Polyakov, B. Karasev, J. Hursey // EuroMPI '19: Proceedings of the 26th European MPI Users' Group Meeting. – 2019. – Article 9.
6. PMIx Consortium. 2017–2019. Process Management Interface for Exascale (PMIx) Standard (version 3.1) [электронный ресурс]. – URL: https://github.com/pmix/pmix-standard/releases/download/v3.1/pmix-standard-3.1.pdf (дата обращения: 10.04.2022)
7. PMIx Consortium. 2017-2019. PMIx Reference Library [электронный ресурс]. – URL: https://github.com/pmix/pmix (дата обращения: 29.05.2020)
8. Argonne National Laboratory. 2014. Hydra Process Management Framework [электронный ресурс]. – URL: https://wiki.mpich.org/mpich/index.php/Hydra_Process_Management_Framework (дата обращения: 10.04.2020)
9. PMIx Consortium. 2017–2018. PMIx-based Reference RunTime Environment (PRRTE) [электронный ресурс]. – URL: https://github.com/pmix/prrte (дата обращения: 10.04.2022)
10. SchedMD, LLC. 2017. SLURM Workload Manager [сайт]. – URL: https://slurm.schedmd.com/ (дата обращения: 10.04.2022)
11. Open PMIx master repository [электронный ресурс]. – URL: https://github.com/openpmix/openpmix (дата обращения: 10.04.2022).
12. **Mellor-Crummey, J.** Algorithms for scalable synchronization on shared-memory multiprocessors / J. Mellor-Crummey, M. Scott // ACM Trans. Comput. Syst. – 1991. – № 9. – P. 21–65.
13. Микро-бенчмарк [электронный ресурс]. – URL: https://bitly.su/Wtxe0Z (дата обращения: 10.04.2022).

А.С. Лебедев

ПОСТРОЕНИЕ ПРОСТРАНСТВЕННЫХ И ВРЕМЕННЫХ ПРЕОБРАЗОВАНИЙ ПРОГРАММ ДЛЯ РАСПАРАЛЛЕЛИВАНИЯ ВЛОЖЕННОСТЕЙ ЦИКЛОВ

Центр информационных технологий в проектировании РАН,
г. Одинцово,
tementy@gmail.com

Введение

Эффективное программирование параллельных архитектур всегда было сложной задачей, и особенно усложняется при возрастающих требованиях к быстродействию современного программного обеспечения с объемной кодовой базой, созданной в процессе многолетней разработки. Одним из подходов к решению обозначенных проблем является автоматическое распараллеливание программ, исключающее усилия программиста по анализу потока данных в программе, выявлению параллелизма, синтезу параллельного вычислительного кода.

Наибольшая вычислительная трудоемкость сосредоточена в циклических конструкциях. Особый интерес для исследователей представляют программы линейного класса [1, с. 320], встречающиеся в научных и инженерных приложениях, которые тратят значительную часть времени именно на исполнение вложенностей циклов. Модель многогранников [2] предоставляет мощный математический аппарат, упрощающий анализ и преобразование таких программ с целью улучшения быстродействия путем повышения степени параллелизма и улучшения локальности использования данных при вычислениях.

Научная задача, решаемая в работе – разработка методов построения пространственных и временных преобразований программ линейного класса для кластеров многопроцессорных NUMA-машин, обеспечивающих повышение быстродействия вычислений при распараллеливании вложенностей циклов.

Построение преобразований программ линейного класса

Обобщенный граф зависимостей – это ориентированный мультиграф, каждая вершина которого представляет множество соответственных операций (динамических экземпляров одной и той же инструкции). Каждая операция принадлежит только одной вершине. Ребро графа представляет временное ограничение на операции, соответствующие начальной и конечной вершине (принцип причинности для информационно зависимых операций u и v: $u \to v \Rightarrow \theta(v) > \theta(u)$, где θ – логическое время). Каждой инструкции X соответствует домен – многогранник D_X на множестве $\mathbb{Q}^{p_X}$, где p_X – количество циклов в программе, включающих X. Каждая операция представляется как $\langle \vec{\imath}, \vec{z}; X \rangle$, где $\vec{\imath} \in D_X$ – целочисленный вектор индекса итерации, $\vec{z}$ – целочисленный вектор внешних переменных программы, имеющий размерность q_z. Все операции над памятью, которые требуется выполнить, формируют множество Ω. Каждому ребру e, исходящему из вершины $\sigma(e) = X$ в $\delta(e) = Y$, ставится в соответствие многогранник зависимостей R_e на множестве $\mathbb{Q}^{p_X + p_Y}$ такой, что условие причинности должно быть наложено на операции $\langle \vec{\imath}, \vec{z}; X \rangle$ и $\langle \vec{\jmath}, \vec{z}; Y \rangle$ тогда и только тогда, когда составной вектор $\begin{bmatrix} \vec{\imath} \\ \vec{\jmath} \end{bmatrix} \in R_e$. Пусть f и g – индексные функции ячейки памяти, к которой обращаются конфликтующие операции, тогда p_e называют глубиной зависимости:

$$\begin{bmatrix} \vec{x} \\ \vec{y} \end{bmatrix} \in R_e \equiv f(\vec{x}) = g(\vec{y}) \wedge (\vec{x}[1..p_e] = \vec{y}[1..p_e]) \wedge (\vec{x}[p_e + 1] < \vec{y}[p_e + 1]).$$

Пусть E – множество ребер обобщенного графа зависимостей. Тогда d-мерное ($d \geq 1$) расписание вычислений [3] есть функция $\theta: \Omega \to \mathbb{N}_0^d$ такая, что $\forall e \in E, \vec{x} \in D_{\sigma(e)}, \vec{y} \in D_{\delta(e)}$:

$$\begin{bmatrix} \vec{x} \\ \vec{y} \end{bmatrix} \vec{x} \in R_e \Rightarrow \theta(\langle \vec{y}, \vec{z}; \delta(e) \rangle) >_{lex} \theta(\langle \vec{x}, \vec{z}; \sigma(e) \rangle).$$

Множество операций $F(t) = \{u \in \Omega | \theta(u) = t\}$ называется фронтом расписания на t. Для каждой зависимости e вычисляется многогранник P_e и аффинное преобразование h_e такое, что:

$$\begin{bmatrix} \vec{x} \\ \vec{y} \end{bmatrix} \in R_e \equiv (\vec{x} = h_e(\vec{y}) \wedge \vec{y} \in P_e).$$

Размещение вычислений есть функция $\pi: \Omega \to \mathbb{N}_0$, которая ставит в соответствие операции номер виртуального процессора, на котором она будет исполняться. Пусть $\langle g_a, \vec{z}; A_a \rangle$ – доступ к массиву A_a с аффинной индексной функцией $g_a(\vec{\imath}), \vec{\imath} \in D_X$ в некоторой позиции a инструкции X. Тогда $\eta(\langle g_a, \vec{z}; A_a \rangle)$ – номер виртуального процессора, на котором располагается элемент $A_a[g_a(\vec{\imath})], \vec{\imath} \in D_X$.

Определение 1. Одномерное аффинное отображение для инструкции X есть функция вида $\phi_X(\vec{\imath}, \vec{z}) = \vec{v}_X \cdot \vec{\imath} + \vec{v}'_X \cdot \vec{z} + v_X^0, \quad v_X^0 \in Z, \vec{v}_X \in Z^{p_X}, \vec{v}'_X \in Z^{q_z}, \quad \vec{\imath} \in D_X$. Многомерное аффинное отображение для инструкции X задается конечным набором (вектором) одномерных отображений.

Рассмотрим функцию $\chi_e(\vec{y}, \vec{z}) = \phi_{\delta(e)}(\vec{y}, \vec{z}) - \phi_{\sigma(e)}(h_e(\vec{y}), \vec{z})$, $\vec{y} \in P_e$. Если отображения $\phi_{\delta(e)}(\vec{y}, \vec{z})$ и $\phi_{\sigma(e)}(h_e(\vec{y}), \vec{z})$ представляют расписание вычислений, то $\chi_e(\vec{y}, \vec{z})$ – количество отсчетов логического времени, проходящего между исполнением двух инструкций $\sigma(e)$ и $\delta(e)$, связанных зависимостью, то есть мера задержки использования данных $d_e^\tau(\vec{z})$. Если отображения представляют размещение вычислений, то $\chi_e(\vec{y}, \vec{z})$ – расстояние в пространстве виртуальных процессоров межу процессорами, выполняющими две инструкции $\sigma(e)$ и $\delta(e)$, связанные зависимостью, то есть мера расстояния использования данных $d_e^\rho(\vec{z})$.

Пусть размещение данных задается явно. Рассмотрим функцию $\nu_a(\vec{\imath}, \vec{z}) = \left| \pi_X(\vec{\imath}, \vec{z}) - \eta_{A_a}(g_a(\vec{\imath}), \vec{z}) \right|, \quad \vec{\imath} \in D_X$ для доступа к памяти $\langle g_a, \vec{z}; A_a \rangle$ в некоторой позиции a инструкции X. $\nu_a(\vec{\imath}, \vec{z})$ – расстояние в пространстве виртуальных процессоров межу процессором, выполняющим инструкцию X, и процессором, владеющим элементом данных, к которому осуществляется доступ в позиции a инструкции X, то есть мера расстояния использования данных $d_a^\rho(\vec{z})$.

