№7-2020

ISSN 1999-9429

ИЗВЕСТИЯ ЮФУ

ТЕХНИЧЕСКИЕ НАУКИ

- Принципы построения и архитектура суперкомпьютеров
- Математическое и системное программное обеспечение суперкомпьютеров
- Реконфигурируемые вычислительные системы
- Проблемно-ориентированные и встраиваемые вычислительные системы
- Интеграция параллельных и гибридных распределенных вычислений

ИЗВЕСТИЯ ЮФУ. ТЕХНИЧЕСКИЕ НАУКИ
IZVESTIYA SFedU. ENGINEERING SCIENCES

Свидетельство о регистрации средства массовой информации
ПИ № ФС77-28889 от 12.07.2007
Научно-технический и прикладной журнал
Издается с 1995 года, до середины 2007 года под названием «Известия ТРТУ»
Подписной индекс 41970

№ 7 (217). 2020 г.

Тематический выпуск

СУПЕРКОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ

Журнал включен в «Перечень рецензируемых научных изданий, в которых должны быть опубликованы основные научные результаты диссертаций на соискание ученой степени кандидата наук, на соискание ученой степени доктора наук».

Учредитель Южный федеральный университет.
Издатель Южный федеральный университет.
Ответственный за выпуск Кухаренко А.П.
Технический редактор Ярошевич Н.В.
Оригинал-макет выполнен Ярошевич Н.В.

Подписано к печати . Формат $70\times108\ {}^{1}\!/_{16}$. Бумага офсетная.

Офсетная печать. Усл. печ. л. – 15,8. Уч.-изд. л. – 14,5.
Заказ № . Тираж 250 экз.

Адрес издателя: 344091, г. Ростов-на-Дону, пр. Стачки, 200/1. Тел. 8(863)2478051.

Адрес типографии: Отпечатано в отделе полиграфической, корпоративной и сувенирной продукции Издательско-полиграфического комплекса КИБИ МЕДИА ЦЕНТРА ЮФУ. 344090, г. Ростов-на-Дону, пр. Стачки, 200/1, тел. 8 (863) 247-80-51.

Адрес редколлегии: 347922, г. Таганрог, ул. Чехова, 22, ЮФУ, тел. +7 (928) 909-57-82, e-mail: iborodyanskiy@sfedu.ru, http://izv-tn.tti.sfedu.ru/.

16+ Цена свободная

ISSN 1999-9429 (Print)
ISSN 2311-3103 (Online)

СОДЕРЖАНИЕ

РАЗДЕЛ I. ПРИНЦИПЫ ПОСТРОЕНИЯ И АРХИТЕКТУРА СУПЕРКОМПЬЮТЕРОВ

РАЗДЕЛ II. МАТЕМАТИЧЕСКОЕ И СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ СУПЕРКОМПЬЮТЕРОВ

РАЗДЕЛ III. РЕКОНФИГУРИРУЕМЫЕ ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ

РАЗДЕЛ IV. ПРОБЛЕМНО-ОРИЕНТИРОВАННЫЕ И ВСТРАИВАЕМЫЕ ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ

РАЗДЕЛ V. ИНТЕГРАЦИЯ ПАРАЛЛЕЛЬНЫХ И ГИБРИДНЫХ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛЕНИЙ

CONTENT

SECTION I. PRINCIPLES OF CONSTRUCTION AND ARCHITECTURE OF SUPERCOMPUTERS

SECTION II. MATHEMATICAL AND SYSTEM SOFTWARE FOR SUPERCOMPUTERS

SECTION III. RECONFIGURABLE COMPUTING SYSTEMS

SECTION IV. PROBLEM-ORIENTED AND EMBEDDED COMPUTING SYSTEMS

SECTION V. INTEGRATION OF PARALLEL AND HYBRID DISTRIBUTED COMPUTING

Раздел I. Принципы построения и архитектура суперкомпьютеров

УДК 004.382.2 DOI 10.18522/2311-3103-2020-7-6-19

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

ПЕРСПЕКТИВНЫЕ ВЫСОКОПРОИЗВОДИТЕЛЬНЫЕ РЕКОНФИГУРИРУЕМЫЕ ВЫЧИСЛИТЕЛИ С ИММЕРСИОННЫМ ОХЛАЖДЕНИЕМ

Рассматриваются перспективны создания высокопроизводительных реконфигурируемых вычислительных устройств на основе современных ПЛИС фирмы Xilinx семейства UltraScale+. Целью работы является достижение в одном изделии с конструктивом 3U 19' вычислительной плотности до 128 ПЛИС высокой степени интеграции при обеспечении соответствующих электропитания и охлаждения вычислительных элементов системы для решения вычислительно трудоемких задач. Обеспечение требуемых характеристик изделия в заданном конструктиве потребовало усложнения топологии печатных плат и технологии изготовления его составных частей. Для охлаждения компонентов вычислительной системы используется иммерсионная (погружная) технология. Особенностью разрабатываемых вычислительных систем является широкие возможности информационного обмена внутри блока и между блоками для решения сильносвязанных задач, в которых количество пересылок данных между функциональными устройствами больше, чем количество таких устройств. В качестве основных связей между ПЛИС используются дифференциальные линии с подключенными к ним мультигигабитными трансиверами (MGT). Разработанная на основе оптических каналов система информационного обмена между блоками обеспечивает пропускную способность более 2 Тбит/с. Разработан и изготовлен опытный образец вычислительного модуля на основе ПЛИС UltraScale+. На его основе изготовлен прототип реконфигурируемого вычислительного блока. Вычислительный блок содержит в своем составе универсальный процессор и необходимые интерфейсы ввода-вывода, являясь функционально законченным устройством. На вычислительном модуле нового поколения был реализован ряд алгоритмов различных научно-технических задач, что подтвердило возможность широкого применения вычислителей. Разработана модернизированная иммерсионная подсистема охлаждения, которая обеспечивает отвод выделяемой суммарной тепловой мощности до 20 кВт. Для достижения такого уровня теплоотвода реализованы технические решения по всем компонентам системы охлаждения: хладагенту, радиаторам, насосу, теплообменнику. Объединение множества блоков в единый вычислительный контур позволит создавать вычислительные комплексы с производительностью до нескольких десятков петафлопс. Такие комплексы требуют наличия соответствующей инженерной инфраструктуры.

Реконфигурируемые вычислительные системы; производительность вычислений; иммерсионные системы охлаждения; энергоэффективность вычислений; вычислительная плотность; сильносвязанные задачи.

I.I. Levin, A.M. Fedorov, Yu.I. Doronchenko, M.K. Raskladkin

ADVANCED HIGH-PERFORMANCE RECONFIGURABLE COMPUTERS WITH IMMERSION COOLING

The paper deals with prospects for the design of high-performance reconfigurable computing devices based on modern Xilinx UltraScale+ FPGAs. The aim of the research is the computational density up to 128 LSI FPGAs in one 3U 19' product. Besides, it is necessary to provide the required power

supply and cooling of the system's computing elements during execution of computationally expensive tasks. To provide the product's required parameters in the given design package, we made the topology of our printed circuit boards and the design technology of their parts more complex. To cool the components of our computer system, we use the immersion technology. The distinction of the designed computer systems is wide capabilities of data exchange inside the block and among the blocks. It is crucial for tightly-coupled tasks, where the number of data transfers among functional devices is much higher than the number of such devices. As the main links between FPGAs, we use differential lines with multi-gigabit transceivers (MGT). The optical channels based system of data exchange among the blocks is provides the throughput over 2 Tbit/sec. We designed and produced a prototype of a computational module based on UltraScale+ FPGAs, and a prototype of a reconfigurable computational block. The computational block contains a general-purpose processor and all needed input-output interfaces. It is a stand-alone device. We used the new-generation computational module for implementation of several algorithms of various scientific and technical problems, and proved its wide applicability. We designed a modified immersion cooling subsystem, which dissipates total heat up to 20 kW. To achieve such level of heat dissipation, we implemented technical solutions concerning all components of the cooling system: the cooling agent, heat sinks, the pump, the heat-exchange unit. It is possible to unite several blocks into computer complexes with the performance up to tens of petaflops. Such complexes require a suitable engineering infrastructure.

Reconfigurable computer system; computational performance; immersion cooling system; power efficiency of calculations; computational density; tightly-coupled task.

Введение. В настоящее время в области высокопроизводительных вычислений наблюдается существенное замедление роста производительности традиционных многопроцессорных систем, повышение тактовых частот процессоров остановилось. В то же время для реконфигурируемых вычислительных систем (РВС), построенных на основе программируемых логических интегральных схем (ПЛИС), удается поддерживать рост производительности вычислений в 2,5-3 раза с выходом каждого нового поколения ПЛИС.