Минимизация задержки и расстояния использования данных приводит к улучшению временной и пространственной локальности использования данных соответственно. Величины $d_e^\tau(\vec{z})$, $d_e^\rho(\vec{z})$, $d_a^\rho(\vec{z})$ желательно ограничить постоянным значением. Однако, это не всегда возможно. Доказаны следующие утверждения.

Утверждение 1. Если домены $D_{\delta(e)}$ и $D_{\sigma(e)}$ являются ограниченными, и заданы два аффинных отображения $\phi_{\delta(e)}(\vec{y},\vec{z})$ и $\phi_{\sigma(e)}(h_e(\vec{y}),\vec{z})$, то найдется аффинная функция $L_e(\vec{z})$, зависящая только от внешних переменных программы, такая, что $L_e(\vec{z}) - \chi_e(\vec{y},\vec{z}) \geq 0, \quad \vec{y} \in P_e$.

Утверждение 2. Если домен D_X является ограниченным, и заданы два аффинных отображения: $\pi_X(\vec{\imath},\vec{z})$ и $\eta_{A_a}(g_a(\vec{\imath}),\vec{z})$ для некоторой позиции a инструкции X, то найдется аффинная функция $L_a(\vec{z})$, зависящая только от внешних переменных программы, такая, что $L_a(\vec{z}) - \nu_a(\vec{\imath},\vec{z}) \geq 0, \quad \vec{\imath} \in D_X$.

Следовательно, найдутся функции $L_e^\tau(\vec{z}) \geq d_e^\tau(\vec{z})$, $L_e^\rho(\vec{z}) \geq d_e^\rho(\vec{z})$ и $L_a^\rho(\vec{z}) \geq d_a^\rho(\vec{z})$. Степень временной (пространственной) локальности использования данных тем выше, чем меньше величина $L_e^\tau(\vec{z})$ ($L_e^\rho(\vec{z})$ или $L_a^\rho(\vec{z})$). Оптимальное расписание программы минимизирует целевую функцию $\sum_{e\in E} \alpha_e L_e^\tau(\vec{z})$, где $\alpha_e \geq 0$ – весовые коэффициенты, отражающие предпочтения относительно качества оптимизации для каждой зависимости по данным. Аналогично, оптимальное размещение программы минимизирует целевую функцию $\sum_{e\in E} \alpha_e L_e^\rho(\vec{z})$. В работе применяется $\alpha_e = \#P_e$, где $\#P_e$ – количество точек с целочисленными координатами внутри многогранника зависимостей P_e. Для случая явно заданного размещения данных оптимальное размещение вычислений и данных, вычисленное совместно, минимизирует целевую функцию $\sum_a \alpha_a L_a^\rho(\vec{z})$, где $\alpha_a \geq 0$ – весовые коэффициенты, отражающие предпочтения относительно качества оптимизации для каждого обращения к памяти. В работе применяется $\alpha_a = \#D_{S_a}$, где $\#D_{S_a}$ – количество точек с целочисленными координатами внутри домена D_{S_a}.

Разработанные методы были применены к трем реализациям алгоритмов линейной алгебры: LU-разложение квадратной матрицы, матричные произведения atax и syr2k. Каждая программа была распараллелена с помощью OpenMP для выполнения на одном компьютере, и с помощью MPI для выполнения на кластере. Вычисления производились с двойной точностью. Эксперименты проводились на двух машинах под управлением ОС Linux CentOS 7.3 x64. Машина 1 имеет Intel Xeon CPU E5-2690 v2, 8GB RAM. Машина 2 имеет Intel Xeon CPU E5-2640 v3, 4GB RAM. Сеть передачи сообщений – InfiniBand 40Gb/sec. Запуск OpenMP-программ осуществлялся на машине 1. Сравнивалась производительность параллельной программы, полученной применением разработанной системы и современного компилятора Pluto 0.11.4 [4]. Ускорение

оценивалось относительно запуска последовательного варианта программы на машине 1. В таблице 1 приведены результаты распараллеливания трех линейных программ. Варианты параллельных программ получены применением разработанной системы (помечены ILP) и компилятора Pluto (помечены Pluto). Параллельный код запускался для исполнения в 2, 4, 6, 8 потоках. В экспериментах с OpenMP работали множественные потоки внутри одного процесса, запущенного на машине 1. В экспериментах с MPI работали единственные потоки внутри множественных MPI-процессов, запущенных на обеих машинах (количество процессов разделялось поровну между машинами).

Таблица 1

Ускорение вычислений при распараллеливании программ

Код	Потоки	OpenMP			MPI		
		ILP	Pluto	ILP/Pluto	ILP	Pluto	ILP/Pluto
lu	2	1,48	1,22	21,58%	1,62	1,47	10,38%
	4	2,08	1,60	29,50%	2,26	2,32	-2,54%
	6	2,27	1,82	24,59%	2,48	2,71	-8,47%
	8	2,35	1,85	26,85%	2,56	2,94	-13,06%
atax	2	1,15	0,14	746,78%	0,97	0,04	2210%
	4	1,24	0,26	381,11%	1,05	0,05	1888%
	6	1,27	0,37	243,60%	1,09	0,06	1884%
	8	1,21	0,51	138,87%	1,21	0,05	2190%
syr2k	2	26,96	26,66	1,13%	34,87	21,26	64,03%
	4	45,57	44,80	1,70%	53,03	34,51	53,69%
	6	62,90	61,92	1,59%	61,68	47,78	29,09%
	8	79,88	83,03	-3,78%	63,97	59,70	7,14%

Заключение

Экспериментальные результаты исследования производительности параллельных вариантов программ lu, atax, syr2k, полученных с помощью разработанного программного обеспечения и современного распараллеливающего компилятора Pluto, свидетельствуют о выигрыше в быстродействии до 64%, достигаемом благодаря применению разработанного метода построения преобразований программы.

Теоретическая значимость работы состоит в доказанных утверждениях для линейных программ, предложенных критериях оптимальности и методе построения их преобразований, а именно: доказанных утверждениях о верхних границах задержки и расстояния использования данных для линейных программ; критериях оптимальности пространственных и временных преобразований программ с точки зрения локальности

использования данных; методе построения пространственных и временных преобразований линейных программ для распараллеливания вложенностей циклов, ориентированном на оптимизацию локальности использования данных; методе генерации параллельной MPI-программы, позволяющем организовать информационный обмен между параллельными процессами в случае явно заданного размещения данных. Практическая значимость работы заключается в: возможности реализации разработанных методов в системах трансляции программного кода (компиляторах, интерпретаторах, JIT-подсистемах сред выполнения) для поддержки распараллеливающих преобразований наряду с улучшением локальности использования данных при выполнении линейных программ на вычислительных системах, построенных на основе универсальных многоядерных процессоров; разработанных компонентах автоматически распараллеливающего компилятора, поддерживающих распараллеливание программ, написанных на языке Си.

1. **Воеводин, В.В.** Параллельные вычисления. / В.В. Воеводин. – БХВ-Петербург, 2004.
2. **Lengauer, C.** Loop parallelization in the polytope model / C. Lengauer // International Conference on Concurrency Theory. – 1993. – P. 398–416.
3. **Feautrier, P.** Some efficient solutions to the affine scheduling problem. I. One-dimensional time / P. Feautrier // International journal of parallel programming. – 1992. – Vol. 21, № 5. – P. 313–347.
4. **Bondhugula, U.** Automatic transformations for communication-minimized parallelization and locality optimization in the polyhedral model / U. Bondhugula // International Conference on Compiler Construction. – 2008. – P. 132–146.

Э.В. Мельник, М.В. Орда-Жигулина, Д.В. Орда-Жигулина

ИНТЕГРАЦИЯ КОМПОНЕНТОВ СИСТЕМЫ МОНИТОРИНГА И ДИАГНОСТИКИ ПРИ РЕШЕНИИ ЗАДАЧ РАЗЛИЧНОЙ НАПРАВЛЕННОСТИ*

Южный научный центр РАН, г. Ростов-на-Дону,
evm17@mail.ru, jigulina@mail.ru, dinazhigulina@mail.ru

В рамках географически распределенных современных информационных систем существует необходимость объединения участников, относящихся к различным областям мониторинга и диагностики, каждый из которых имеет свой набор различных данных для мониторинга и диагностики. В этом случае каждый участник должен обладать своим комплексом методов и алгоритмов сбора, обработки и хранения данных, а взаимодействие всех

* Исследование выполнено при финансовой поддержке ГЗ 122020100270–3.

участников системы может обеспечиваться посредством комплекса методов и алгоритмов взаимодействия участников без их иерархической подчиненности. При этом необходимо разрабатывать комплекс инструментов как для мониторинга, так и для организации самой системы (включая ассистентов пользователя с элементами искусственного интеллекта).