Одной из труднейших проблем построения эффективных вычислителей является обеспечение приемлемых тепловых режимов работы как отдельных компонентов с высоким тепловыделением, так и функционально законченного вычислительного устройства в целом. Важным конструкторским решением является выбор той или иной системы охлаждения. В данной области преобладает использование воздушных систем охлаждения, а также систем с жидкостным охлаждением с помощью тепловых трубок. Такие способы охлаждения при высоком тепловыделении становятся очень громоздкими, а использование воды как хладагента существенно снижает надежность изделий.

Проведенные в Научно-исследовательском центре супер-ЭВМ и нейрокомпьютеров (НИЦ СЭ и НК, г. Таганрог) исследования показали [1], что из различных вариантов жидкостного охлаждения наиболее эффективной является иммерсионная технология [2], которая стала активно развиваться и получила свою реализацию в образцах вычислительной техники. Однако изготавливаемые в настоящее время различными производителями конструктивы [3–5] для жидкостного охлаждения обладают очень низкой плотностью компоновки вычислительных элементов и предназначены только для устройств на основе CPU или GPU.

Эффективность погружного охлаждения подтверждена применением этой технологии в серийно выпускаемом в настоящее время изделии «Неккар» [6], построенном на основе ПЛИС Xilinx семейства UltraScale.

В настоящее время в НИЦ СЭ и НК ведется разработка перспективных реконфигурируемых вычислительных блоков (РВБ) [6–21], состоящих из 12 вычислительных модулей (ВМ) с ПЛИС Xilinx семейства UltraScale+ с применением иммерсионной системы охлаждения. Применение ПЛИС фирмы Xilinx семейства UltraScale+, выполненных по 16-нм технологии, позволит до 3-х раз повысить производительность вычислений за счет роста тактовой частоты и степени интеграции ПЛИС, причем без увеличения объема вычислительной системы.

Реконфигурируемые вычислительные блоки «Арктур» и «Сегин». Разрабатываемый РВБ «Арктур» представляет собой функционально законченное изделие с 12-ю ВМ на основе ПЛИС XCVU35P (или XCVU45P). Особенностями данного устройства являются его широкие возможности информационного обмена. РВБ «Арктур» должен обеспечивать решение сильносвязанных задач, т.е. задач, в которых количество пересылок данных между функциональными устройствами больше, чем количество таких устройств. В качестве основных связей между ПЛИС предполагается использование дифференциальных линий с подключенными к ним мульти-гигабитными трансиверами (MGT); вспомогательными связями являются дифференциальные линии, подключенные к HP-банкам. Структурная схема РВБ «Арктур» показана на рис. 1.

В вычислительном модуле ПЛИС связаны «горизонтальными» и «вертикальными» линиями связей. «Горизонтальные» связи между ПЛИС представлены 28-ю дифференциальными парами со скоростью передачи данных до 25 Гбит/с по одной паре, что обеспечивает пропускную способность до 700 Гбит/с независимо в каждом направлении.

Также имеется канал, связывающий по кольцу первую и последнюю ПЛИС в ВМ с помощью 12-ти дифференциальных пар и со скоростью до 300 Гбит/с независимо в каждом направлении.

Дополнительно имеются 48 дифференциальных линий с максимальной скоростью до 1,2 Гбит/с, каждая из которых может работать в любом направлении.

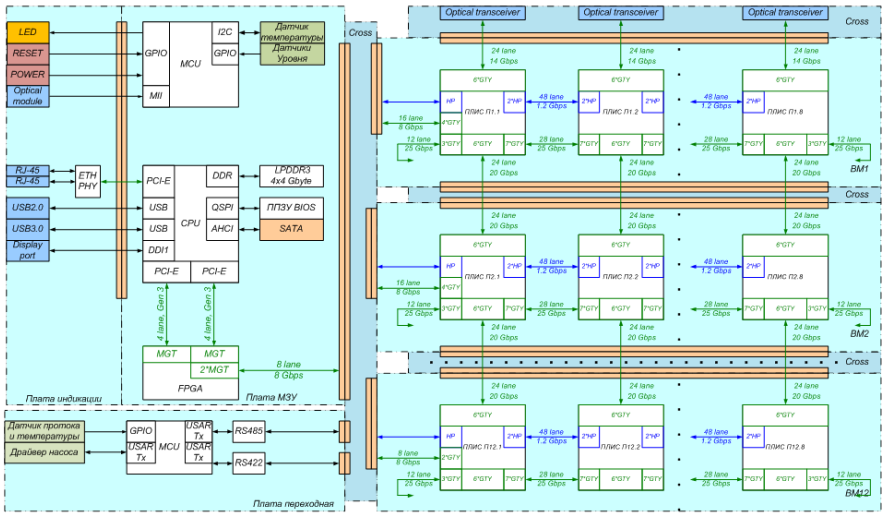


Рис. 1. Структурная схема РВБ «Арктур»

К «вертикальным» связям относятся 192 дифференциальные пары с допустимой скоростью передачи данных до 20 Гбит/с по одной паре, что обеспечивает пропускную способность до 3840 Гбит/с независимо в каждом направлении.

Кроме того, между платами существуют связи, предназначенные для служебных целей, таких как загрузка конфигурации ПЛИС, передача команд и данных от процессора к любому вычислительному модулю. Пропускная способность данного канала – 64 Гбит/с для обоих направлений независимо.

Общая пропускная способность каналов связи ВМ – 13 Тбит/с, в том числе между ВМ – 7,68 Тбит/с.

Возможна организация информационного взаимодействия между РВБ при построении вычислительных комплексов, которое осуществляется через оптические каналы. Многоканальные оптические приемо-передатчики установлены на отдельной кросс-плате и подключаются к первому вычислительному модулю через разъемное соединение. Скорость приема и передачи данных в каждом из оптических каналов достигает 14 Гбит/с. Общее количество оптических каналов составляет 192 канала на прием и 192 канала на передачу, что позволяет получить пропускную способность до 2688 Гбит/с независимо в каждом направлении.

Для осуществления загрузки ПЛИС и управления процессом вычислений разработана оригинальная материнская плата на основе процессора Intel Skylake®Core I5-6300U и ПЛИС семейства Kintex Ultrascale – модуль загрузки и управления (МЗУ). МЗУ обеспечивает функциональную законченность изделия и выполнен в виде отдельной платы. ПЛИС МЗУ выполняет функцию контроллера и через кросс-плату обеспечивает взаимодействие между процессором и ПЛИС в первом вычислительном модуле. На плату индикации выведены кнопки управления, индикаторы, а также интерфейсы для подключения клавиатуры, мыши, дисплея, Ethernet, и оптический интерфейс.

РВБ «Арктур» должен обеспечивать решение задач с особыми требованиями к объему динамической памяти. Типовым техническим решением, применявшимся ранее, было размещение на вычислительном модуле микросхем статической или динамической памяти, связанных с вычислительными ПЛИС. Фирма Xilinx в рамках семейства UltraScale+ выпустила особую линейку «HBM», в которой ПЛИС в рамках одного корпуса обладает новым аппаратным ресурсом – встроенным модулем DDR памяти HBM2. Память HBM2 имеет объем до 16 Гбайт и может обеспечивать многоканальный доступ 16x64 бит. Использование ПЛИС линейки «HBM» позволяет исключить из компоновки платы микросхемы памяти и упростить топологию.

В качестве целевой ПЛИС планируется использование XCVU35P (или XCVU45P), в которой объем HBM2 составляет 8 Гбайт (16 Гбайт).

В связи с тем, что габариты корпуса ПЛИС XCVU35P на 5 мм больше габаритов корпуса ПЛИС семейства UltraScale, используемых в предыдущих РВС, например, РВБ «Скат» [5], РВБ «Неккар», увеличивается длина печатной платы ВМ, что не позволяет разместить, как обычно, восемь вычислительных ПЛИС и контроллер ВМ, также реализованный на ПЛИС. Для сохранения вычислительной плотности изделия из состава вычислительного модуля исключена ПЛИС контроллера. Функции контроллера (обеспечение процесса загрузки конфигураций ПЛИС, реализация интерфейса доступа к вычислительным ресурсам и мониторинг параметров) возложены на вычислительную ПЛИС, что потребует не более 5 % её ресурса.

Реализация в РВБ «Арктур» для решения как можно более широкого класса научно-технических задач мощной системы информационных связей, требующей наличия достаточно габаритных соединителей, неизбежно приводит к ряду ограничений. Приходится жертвовать наличием внешней распределенной памяти, и если в случае с динамической памятью проблему решает применение HBM, то при необходимости реализации максимального объема статической памяти места для неё на плате нет, а объем встроенной в ПЛИС статической памяти пока невелик. ПЛИС XCVU35P не обладает в данном случае максимальной логической емкостью, использование которой возможно в заданном конструктиве. Наиболее эффективной по производительности является ПЛИС XCVU9P, у которой количество System Logic Cells в 1,35 раза больше, чем в XCVU35P.