Для решения задач организации информационной структуры систем мониторинга и диагностики авторами работы был разработан комплекс методов интеллектуальной обработки данных на базе распределенного реестра, метод комплексирования информации, метод робастно устойчивого управления при решении задач мониторинга мультиробототехническими комплексами и набор компонентов системы мониторинга и прогнозирования. Структура системы при этом является децентрализованной, масштабируемой и «открытой», поэтому в зависимости от области применения системы, ее можно дополнять различными гетерогенными компонентами по мере необходимости.

Авторами работы был определен следующий перечень компонентов: модуль интеллектуального анализа данных (на базе аппарата когнитивного моделирования), модель распределенного хранилища данных и мультиробототехнический комплекс [1-9]. Затем была проведена интеграция этих компонентов в прототип децентрализованной распределенной подсистемы поддержки принятия решений, как показано на рис. 1.

Рис. 1. Интеграция моделей компонентов системы мониторинга и прогнозирования для различных областей человеческой деятельности.

Принципы совместной работы мультиробототехнического комплекса и прототипа модели распределенного хранилища данных реализованы с помощью технологии распределенного реестра и технологий робототехники. Интеграция всех компонентов проводилась на базе аппарата когнитивного анализа и технологий интеллектуального анализа данных и была реализована в виде подсистемы принятия решений.

Выводы

Была осуществлена интеграция прототипов моделей распределенного хранилища данных, модуля интеллектуального анализа и мультироботехнического комплекса в прототип модели системы мониторинга и прогнозирования для различных областей применения на базе технологий цифровой экономики. Были проведены экспериментальные исследования работоспособности и эффективности разработанных методов и алгоритмов для каждого из прототипов модулей.

Полученные научные результаты позволяют разрабатывать распределенные системы мониторинга и диагностики без централизации функции управления, реализовывать расширенные возможности масштабирования, обеспечить повышенную отказоустойчивость и повысить надёжность систем мониторинга и диагностики. Также полученные результаты охватывают широкий спектр типов систем мониторинга и диагностики, что обуславливает возможность их применения в различных областях деятельности.

1. **Карелин, В.П.** Интеллектуальные технологии и системы искусственного интеллекта для поддержки принятия решений / В.П. Карелин // Вестник Таганрогского института управления и экономики. – 2011. – № 2.
2. **Melnik, E.V.** Distributed Library Model Based on Distributed Ledger Technology for Monitoring and Diagnostics System / E.V. Melnik, M.V. Orda-Zhigulina, D.V. Orda-Zhigulina // Lecture Notes in Networks and Systems. – 2021. – Vol. 230. – P. 501–509. DOI: 10.1007/978-3-030-77442-4_43.
3. **Kapustyan, S.G.** Model of Multi-robotic Complex at the Base of Distributed Registry for Monitoring and Diagnostics System / S.G. Kapustyan, D.V. Orda-Zhigulina, M.V. Orda-Zhigulina [et al.] // Lecture Notes in Networks and Systems. – 2021. – Vol. 228. – P. 659–669. – DOI: 10.1007/978-3-030-77448-6_64.
4. **Kapustyan, S.G.** Control of the Group of Mobile Robots Without Preplanned Route for Monitoring and Diagnostic Purposes / S.G. Kapustyan, M.V. Orda-Zhigulina, D.V. Orda-Zhigulina // Lecture Notes in Networks and Systems. – 2021 – Vol. 232. – P. 945–953. – DOI: 10.1007/978-3-030-90318-3_75.
5. **Gorelova, G.** The Problem Statement of Cognitive Modeling in Social Robotic Systems / G. Gorelova, E. Melnik, I. Safronenkova // Lecture Notes in Computer Science. – 2021. – Vol. 12998. – P. 62–75. DOI: 10.1007/978-3-030-87725-5_6.
6. Программа для когнитивного анализа и анализа социально-экономических систем регионального уровня / Г.В. Горелова // Свидетельство о государственной регистрации программы для ЭВМ № 2018661506; заявл. 07.09.2018.

7. **Мунтян, Е.Р.** Исследование структурных характеристик распределенных вычислительных систем на основе графов с множественными разнотипными связями / Е.Р. Мунтян, Э.В. Мельник // Известия ЮФУ. Технические науки. – 2021. – № 3.– С. 186–198 DOI: 10.18522/2311-3103-2021-3-186-198.
8. **Мельник, Э.В.** Разработка элементов распределенной библиотеки на базе распределенного реестра для систем мониторинга и диагностики / Э.В. Мельник, И.С. Пуха, М.В. Орда-Жигулина, Д.В. Орда-Жигулина // Известия Тульского государственного университета. Технические науки. – 2021. – № 2. – С. 349–357.
9. **Мельник, Э.В.** Метод повышения надежности за счет реконфигурации ресурсов в системах мониторинга и диагностики опасных природных явлений / Э.В. Мельник, М.В. Орда-Жигулина, Д.В. Орда-Жигулина, А.А. Родина // Известия ТулГУ. Технические науки. – 2020. – № 2. – С. 18–26.

А.К. Мельников

УТОЧНЕНИЕ КЛАССИФИКАЦИИ ТИПОВ ДИСКРЕТНЫХ РАСПРЕДЕЛЕНИЙ ДЛЯ ОРГАНИЗАЦИИ ЭФФЕКТИВНОЙ ОБРАБОТКИ ПОСЛЕДОВАТЕЛЬНОСТЕЙ НА МНОГОПРОЦЕССОРНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

АО «Вычислительные решения», г. Москва,
ak@comp-sol.ru

Введение

Постоянное увеличение масштабов информатизации и цифровизации всех видов деятельности ведет к постоянному увеличению количества данных, передаваемых, в сетях связи и ГИС Internet. Обработка этих данных требует применения вычислительных ресурсов все большего объема, которые могут быть предоставлены только многопроцессорными вычислительными системами (МВС) различных архитектур [1], значительную часть из которых составляют реконфигурируемые вычислительные системы (РВС), позволяющие адаптировать свою архитектуру под структуру решаемой задачи [2], что определяет наибольшую скорость её решения.

Большинство задач автоматической обработки данных могут быть сведены к задаче обработки потоков дискретных данных, состоящих из дискретных последовательностей, в дальнейшем просто последовательностей или выборок.

Для увеличения скорости обработки потоков последовательностей при применении двухэтапной модели обработки [3], состоящей из предварительного отбора и углубленной обработки отобранных последовательностей, необходимо на этапе отбора максимально снизить число ложно принятых решений о справедливости проверяемых гипотез, что

возможно при применении статистических критериев с наибольшей эффективностью [4].

Постановка задачи

Пусть N мощность алфавита $A_N = \{a_1, a_2, \dots, a_N\}$, $|A_N| = N$ и n – объём (длина) выборки (последовательности) $x = x_{N,n}$, состоящей из знаков алфавита A_N, $x_{N,n} = \{x_1 = a_{i_1}, x_2 = a_{i_2}, \dots, x_n = a_{i_n}\}$., принимаемыми с вероятностями $P_N = \{p_1, p_2, \dots, p_N\}$, где j-й знак выборки x принимает значение i-го знака алфавита A_N с вероятностью значение $\{p_i = P\{x_j = a_i\} \mid \forall j = \overline{1,n}, \ \forall i = \overline{1,N}\}$.

Если $S_{N,n}$ является статистикой, т.е. некоторой функцией от $x_{N,n}$ и P_N: $S_{N,n}(x_{N,n}, P_N) = g(x, P_N)$, тогда под её распределение $P\{S_{N,n} \geq c\}$ [5] на некотором выборочном пространстве X понимается вероятность события

$$P\{S_{N,n} \leq c\} = \sum_{x \in X} I(g(x, P_N) \leq c) \cdot P\{x \in X\}, \tag{1}$$

где $P\{x \in X\}$ есть вероятность принадлежности выборки x к выборочному пространству $\{X | x \in X\}$.

Одновременно при решении статистических задач активно используются предельные распределения [6] не зависящие от значений статистики $g(x)$ и $P\{x \in X\}$, а зависящие от мощности алфавита N и предельных свойств функции статистики $g(x)$. Для многих основных используемых на практике статистик таким предельным распределение является распределение хи-квадрат χ^2 [7].

Данная работа посвящена уточнению классификации типов дискретных распределений, применяющихся при анализе последовательностей и определению границ их применения на множестве параметров выборок в зависимости от величины предоставляемого вычислительного ресурса. Целью применения различных типов распределений является построение критерия отбора последовательностей, на базе которого разрабатывается метода обработки последовательностей, обладающего наибольшей возможной эффективностью.