В этой связи для решения задач, не обладающих сверхсильной связностью, в НИЦ СЭ и НК разрабатывается также РВБ «Сегин» с еще более мощным логическим ресурсом.

Структурная схема РВБ «Сегин» схожа с представленной на рис. 1 схемой РВБ «Арктур». Основным отличием, кроме использования другого типа ПЛИС, является отсутствие межмодульных и межблочных соединений, но при этом на 10% увеличена пропускная способность между ПЛИС. Данное решение позволит достигнуть максимума производительности для заданных габаритов, обеспечив при этом необходимый уровень электропитания и охлаждения компонентов. Структурная схема РВБ «Сегин» показана на рис. 2.

В РВБ нового поколения также будет использоваться электропитание 400 В постоянного тока. Для электропитания вычислительных модулей разработан модуль питания, обеспечивающий четыре канала преобразования DC/DC 400/12 В мощностью до 6,4 кВт для электропитания четырех ВМ. Максимальная потребляемая мощность РВБ – 20 кВт.

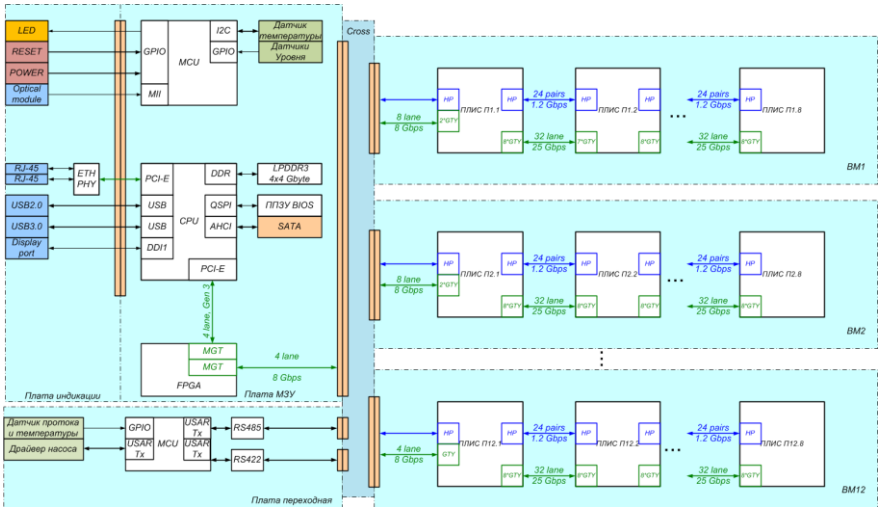


Рис. 2. Структурная схема РВБ «Сегин»

Для охлаждения электронных компонентов РВБ был разработан диэлектрический хладагент – масло маловязкое, диэлектрик МД-4,4 Gazpromneft для охлаждения электронных компонентов ЭВМ. Хладагент обладает высокой электрической прочностью и теплопроводностью, а также максимально возможной теплоемкостью при низкой вязкости.

Исследования на макете и опытном образце вычислительного модуля. Все технические решения по созданию вычислительных модулей «Арктур» и «Сегин» отмакетированы. Макет представляет собой печатную плату с двумя ПЛИС UltraScale+ XCVU9P-1FLGC2104E, необходимой системой питания и элементами системы информационного обмена, в том числе по оптическим каналам. На макете апробированы новая подсистема питания, подсистема мониторинга, технология высокоскоростного обмена (в том числе между РВБ), несколько конфигураций подсистемы синхронизации и другие решения. Фотография макета представлена на рис. 3. В настоящее время произведен и проходит испытания ВМ «Сегин» (рис. 4).

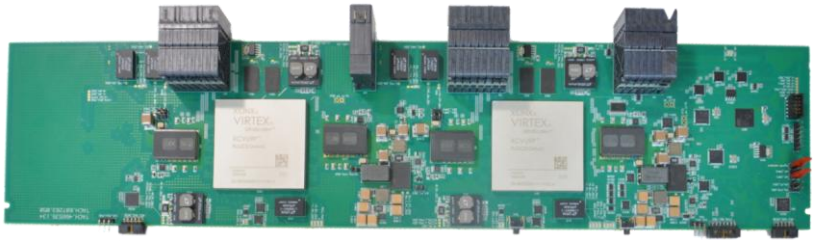


Рис. 3. Макет вычислительного модуля «Сегин»

Как показали исследования ВМ, при выполнении теста с использованием логического ресурса ПЛИС, близкого к максимальному, значение потребляемой мощности в цепи питания ядра ПЛИС XCVU9P-1FLGC2104E достигает 150 Вт при температуре кристалла ПЛИС 50ºС.

Рис. 4. Вычислительный модуль «Сегин»

На макете и на ВМ был реализован ряд алгоритмов различных научно-технических задач, что подтвердило возможность широкого применения вычислителей. На основе результатов выполнена оценка производительности разрабатываемых блоков и системы на их основе – вычислительной стойки из 16-ти блоков (табл. 1).

Таблица 1

Производительность изделий

Вычислительные блоки и вычислительная стойка	Производительность решения задачи КИХ-фильтр (Single precision IEEE-754)	Производительность решения задачи LU-разложения – (Double precision IEEE-754)
РВБ «Сегин»	220 Tflop/s	64 Tflop/s
РВБ «Арктур»	170 Tflop/s	40 Tflop/s
Вычислительная стойка из 16-ти РВБ «Сегин»	3,5 Pflop/s	1000 Tflop/s
Вычислительная стойка из 16-ти РВБ «Арктур»	2,7 Pflop/s	640 Tflop/s

Задача фильтрации позволила оценить достижимую реальную производительность, близкую к пиковой производительности РВБ, так как её алгоритм состоит в основном из вычислительных операций, и его реализация практически полностью задействовала вычислительный ресурс ПЛИС.

Также выполнено решение СЛАУ с помощью конвейерной вычислительной структуры, реализующей функцию LU-разложения, что является алгоритмической основой теста производительности компьютеров LINPACK. Данная оценка, очевидно, не может являться характеристикой для сравнения с другими вычислительными архитектурами, однако подтверждает реализуемость на РВС тестовых задач, используемых для традиционных архитектур, и даёт представление об уровне производительности.

Модернизация системы охлаждения. Увеличение потребляемой мощности разрабатываемых блоков до 20 кВт требует кардинальной переработки подсистемы охлаждения – необходимо повысить её производительность в 2 раза. Эффективность применения открытой 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. Хладагент должен обладать наилучшей электрической прочностью, высокой теплопроводностью, максимально возможной теплоемкостью при низкой вязкости. Радиатор должен обеспечивать максимальную поверхность теплосъема, возможность организации циркуляции хладагента через радиатор, турбулентность потока хладагента в радиаторе, технологичность изготовления. Применяемый термоинтерфейс должен не деградировать и не вымываться хладагентом, иметь стабильно высокий коэффициент теплопроводности. Теплообменник должен обеспечивать высокий коэффициент теплопередачи между основным и вторичным контуром охлаждения. Для обеспечения циркуляции хладагента в объеме РВБ используется насос, который должен обладать необходимой производительностью.

В рамках модернизации системы охлаждения решены следующие сложнейшие задачи:

- разработка новой конструкции радиатора, позволившей увеличить эффективную площадь поверхности теплообмена;
- увеличение производительности насоса, а также повышение надежности посредством применения погружных насосов, что позволит также отказаться от принудительной циркуляции холодного воздуха в вычислительных стойках;
- разработка нового, более производительного теплообменника (в настоящее время ведутся работы по производству в России).

Для отвода большего количества тепла от ПЛИС необходимо увеличивать площадь теплосъема у радиаторов, монтируемых на ПЛИС, исключив при этом увеличение высоты радиатора. В рамках проведенных исследований рассматривалось множество вариантов конструкции радиаторов, часть из которых были эффективны, но очень дорогостоящие в технологии изготовления. Найдено оптимальное решение по исполнению радиаторов, изготовленных по традиционной отработанной технологии, но с увеличенной площадью теплосъема за счет уменьшения толщины ребер (увеличив тем самым количество каналов) и увеличения числа канавок в каналах (рис. 5). Для серийного радиатора с учетом увеличения габаритов ПЛИС удалось увеличить площадь теплосъема более чем в два раза.

Циркуляцию хладагента в системе жидкостного охлаждения обеспечивает насос с приводом от электродвигателя постоянного тока.

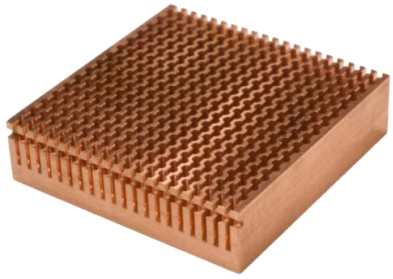


Рис. 5. Радиатор РВБ «Сегин»

Электродвигатель является единственным элементов в РВБ с жидкостным охлаждением, требующим отдельного охлаждения, для чего в вычислительных стойках предусматривается принудительная циркуляция холодного воздуха. Предложено техническое решение по погружению насоса в хладагент РВБ, где происходит непосредственное охлаждение электродвигателя. Данное решение позволяет снизить требования к вычислительным стойкам и помещениям, где эксплуатируются РВС с иммерсионным охлаждением. Кроме того, значительно упрощается техническое обслуживание, а также повышается надежность при эксплуатации за счет уменьшения количества открытых гидравлических соединений.