Уточненная классификация типов дискретных распределений

Множество всех возможных выборок $x = x_{N,n}$ объема n в алфавите A_N образует полное множество элементарных событий, обозначим его через $\{X_{N,n} | x_{N,n} \in X_{N,n}\}$. При этом справедливо, что для любой выборки $y_{N,n}$ объема n из алфавита A_N, существует одна и только одна выборка $x_{N,n}$, принадлежащая множеству $X_{N,n}$, такая что $y_{N,n} = x_{N,n}$

$\{\forall y_{N,n},\ \exists x_{N,n} \in X_{N,n}$ такое, что $y_{N,n} = x_{N,n}\}$, мощность этого множества $|X_{N,n}| = N^n$. Если $X = X_{N,n}$, то можно считать, что при расчете вероятности $P\{S_{N,n} \le c\}$ (1) учтены все возможные значения статистики $S_{N,n}$ и вероятность (1) можно назвать точным распределением и обозначить через $P_{\mathrm{T}}\{S_{N,n} \le c\}$

$$P_T\{S_{N,n} \le c\} = \sum_{x_{N,n} \in X_{N,n}} I(S_{N,n}(x_{N,n}, P_N) \le c) P\{x_{N,n} \in X_{N,n}\}. \qquad (2)$$

Но расчет точного распределения является вычислительно трудоёмкой задачей за счет большого количества элементов полного выборочного пространства $X_{N,n}$ и возможен для очень ограниченного количества параметров.

Для увеличения значений параметров выборок N и n, для которых могут быть рассчитаны распределения (2), предлагается сократить выборочное пространство с $X_{N,n}$ до $X'_{N,n}$, так чтобы $|X_{N,n}| \ge |X'_{N,n}|$. Вычисленные на сокращенном выборочном пространстве $X'_{N,n}$ распределения предлагается назвать точными приближениями распределений или просто точными приближениями.

Предложена методика [8] сокращения выборочного пространства с $X_{N,n}$ до $X'_{N,n}$, основанная на оценке вероятности статистики максимальной частоты, позволяющая сократить выборочное пространство расчета точного распределения до $X'_{N,n} = X^{\Delta}_{N,n}$. При этом, вычисленное с помощью (2) но на пространстве $X^{\Delta}_{N,n}$ распределение $P_{\Delta}\{S_{N,n} \le c\}$

$$P_{\Delta}\{S_{N,n} \le c\} = \sum_{x_{N,n} \in X^{\Delta}_{N,n}} I(S_{N,n}(x_{N,n}, P_N) \le c) P\{x_{N,n} \in X^{\Delta}_{N,n}\} \qquad (3)$$

будет отличаться от точного распределения $P_T\{S_{N,n} \le c\}$ на задаваемую заранее сколь угодно малую величину Δ

$$|P_T\{S_{N,n} \ge c\} - P_{\Delta}\{S_{N,n} \ge c\}| \le \Delta,$$

в соответствии с принятой терминологией назовём полученное таким образом точное приближение – Δ-точным приближением, указывающим на задаваемую величину его отличия от точного распределения.

Предельные распределения [6], например хи-квадрат χ^2 распределение $P\{\chi^2_{(N-1)} \ge c\}$ назовём предельными приближениями и обозначим через $P_{\lim}\{S_{N,n} \le c\} = P\{\chi^2_{(N-1)} \ge c\}$.

Предлагается следующая уточненная классификация дискретных распределений, состоящая из таких типов распределений как точные распределения, точные (Δ-точные) и предельные приближения.

Описание областей применения распределений различных типов

Эффективность обработки $\omega(\alpha,\ P\{S_{N,n} \geq x\})$, определяет количество ложно отобранных последовательностей при применении в критерии отбора распределения $P\{S_{N,n} \geq x\}$, определяющего через задаваемый размер критерия α разделяющую константу c: $P\{S_{N,n} \geq c\} = \alpha$.

Границы областей параметров применения различных типов распределений обуславливаются вычислительной сложностью их расчетов: $C(P_T\{S_{N,n} \geq c\})$ [9] для точных распределений и $C(P_\Delta\{S_{N,n} \geq c\})$ [10] для Δ-точных приближений, выделенным вычислительным ресурсом BP и определяются следующими соотношениями

$$C(P_T\{S_{N,n} \geq c\}) = BP, \qquad C(P_\Delta\{S_{N,n} \geq c\}) = BP \cdot \tag{4}$$

Нижняя граница области применения предельных приближений определена из гипотезы Р. Фишера [11] $n = 5 \cdot N$.

Разработка обобщенного статистического метода анализа последовательностей

Основная идея метода ОСМА состоит в выборе распределения $P\{S_{N,n} \geq x\}$, используемого при построении критерия отбора последовательностей в качестве распределения эталонной статистики. В соответствии с параметрами выборки N и n рассчитывается вычислительная сложность получения точного распределения $C(P_T\{S_{N,n} \geq c\})$ [9: 127] и точного приближения $C(P_\Delta\{S_{N,n} \geq x\}$ [10: 42]. В соответствии выделенным вычислительным ресурсом BP определяем применяемое в критерии распределение $P\{S_{N,n} \geq x\}$

$$P\{S_{N,n} \geq x\} = \begin{cases} P_T\{S_{N,n} \geq x\}, \textit{если } C(P_T\{S_{N,n} \geq c\}) \leq BP \\ P_\Delta\{S_{N,n} \geq x\}, \textit{если } C(P_\Delta\{S_{N,n} \geq c\}) \leq BP \\ P\{\chi^2_{(N-1)} \geq x\}, \qquad \textit{противном случае} \end{cases}$$

В [6: 129] доказывается, что эффективность ОСМА будет во всех случаях не хуже эффективности любого другого статистического метода, а в вычисленных примерах в 4 и 2,5 раза больше [6: 122, 124].

Заключение и выводы

В работе уточняются типы дискретных распределений вероятностей значений статистик, проводится их классификация. Вводятся понятия точного распределения, точного и предельного приближений распределения.

Определяются границы областей параметров выборок, для которых без потери эффективности могут быть применены определенные типы дискретных распределений.

Предложенная уточнённая классификация дискретных распределений выборок позволила непрерывно на всем множестве параметров выборок определить непересекающиеся области применения различных типов распределений, позволяющих строить статистические критерии с наибольшей возможной эффективностью.

Критерий отбора последовательностей, обладающий наибольшей эффективностью, строится с помощью применения в критерии согласия, в зависимости от параметров последовательностей, различных типов распределений.

На основе критерия отбора строится обобщенный статистический метод анализа и обработки последовательностей, обладающий наибольшей возможной эффективностью из всех других методов данного класса.

1. **Муравьёв, С.А.** История отечественной электронной вычислительной техники / С.А. Муравьёв; под общей редакцией директора Департамента радиоэлектронной промышленности Минпромторга России С.В. Хохлова. – Москва: ООО «Издательский дом «Столичная энциклопедия», 2017. – 680 с.
2. **Гузик, В.Ф.** Реконфигурируемые вычислительные системы: учебное пособие / В.Ф. Гузик, И.А. Каляев, И.И. Левин; под общей редакцией И.А. Каляева. – Таганрог: Издательство Южного федерального университета, 2016. – 472 с.
3. **Мельников, А.К.** Применение точных распределений в процедуре двухэтапной обработки текстов / А.К. Мельников // Обозрение прикладной и промышленной математики. – 2018. – Т. 25, Вып. 2. – С. 89–95. – ISSN 0869–8325.
4. **Мельников, А.К.** Сравнение эффективности обработки текстов при применении в статистических критериях точных и предельных приближений базовых распределений вероятностей значений тестовых статистик / А.К. Мельников // Обозрение прикладной и промышленной математики. – 2018. – Т. 25, Вып. 4. – С. 375–378. – ISSN 0869–8325.
5. **Феллер, В.** Введение в теорию вероятностей и её приложения / В. Феллер. – в 2-х томах: – Т.1; пер. с англ. – Москва: Мир, 1984. – 528 с.
6. **Мельников, А.К.** Применение точных и предельных приближений распределений вероятностей значений статистик при решении задачи обработке текстов / А.К. Мельников // Известия ЮФУ. Технические науки. Тематический выпуск. Суперкомпьютерные технологии. – 2018. – № 8 (202). – С. 114–135. – ISSN 1999–9429. DOI: 10.23683/2311–3103-2018-8-114-135.
7. **Крамер, Г.** Математические методы статистики. / Г. Крамер. – Москва: Мир, 1975. – 648 с.

8. **Мельников, А.К.** Методика расчета распределения вероятностей значений симметричных аддитивно разделяемых статистик, приближенных к их точному распределению / А.К. Мельников // Научный вестник НГТУ. – 2018. – № 1 (70). – С. 153–166. – ISBN 1814–1196. DOI: 10.17212/1814–1196-2018-1-153-166.
9. **Мельников, А.К.** Сложность расчета точных распределений вероятности симметричных аддитивно разделяемых статистик и область применения предельных распределений / А.К. Мельников // Доклады ТУСУР. – 2017. – Т. 20, № 4, – С. 126–130. – ISSN 1818–0442.
10. **Мельников, А.К.** Выбор метода расчета точных приближений дискретных распределений вероятностей значений статистик / А.К. Мельников // Вестник компьютерных и информационных технологий. – 2021. – Т. 18, № 6 (204). – С. 39–48. DOI 10.14489/vkit.2021.06.pp.039-048.
11. **Fisher, R.A.** Statistical methods for research workers / R.A. Fisher // Edinburgh: Oliver and Boyd. – 12th Ed. – 1954. – 356 p.

А.В. Подопригора

ОБЪЕДИНЕНИЕ БАЗОВЫХ БРН-МАТРИЧНЫХ МАКРООПЕРАЦИЙ

Южный федеральный университет, г. Таганрог,
apodoprigora@sfedu.ru

Одной из актуальных тенденций информационных технологий является создание цифровых прототипов и моделей в разных сферах технической и научной деятельности с использованием вычислительных средств. Характерные сложность, неоднообразность и несимметричность формы моделируемых объектов и ряд других факторов непременно ведут к увеличению вычислительной трудоемкости обработки таких моделей. Применение для такого рода задач универсальных методов и средств решения зачастую является крайне неэффективным.

В процессе анализа сложных прикладных задач, связанных с разработкой моделей технических комплексов, естественных, социальных и экономических моделей было выявлено, что соответствующие задачам матрицы имеют достаточно сложный вид. Была выделена группа матриц, для которых характерным признаком является большая размерность, сильная разреженность, которая характеризуется большим количеством незначащих элементов и отсутствием видимой функциональной структуры по расположению значимых элементов матрицы. В ходе исследования такие матрицы обозначаются как большие разреженные неструктурированные матрицы (БРН-матрицы).

Дальнейший разбор методов и средств, предназначенных для обработки БРН-матриц кластерными вычислительными системами с традиционной архитектурой процессора, показал, что для больших размерностей с числом строк порядка 105 производительность снижается в 5–7 раз. Такой вывод подтверждается множеством работ, посвященных модификации алгоритмов распределения загрузки вычислительных ядер и методов хранения данных, результаты которого показаны на рис. 1 [1].

Снижение производительности таких вычислительных систем возникает из-за комплекса пересекающихся факторов. Большая размерность обрабатываемых матриц обеспечивает большую интенсивность обменов данными между вычислительными ядрами и памятью, неструктурированность и разреженность обрабатываемой матрицы не позволяют равномерно загружать вычислительные ядра. В сумме эти причины ведут к увеличению времени подготовки операнда и простою ядер вычислительной системы.

Рис. 1. График зависимости производительности кластерных ВС от размера обрабатываемой БРН-матрицы.

В рамках исследования предлагается использовать РВС на основе ПЛИС-технологий как основной тип вычислительных средств, поскольку они позволяют применять более адаптивные подходы к организации вычислений. Широкие возможности архитектуры РВС позволяют организовать структуру, которая будет с большей эффективностью обрабатывать БРН-матрицы [2].

Поскольку на данном этапе эффективных методов для обработки БРН-матриц для РВС не существует, и тема является нераскрытой, возникает необходимость разработать специальные методы конвейерной обработки. Использование таких методов для РВС позволит повысить эффективность обработки относительно методов, которые в большинстве своем разрабатывались и используются для кластерных вычислительных систем с традиционной архитектурой процессора.

Для эффективной обработки БРН матриц необходимо исключить незначащие элементы матрицы из процесса хранения и обработки. Решение этих задач выполняется за счет использования специального формата хранения данных, который предполагает хранение только значимых элементов БРН-матрицы, и дискретно-событийной организации потоков данных [3].

Одной из основных целей обработки БРН-матриц является решения СЛАУ, где основная матрица системы – большая разреженная и неструктурированная. Для таких матриц используются итерационные методы решения, поскольку прямые методы решения предполагают приведение матрицы к треугольной форме, вследствие чего матрица с большой вероятностью будет преобразована к плотной форме, и объем вычислений существенно увеличится. Базовая итерация метода решения СЛАУ рассматривается как функция над тремя и более БРН-матрицами. Например, для метода решения СЛАУ методом простых итераций в качестве базовой итерации выполняется функция вычисление невязки.

Для эффективной реализации решения СЛАУ методом простых итераций необходимо использовать метод распараллеливания по итерациям [4], с применением которого структура задачи преобразуется в цепь последовательно соединенных структур базовой итерации, представленной в виде функции над тремя и более БРН-матрицами. Использование метода позволяет изменить критический ресурс при реализации структуры задачи в базисе РВС, которым являлся ресурс доступа к памяти, и максимально задействовать вычислительного ресурса кристалла.

Для эффективной реализации на РВС базовой итерации, представленной в виде функции над тремя и более БРН-матрицами, необходимо использовать разрабатываемый автором метод распараллеливания по базовым макрооперациям. Этот метод позволит повысить производительность вычислительной системы выполняемой функции за счет минимизации скважности обработки данных, которая появляется из-за вероятностных процессов, вызванных неструктурированностью операндов. В результате применения метода однородная структура функции преобразуется в группу связанных базовых макроопераций, в каждую из которых включены вычислительная часть и локальное управление потоками данных.

В результате такого разделения структуры задачи на базовые макрооперации необходимо обеспечить объединение базовых макроопераций между собой внутри многоместных функций и на стыке базовых итераций в итерационных методах решения СЛАУ. Процесс объединения заключается в том, чтобы обеспечить одинаковую интенсивность потоков данных на входах и выходах каждой базовой макрооперации. В процессе анализа были выделены категории базовых макроопераций: по количеству входных операндов и характеру изменений структуры матрицы. В табл. 1 показаны характерные примеры таких базовых макроопераций.

Таблица 1

Типизация базовых макроопераций над БРН-матрицами

Количество БРН-матриц	**Изменение адреса**	**Изменение данных**	**Изменение адреса и данных**
Одна БРН-матрица	Транспонирование БРН-матрицы	Скалярные операции БРН-матрицами	–
Две БРН-матрицы	–	Суммирование БРН-матриц	Умножение БРН-матриц

Объединение базовых макроопераций с одной матрицей с другими типами базовых макроопераций не нуждается в дополнительных подходах, поскольку скорость обработки данных базовой макрооперацией над одной БРН-матрицей соответствует интенсивности входного потока данных.

Необходимость особого подхода к объединению возникает для базовых макроопераций, скорость обработки которых ниже интенсивности входного потока данных, что характерно для базовых макроопераций с двумя БРН-матрицами. Для компенсации этой разности на каждой базовой макрооперации используются входные буферные элементы, размер которых определяться из расчетов вероятности потери элемента методами сетевых технологий [5].

Помимо использования буферизации входных потоков данных, может быть использовано интегрирование базовых макроопераций в базовую макрооперацию с большим временем обработки. В случае с предсуммированием для операции умножения необходимо скорректировать глубину входного буферного элемента для базовой макрооперации умножения. Поскольку характер обработки данных на базовой макрооперации суммирования предполагает формирование результирующего элемента каждый такт, то интенсивности обработки входных и выходных данных равны. В случае с пост-суммированием для операции умножения необходимо применять дополнительные действия по включению суммирования в процесс умножения. Это необходимо, поскольку количество вычислений для получения каждого элемента произведения требует больше операций, чем для базовой макрооперации суммирования, ввиду чего предлагается в процессе выполнения алгоритма получения элемента произведения проводить анализ наличия значащего элемента в суммируемой матрице и добавление его к результату произведению. При совпадении позиций необходимо включить значение суммируемой матрицы в аккумулятор частичных произведений вычисляемого элемента. Так, в обоих

рассмотренных случаях время выполнения функции из базовых макрооперaций умножения с суммированием будет определяться временем умножения двух БРН-матриц.

Наиболее важным для организации обработки БРН-матриц является соединение двух и более базовых макрооперaций умножения. Такой тип соединения является основой не только для объединения базовых макрооперaций внутри многоместной функции, но и объединением базовых итераций в рамках решения СЛАУ итерационными методами. Для выполнения операции умножения выполняется вариация метода распараллеливания по итерациям, сам процесс умножения нескольких матриц происходит следующим образом. Базовая макрооперация умножения принимает в качестве первого операнда одну строку БРН-матрицы, второй операнд предполагает полный проход второй БРН-матрицы по столбцам. Результатом одного такого прохода БРН-матрицы второго операнда является одна строка, которая передается как первый операнд для следующей макрооперации умножения БРН-матриц. В следующий проход вторые операнды БРН-матриц для текущей и следующей базовых макрооперaций будут выполняться одновременно. Таким образом, время перемножения n-го числа БРН-матриц будет примерно равняться времени умножения двух БРН-матриц.

Достижение эффективности вычислений 50% и выше для задачи обработки БРН-матриц на РВС обеспечивается использованием следующих компонентов: специального формата хранения данных, дискретно-событийной организации потоков данных для базовых макрооперaций БРН-матрицами, методов распараллеливания по операциям и по итерациям. Ключевым подходом в организации вычислительных структур для обработки БРН матриц является разработка подхода объединения базовых макрооперaций между собой, которая возникает из-за необходимости приведения скорости обработки данных к интенсивности входных потоков данных. Использование для этого буферных элементов оптимального размера и интеграции вычислений в более вычислительно трудоемкие базовые макрооперации делает возможным достижение наибольшей эффективности, которая доступна для разработанных методов.