Увеличение выделяемой тепловой мощности ПЛИС влечет увеличение мощности теплообменника не менее чем в 1,5 раза, что потребовало нарастить количество рабочих пластин с 24 до 36 штук, что неизменно повлияло на его высоту. При существующей схеме расположения гидравлических элементов сохранение габарита изделия 3U стало невозможным. Проблему могло решить расположение теплообменника на боковой поверхности, но в этом случае теплообменник переставал выполнять свою основную функцию – отвод тепла. Дело в том, что у современных теплообменников штуцеры для подключения теплоносителей одной среды, как правило, расположены в одной плоскости, параллельной боковой поверхности, а не по диагонали. Поэтому при установке теплообменника на боковой поверхности невозможно удалить воздух из теплообменника для среды, у которой штуцеры расположены внизу.

Для конструкции РВБ идеальным виделось диагональное расположение штуцеров сред на противоположных сторонах. Это позволило бы разделить подключение хладагента и воды на разных сторонах теплообменника, сократить количество гидравлической арматуры и минимизировать последствия нарушения герметичности подключения изделия. Кроме того, необходимо, чтобы теплообменник эффективно работал с более вязкой средой, чем вода, – хладагентом МД-4,4 Gazpromneft. Это потребовало применения пластин специальной конфигурации.

В сотрудничестве с немецким производителем пластинчатых теплообменников фирмой «Funke» данная проблема была решена. За основу были взяты теплообменники серии TPL, которые разрабатывались специально для охлаждения гидравлических и моторных масел водой. Ширина канала теплообменников TPL до 80% больше, чем у классических паяных водо-водяных теплообменников. Специальными встроенными турбулизаторами и эффективными диагональными направляющими создается максимальная эффективность. Теплообменники TPL обладают повышенными требованиями к технологии пайки, исключающими возможность перемешивания двух сред. В соответствии с требованиями были доработаны рабочие пластины, обеспечивающие диагональное расположение штуцеров рабочих сред на противоположных сторонах изделия (рис. 6). В настоящее время исследуется возможность производства теплообменников в России.

Рис. 6. Теплообменник РВБ «Сегин»

Претерпела изменения и система компенсации объемного расширения хладагента вследствие изменения его температуры. Ранее для этих целей использовался расширительный бачок, сообщающийся с атмосферой, где изменялся уровень хладагента в зависимости от его объемного расширения. Такая система имела ряд недостатков: сложность конструкции расширительного бачка и наличие соединительных гидравлических узлов требовали тщательной герметизации изделия и регулярного контроля уровня хладагента. Кроме того, хладагент, находящийся в расширительном бачке, контактировал с атмосферным воздухом, что приводило к его дополнительному окислению и неизбежному сокращению срока службы.

В предложенном техническом решении роль компенсатора объемного расширения хладагента играют гофрированные резиновые рукава (рис. 7). При этом полностью исключен контакт хладагента с атмосферным воздухом и минимизированы места потенциальных протечек.

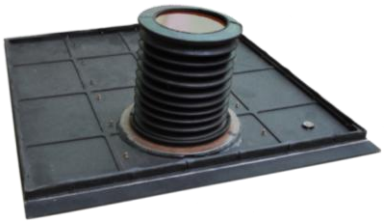


Рис. 7. Компенсатор объемного расширения хладагента

Изготовлены МЗУ, модули питания, собран прототип вычислительного блока (рис. 8).

Прототип РВБ «Сегин» состоит из одного ВМ и 11-ти тепловых эквивалентов, позволяющих эмулировать рассеивание тепловой мощности, эквивалентной 11-ти вычислительным модулям.

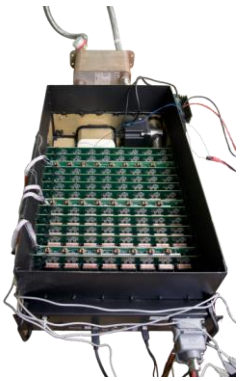


Рис. 8. Прототип РВБ «Сегин»

Заключение. Таким образом, проверена работоспособность технических решений по созданию РВБ нового поколения с модернизированной иммерсионной системой охлаждения, позволяющей обеспечить отвод тепла до 20 кВт.