1. **Yang, C.** Design Principles for Sparse Matrix Multiplication on the GPU / C. Yang, A. Buluc, J. Owens // International European Conference on Parallel and Distributed Computing. – 2018. – arXiv:1803.08601v2
2. **Kalyaev, I.A.** FPGA-based reconfigurable computer systems / I.A. Kalyaev, I.I. Levin; A.I. Dordopulo, L.M. Slasten // Science and Information Conference. – 2013 – SAI 2013. – P. 148–155.
3. **Подопригора, А.В.** Метод организации дискретно-событийных вычислений для обработки больших разреженных неструктурированных матриц на

РВС / А.В. Подопригора // Известия ЮФУ. Технические науки. – 2021. – № 7(207) – С. 189–197. DOI 10.18522/2311–3103–2021–7–189–197.
4. **Каляев, А.В.** Модульно-наращиваемые многопроцессорные системы со структурно-процедурной организацией вычислений / А.В. Каляев, И.И. Левин. – М.: Янус-К, 2003. – 380 с.
5. **Коннов, А.Л.** Методы расчета показателей производительности сетей ЭВМ с неоднородным трафиком: монография / А.Л. Коннов, Ю.А. Ушаков. – Оренбург: ОГУ, 2013. – 139 с.

Е.А. Титенко

МЕТОД ПАРАЛЛЕЛЬНО-КОНВЕЙЕРНОГО ПОИСКА ПО ОБРАЗЦУ

Юго-Западный государственный университет, г. Курск,
johntit@mail.ru

Введение

Современный этап развития микропроцессоров и специализированных вычислительных средств характеризуется повышением уровня интеллектуализации вычислений, направленных на снижение временных затрат за счет аппаратной поддержки нетиповых операций и процедур. Особенно актуально созданием однородных вычислительных структур под информационный граф задачи с аппаратной поддержкой базовых операций [1-4].

Математические, программные и технические средства обработки информации в значительной степени ориентируются на проблемно-поисковые и комбинаторные задачи, связанные с принятием решений в условиях неопределенности исходных данных и схемы решения. Их эффективное решение возможно, в том числе, на основе аппарата продукционных систем [5-7], отличающегося модульностью структуры, параллелизмом срабатывания правил (продукций). Тем не менее, по мнению специалистов в области моделей и методов искусственного интеллекта (ИИ) около 80 % времени в аппаратных и программных средствах (процессоры логического вывода, машины баз данных, экспертные и советующие системы языки ИИ, лингвопроцессоры) уходит на процессы сопоставления с образцом сравнения с эталоном, проверкой текущих данных по шаблону. При этом данные преимущественно имеют не числовое, а символьное представление в виде слов, строк, текстов, гипертекстов и др. [8].

В рамках ПС эти процессы получили типовое название «поиск по образцу». Процессы поиска по образцу преимущественно основаны на

переборно-возвратной схеме сопоставлений символов двух операндов (образец и текст), в которой каждое положительное сравнение приводит к инкременту индекса символа в структуре проявляемого слова, а отрицательное сравнение влечет декремент индекса такого символа. Другими словами, вычислительный процесс поиска включает прямые и возвратные передвижения по длине символьных операндов в процессе сравнений символов.

Наиболее эффективные методы последовательного поиска по образцу приведены в трудах Кнута-Морриса-Пратта, Бойера-Мура, Хорспула, Сандея, Гоннета, Бейза-Ейтса и др. В данных методах сокращение времени достигается за счет построения и применения таблиц переходов, таблиц прыжков, учета предыстории сравнений в динамической памяти, что позволяет вести не тотальный, а направленный поиск. Тем не менее, исходные операнды рассматриваются как одномерные конструктивные объекты, трудоемкость вычислительных процессов на которых пропорциональна длине операндов. Ограничение на количество одновременно рассматриваемых символов и одномерный вид операндов приводят к непродуктивным затратам времени поиска по образцу. Увеличение длины операндов приводит к большему числу сравнений из-за наличия возвратных переходов при отрицательных сопоставлениях.

Данное противоречие определяет актуальным создание оригинальных методов параллельного поиска по образцу, использующих двумерное пространственно-временное расположение символов операндов для их множественных сравнений и выдачи результата как массива вхождений образцов.

Постановка задачи

Пусть заданы символьные операнды следующим образом. В рабочем алфавите $\boldsymbol{A}$ заданы слово O – образец длиной n символов и слово S – текст длиной m символов, где $n>0$, $m>0$ и $n \leq m$.

Требуется найти позиции (адреса) всех вхождений образца O в структуре текста S. Другими словами требуется разработать метод и аппаратно-ориентированный алгоритм, формирующий вектор вхождений $\bar{\imath} = (i_1, i_2 \ldots i_z)$, что справедливо

$$\forall i_l (O(1,n) = S(i_l, i_l + n - 1) \mid 1 \leq i \leq k, l = 1 \ldots z, k = m - n + 1, \quad (1)$$

где i_l – адрес l-го вхождения l – номер вхождения, z – количество вхождений, k – общее количество сопоставлений символов образца и текста.

Метод решения

Наиболее перспективным типом параллельного поиска по образцу являются метод матричного поиска, задающий множество простейших поисково-вычислительных процессов (потоков) по диагоналям матрицы сопоставлений символов образца *O* и текста *S*. В каждой диагонали матрицы длины *n* и количеством *m*-*n*+1 вычислительные ячейки сравнивают (*ij*)-ые символы с учетом отношения следования символов (*i*=1 …*n*, *j*=1…*m*).

Исходя из практики решения прикладных задач, особенно в экспертных системах, советующих системах, системах поиска в полнотекстовых базах и др. длина образца как правило имеет верхний порог, исчисляемый единицами-десятками символов. В противоположность этому длина текста, в котором ищется образец, представляется строками длиной тысячи-десятки тысяч символов.

Применение метода и алгоритмов матричного поиска без учета ограничений на длину текста приводит к многократному тиражированию символов (тысячи-десятки раз) в матрице сопоставлений символов, что приводит к существенным аппаратным затратам и снижению тактовой частоты из-за «длинных» связей между вычислительными ячейками.

Для снятия данных ограничений предлагается параллельно-конвейерный метод и соответствующий ему аппаратно-ориентированный алгоритм поиска по образцу.

Главная особенность предлагаемого метода заключается в разбиении исходной строки (текста) *S* на *q* подстрок (фрагментов) по *f* символов. Последняя подстрока дополняется фиктивными символами «х», которые не входят в рабочий алфавит. Текст *S* представляется как цепочка из *q* подстрок в виде

$$S = f_1 \cdot f_2 \ \dots \ f_q \qquad (2)$$

где f_1 – *i*-я подстрока из *S*, $i = 1 - q$.

Поиск по образцу реализуется за *q* шагов, на каждом шаге реализуется параллельный поиск по текущей подстроке, но с учетом частичных вхождений от предыдущей подстроки. Каждый шаг поиска сопровождается подсчетом количества обработанных подстрок. Для этого вводится счетчик обработанных подстрок, он используется в дальнейшем для формирования весового двоичного кода.

Перед началом поиска задаются:

– счетчик обработанных подстрок, *Count=0*;

– равносторонняя треугольная матрица начальных значений, *Edge*={0, 0, … 0}. Эта треугольная матрица имеет размеры сторон

$(n-1) \times (n-1)$ элементов, состоит из хранит частичные вхождения от предыдущего шага поиска в виде «коротких» диагоналей.

Метод параллельно-конвейерного поиска состоит из следующих шагов.

1. Поиск на основе множественных парных сравнений символов образца и символов текущей подстроки. Формируется двоичная прямоугольная матрица сопоставлений символов $M=\{m_{ij}\}$ (i=1 … f), j=1 …n}, содержащая $k=f-n+1$ диагоналей. Поиск по образцу в пределах текущей подстроки реализуется проверкой логических «1» по всем элементам k диагоналей матрицы и по элементам (n-1) дополнительных диагоналей. Таким образом, поиск выполняется по расширенному количеству диагоналей, равному $f-n+1+n-1=f$.

2. Запись найденных позиций вхождений в k+(n-1) триггеров позиций.

3. Сохранение k+(n-1) найденных позиций и счетчика обработанных подстрок Count во внешнюю память.

4. Инкремент счетчика обработанных подстрок Count. Сброс k+(n-1) триггеров позиций.

5. Перезапись *Edge Matrix* в состав прямоугольной матрицы сопоставлений символов M

$$M = M \cup Edge, \tag{3}$$

Перезапись *Edge Matrix* выполняется в верхнюю правую часть прямоугольной матрицы M, она необходима для следующего шага поиска.

6. Если $\mathrm{Count} \leq q$, тогда запись следующей подстроки из текста S. Переход к п.1. Следующий шаг шага поиска сопровождается записью новой подстроки и переходом на п. 3 алгоритма.

7. Параллельно-конвейерный поиск завешен, когда обработано q подстрок из текста S.

Обсуждение и результаты

Метод параллельно-конвейерного поиска реализуется за q итераций, реализуя вычисления по текущей подстроке с учетом граничных значений от предыдущей подстроки. Метод матричного поиска за одну итерацию реализует параллельный поиск по всем диагоналям матрицы с фиксацией позиций вхождений по длине текста S.

Исходя из этапов методу поиска по образцу время матричного поиска (за 1 итерацию) складывается из времени записи в параллельном коде строки-образца O – t_O, времени записи в параллельном коде текста

S – t_S собственно времени поиска $T_{search} = nt_{CELL}$, времени сохранения результатов – t_{tr}

$$T_{Matrix} = t_O + t_S + T_{search} + t_{tr}. \quad (4)$$

Время параллельно-конвейерного поиска (за q итераций) начально описывается как

$$T_{Pipeline} = t_O + q\left(t_S + T_{search} + t_{wr_edge} + t_{ct2} + t_{tr}\right) \quad (5)$$

где t_{wr_edge} – время перезаписи матрицы Edge, t_{ct2} – инкремент счетчика подстрок.

Эффект конвейеризации достигается за счет совмещения времени записи подстроки и перезаписи матрицы Edge, т.е. слагаемым $\max\left(t_S, t_{wr_edge}\right)$.

Принимая,

$$t_O = t_S = t_{wr_{edge}} = t_{ct2} = t_{reg},\ T_{search} = nt_{CELL} = n(2treg + 4\tau), \quad (6)$$

где t_{CELL} – время работы вычислительной ячейки, τ – логический вентиль, итоговые формулы времени поиска имеют вид

$$T_{Matrix} = t_{reg}(3 + 2n) + 4n\tau, \quad (7)$$

$$T_{Pipeline} = t_{reg}(1 + 3q + 2nq) + 4nq\tau. \quad (8)$$

Согласно (8) время поиска пропорционально количеству итераций q. Оценка аппаратной сложности матриц по обоим методам составляет

$$\theta_{Matrix} = mnz_{CELL} + mz_{tr} + nz_{tr} + (m - n + 1)z_{tr}, \quad (9)$$

$$\theta_{Pipeline} = fnz_{CELL} + n(n + 1)/2 \cdot z_{CELL} + fz_{tr} + nz_{tr} + fz_{tr}, \quad (10)$$

Основные аппаратные затраты приходятся на сложность матриц с вычислительными ячейками. Согласно (9) аппаратная сложность матрицы прямо пропорциональна длине m текста S. Сложность матрицы в параллельно-конвейерном поиске по (10) определяется длиной f подстроки, что существенно меньше, чем по (9). Дополнительное слагаемое в виде ячеек не зависит от длины текста m.

Заключение

Создан метод, сочетающий преимущества параллельного поиска и имеющий примерно в q раз меньшую аппаратную сложность, что позволяет создавать эффективные однородные устройства.

1. **Левин, И.И.** Реконфигурируемые компьютеры на основе ПЛИС XILINX VIRTEX UTRASCALE / И.И. Левин, А.И. Дордопуло, Д.А. Сорокин, З.В. Каляев, Ю.И. Доронченко // XIII Междунар. научная конференция «Параллельные вычислительные технологии (ПаВТ'2019)». – 2019. – С. 288–298.
2. **Левин, И.И.** Эффективная реализация распараллеливания на реконфигурируемых системах / И.И. Левин, А.В. Пелипец // Вестник компьютерных и информационных технологий. – 2018. – № 8 (170). – С. 11–16.
3. **Levin, I.I.** Reconfigurable computer systems: from the first FPGAS towards liquid cooling systems / I.I. Levin, A.I. Dordopulo, A.M. Fedorov, I.A. Kalyaev // Supercomputing Frontiers and Innovations. – 2016. – Vol. 3, Is.1. – P. 22–40.
4. **Гузик, В.Ф.** Реконфигурируемые вычислительные систем / В.Ф. Гузик, И.А. Каляев, И.И. Левин. – Ростов-на-Дону: Южный федеральный университет. – 2016. – 472 с.
5. **Титенко, Е.А.** Стратегии быстрых символьных вычислений для исчислительной продукционной системы / Е.А. Титенко, В.М. Довгаль, В.С. Титов // Известия ВУЗОВ. Приборостроение. – 2008. – Т. 51, № 2. – С. 44–47.
6. **Гривачев, А.В.** Модифицированная продукционная система для решения задачи структурного распознавания образов / А.В. Гривачев, С.Г. Емельянов, Е.А. Титенко // Наукоемкие технологии. – 2014. – Т. 15, № 12. – С. 9–12.
7. **Титенко, Е.А.** Однородные вычислительные структуры для параллельных символьных вычислений / Е.А. Титенко, С.Г. Емельянов, И.С. Зерин // Известия Юго-Западного государственного университета. – 2011. – № 6 (39). Ч. 2 – С. 77–82.
8. **Рыбина, Г.В.** Интегрированные экспертные системы / Г.В. Рыбина // Приборы и системы. Управление, контроль, диагностика. – 2011. – № 4. – С. 22–47.

А.Г. Ченцов, А.М. Григорьев, А.А. Ченцов

МАРШРУТИЗАЦИЯ ПЕРЕМЕЩЕНИЙ ПРИ УСЛОВИЯХ ПРЕДШЕСТВОВАНИЯ: МЕТОДЫ И АЛГОРИТМЫ*

Институт математики и механики имени Н. Н. Красовского Уральского отделения РАН, г. Екатеринбург,
chentsov@imm.uran.ru, ag@uran.ru, chentsov_a_a@mail.ru

Рассматриваются задачи последовательного обхода мегаполисов (непустых конечных множеств) при наличии ограничений в виде условий предшествования; предполагается, что оптимизируется аддитивный критерий, в формировании которого участвуют функции стоимости, допускающие зависимость от списка заданий. В качестве основного метода решения используется широко понимаемое динамическое программирование (ДП), восходящее в идейном отношении к [1] (отметим здесь же другой вариант ДП в работе [2]). Применяемый авторами вариант ДП соответствует в существенном [3, раздел 9.4] и, в более полном варианте, [4] (имеется целый ряд других публикаций). Проводимая ниже конкретизация общей задачи маршрутизации ориентирована на применения в атомной энергетике: имеется в виду проблема снижения дозовой нагрузки ликвидаторов при последовательном демонтаже радиационно опасных элементов (см. [5,6] и др.); имеются в виду аварии, подобные Чернобылю и Фукусиме, а также процессы демонтажа энергоблоков, выведенных из эксплуатации. На основе теоретических методов построены алгоритмы, реализованные на ПЭВМ и МВС; приведены примеры решения модельных задач.

Рассматриваем непустое множество, в пределах которого допускаются перемещения с выделенным непустым конечным подмножеством (п/м) возможных точек старта. Предполагается заданным конечный набор непустых конечных п/м исходного множества, именуемых далее мегаполисами. Посещение каждого мегаполиса сопровождается выполнением тех или иных работ, именуемых внутренними. Внешние перемещения, перемещения при выполнении внутренних работ и терминальное состояние оцениваются с помощью специальных функций, допускающих зависимость от списка заданий, не выполненных на текущий момент. Требуется выбрать оптимально маршрутный процесс, включающий перестановку индексов мегаполисов (то есть очередность посещений), именуемую ниже маршрутом, траекторию по-

* Работа выполнена в рамках исследований, проводимых в Уральском математическом центре при финансовой поддержке Министерства науки и высшего образования Российской Федерации (номер соглашения 075-02-2022-874).

сещения мегаполисов и точку старта. Исследуются также более частные задачи с фиксацией точки старта. В задачах большой размерности применяются оптимизирующие вставки и мультивставки.

В настоящей работе общие процедуры решения задач маршрутизации перемещений рассматриваются применительно к задаче о минимизации дозовой нагрузки ликвидаторов при последовательном демонтаже радиационно опасных источников. Предполагается (см. [7]), что с каждым таким источником связывается мегаполис, получаемый дискретизацией границы ближней зоны; возможны также варианты, когда источник находится в помещении, а мегаполис порождается в модели системой входов-выходов. Во всех случаях сам источник содержится в выпуклой оболочке мегаполиса, создаваемого в модели. Предполагается, что внешние перемещения осуществляются с большей скоростью в сравнении с перемещениями при посещении мегаполисов (по смыслу – внутренними, связанными с работами по демонтажу). Условия предшествования имеют самую различную природу. Так, в частности, в числе демонтируемых объектов могут быть насосы и трубопроводы; насосы следует заглушить для работы с трубопроводами, что уже определяет условия предшествования.

Степень воздействия источника на ликвидатора в процессе перемещения последнего определяется интегрированием зависимости, обратно пропорциональной квадрату расстояния. Зависимость от списка заданий, не выполненных на момент перемещения, имеет ясный смысл: «светят» те и только те источники, которые на данный момент еще не демонтированы.

Общая схема исследования на основе ДП включает:

1) построение модели мегаполисов и формализацию задачи; определение маршрутного процесса и математическую постановку задачи;
2) расширение исходной задачи до системы частичных подзадач с заменой допустимости по предшествованию допустимостью по вычеркиванию заданий из списка;
3) построение алгоритма на основе динамического программирования.

Вычислительный эксперимент

1. Пример с ПЭВМ (N=35).

Решается незамкнутая задача о демонтаже 35 источников излучения с расположением исполнителя во время демонтажа на заданном удалении от источника. Заданы мегаполисы, являющиеся 6-элементыми множествами в виде равномерных «сеток» на окружностях, и источники излучения внутри контуров соответствующих окружностей. Имеется 8 источников излучения, не подлежащих демонтажу данным исполнителем: {(-70,-75);(-5,-70);(-10,15);(80,40);(90,5);(90,-80);(40,-55);(-70,45)}.

Задано множество возможных начальных точек маршрута: {(90,35);(0,0);(90,-35);(-90,35);(-50,35);(-70,-100)}

Эвакуация исполнителя производится из одной из заданных точек множества (выбор точки эвакуации подлежит оптимизации):
{(10,70);(-40,-45);(-10,-50);(95,-10);(40,50);(90,-100);(-100,35);(-100,-75)}

Количество адресных пар (условия предшествования): 50.

Скорость перемещения вне мегаполисов в 4 раза больше, чем при перемещении при выполнении внутренних работ и при покидании мегаполиса.

Интенсивности излучения подлежащих демонтажу источников находятся в пределах от 1.3 до 5.5.

Интенсивности излучения не подлежащих демонтажу источников находятся в пределах от 1.4 до 5.5.

Время выполнения демонтажа находится в пределах 1.1 до 1.7.

Размеры (радиусы) «ближних» зон источников излучения в пределах от 1.0 до 1.4.

Результаты счета программы:

- величина совокупных затрат: 261.329;
- выбрана начальная точка: (-50,35);
- выбрана точка эвакуации: (40,50);
- первый пункт маршрута: ((-43,61);(-43,61)) из М30х М30;
- последний пункт маршрута: ((30,-0);(30,-0)) из М35х М35;
- время вычисления: 42 час. 35 мин. 46 сек.

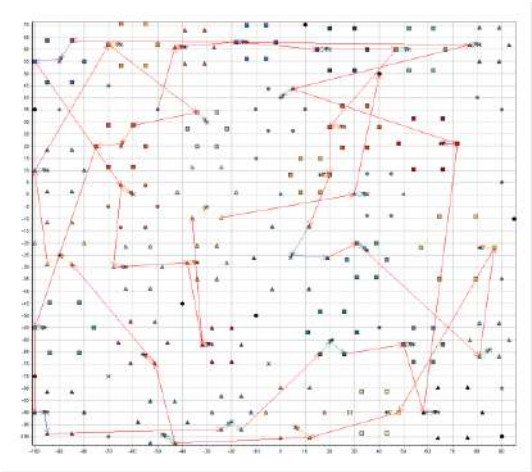


Рис. 1. Пример оптимального решения задачи на ПЭВМ (N=35).

2. Пример оптимального решения на МВС (N=55).

Решается незамкнутая задача о демонтаже 55 источников излучения с расположением исполнителя во время демонтажа на заданном удалении

от источника. Заданы мегаполисы, являющиеся 20-элементыми множествами в виде равномерных «сеток» на окружностях, и источники излучения внутри контуров соответствующих окружностей.

Задано множество возможных начальных точек маршрута: {(0,15);(-50,50);(-50,-50);(35,-35);(35,50);(95,70);(0,0)}

Количество адресных пар (условия предшествования): 70.

Скорость перемещения вне мегаполисов в 4 раза больше, чем при перемещении при выполнении внутренних работ и при покидании мегаполиса.

Интенсивности излучения подлежащих демонтажу источников находятся в пределах от 1.3 до 5.5.

Размеры (радиусы) «ближних» зон источников излучения в пределах от 4.0 до 6.0.

Результаты счета программы:

– величина совокупных затрат: 123.069;
– выбрана начальная точка: (0,15);
– время вычисления: 27 мин. 49 сек.

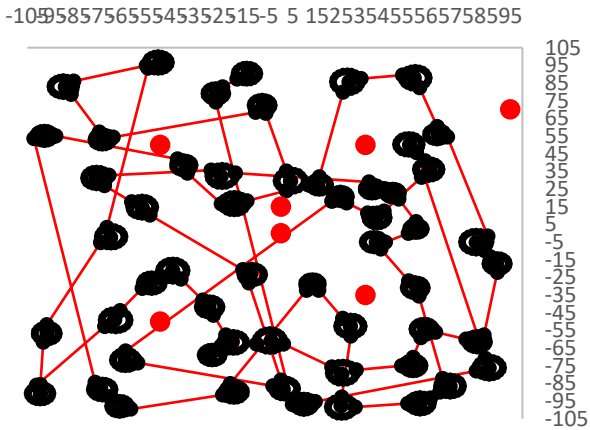


Рис. 2. Пример оптимального решения задачи на МВС (N=55).

3. Пример решения «большой» задачи с использованием оптимизирующей мультивставки на МВС (N=255).

Решается незамкнутая задача о демонтаже 255 источников излучения с расположением исполнителя во время демонтажа на заданном удалении

от источника. Заданы мегаполисы, являющиеся 20-элементыми множествами в виде равномерных «сеток» на окружностях, и источники излучения внутри контуров соответствующих окружностей.

Количество адресных пар (условия предшествования): 10.

Скорость перемещения вне мегаполисов в 4 раза больше, чем при перемещении при выполнении внутренних работ и при покидании мегаполиса.

Интенсивности излучения подлежащих демонтажу источников находятся в пределах от 1.3 до 5.5.

Размеры (радиусы) «ближних» зон источников излучения в пределах от 4.0 до 6.0.

Размер оптимизирующей вставки равен 20 мегаполисам.

Результаты счета программы:

– величина совокупных затрат: 277.339;
– время вычисления: 5 час. 53 мин. 46 сек.

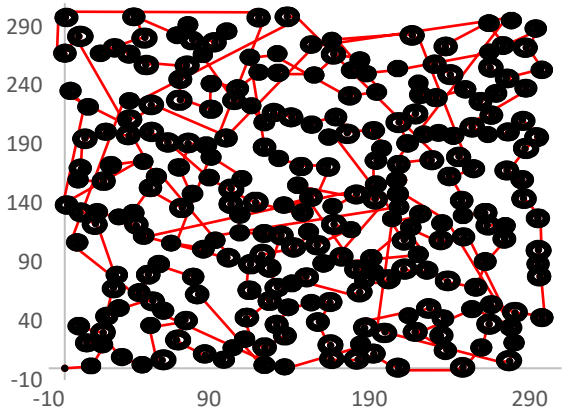


Рис. 3. Пример решения «большой» задачи с использованием оптимизирующей мультивставки на МВС (N=255).

За счет применения оптимизирующей мультивставки удалось добиться улучшение результата, в сравнении с жадной эвристикой на 17%.

1. **Беллман, Р.** Применение динамического программирования к задаче о коммивояжере / Р. Беллман // Кибернетический сборник. – М.: Мир, 1964. – Т.9. – С.219–228.

2. **Хелд, М.** Применение динамического программирования к задачам упорядочения / М. Хелд, Р.М. Карп // Кибернетический сборник. – М.: Мир, 1964. – Т. 9. – С.202–218.
3. **Ченцов, А.Г.** Экстремальные задачи маршрутизации и распределения заданий: вопросы теории. / А.Г. Ченцов. – М.–Ижевск: Регулярная и хаотическая динамика, Ижевский институт компьютерных исследований, 2008.
4. **Chentsov, A.G.** Optimization “In Windows” for Routing Problems with / A.G. Chentsov, A.M. Grigoryev, A.A. Chentsov // Communications in Computer and Information Science. – 2019. – Vol. 1090. – P. 470–485.
5. **Korobkin, V.V.** Routing methods and their applications in improving the safety and efficiency of operation of nuclear power plants / V.V. Korobkin, A.N. Sesekin, O.L. Tashlykov, A.G. Chentsov // Moscow: New technologies, 2019. – 234 p.
6. **Ченцов, А.Г.** Модельный вариант задачи о последовательной утилизации источников излучения (итерации на основе оптимизирующих вставок) / А.Г. Ченцов, А.А. Ченцов // Изв. ИМИ УдГУ. – 2017. – Т. 50. – С. 83–109.
7. **Ченцов, А.Г.** О задаче последовательного обхода мегаполисов с условиями предшествования и функциями стоимости с зависимостью от списка заданий / А.Г. Ченцов, А.А. Ченцов, А.Н. Сесекин // Труды Института математики и механики УрО РАН. – 2020. – Т. 26, № 3. – С.219–234.