Разработанные реконфигурируемые устройства позволяют создавать высокопроизводительные вычислительные системы мирового уровня для решения широкого класса трудоемких научно-технических задач.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Гузик В.Ф., Каляев И.А., Левин И.И.* Реконфигурируемые вычислительные системы: учеб. пособие / под общ. ред. И.А. Каляева. 2016. – Ростов-на-Дону: Изд-во ЮФУ, 2016. – 472 с. – ISBN 978-5-9275-1980-7.
2. *Абрамов Сергей, Амелькин Сергей, Клюев Леонид, Чичковский Александр.* Жидкостное охлаждение вычислительных комплексов // Радиоэлектронные технологии. – 2017. – № 5. – С. 79-82.
3. *Клюев Леонид, Хребтовский Иван*. Опыт построения HPC кластеров IMMERS с использованием погружного жидкостного охлаждения. – http://www2.sscc.ru/Seminars/paper/2015/2015-05-14_IMMERS.pdf (дата обращения: 10.10.2019).
4. *Levin I.I., Dordopulo A.I., Doronchenko Y.I., Raskladkin M.K., Fedorov A.M., Kalyaev Z.V.* Immersion liquid cooling FPGA-based reconfigurable computer system // 14th IFAC International Conference on Programmable Devices and Embedded Systems (PDES 2016), Brno/Lednice, Czech Republic. – 2016. – Vol. No. 3. – Part No. 1. – P. 175-180. – https://www.sciencedirect.com/science/article/pii/S2405896316327082.
5. *Levin I.I., Doronchenko Y.I., Kovalenko A.G., Dordopulo A.I.* Realization of Problem of Liquid Filtering in Porous Medium on Reconfigurable Computer System // 5th International Conference on Informatics, Electronics & Vision (ICIEV), 13~14 May, 2016, Dhaka, Bangladesh. – P. 1117-1121. – Doi: 20.1109/ICIEV/2016.7760173. – https://ieeexplore.ieee.org/document/7760173.
6. *Левин И.И., Дордопуло А.И., Федоров А.М., Доронченко Ю.И.* Развитие технологии построения реконфигурируемых вычислительных систем с жидкостным охлаждением // Суперкомпьютерные технологии (СКТ-2018): Матер. 5-й Всероссийской научно-технической конференции: в 2 т. Т. 1. – Ростов-на-Дону; Таганрог: Изд-во ЮФУ, 2018. – С. 184-187. – ISBN 978-5-9275-2834-9.
7. *Levin Ilya, Dordopulo Alexey, Fedorov Alexander, Doronchenko Yuriy.* High-Performance Reconfigurable Computer Systems with Immersion Cooling // Parallel Computational Technologies (PCT 2018). 12th International Conference (Rostov-on-Don, Russia, April 2–6, 2018). – P. 62-76. – https://link.springer.com/content/pdf/10.1007%2F978-3-319-99673-8_5.pdf (дата обращения: 10.10.2019).
8. Патент № 2643173 на изобретение по заявке № 2016152777. РФ. Иммерсионная система охлаждения для электронных устройств / Левин И.И., Федоров А.М., Удовенко Д.Л. Заявитель и патентообладатель Общество с ограниченной ответственностью «НИЦ супер-ЭВМ и нейрокомпьютеров»; зарегистр. 31.01.2018.
9. Патент № 2683425 на изобретение по заявке № 2018104165. РФ. Система теплообмена для жидкостного охлаждения электронных устройств (варианты) / Левин И.И., Федоров А.М., Удовенко Д.Л. Заявитель и патентообладатель Общество с ограниченной ответственностью «НИЦ супер-ЭВМ и нейрокомпьютеров»; зарегистр. 28.03.2019.
10. *Levin Ilya, Dordopulo Alexey, Fedorov Alexander, Doronchenko Yuriy.* Design Technology for Reconfigurable Computer Systems with Immersion Cooling / In: Voevodin V., Sobolev S. (eds) // Supercomputing. RuSCDays 2018. Communications in Computer and Information Science. Vol. 965. – Springer, Cham, 2018. – P. 554-564. – https://link. springer.com/chapter/10.1007/978-3-030-05807-4_47.
11. *Левин И.И., Дордопуло А.И., Доронченко Ю.И., Федоров А.М.* Построение перспективных реконфигурируемых вычислительных систем с жидкостным охлаждением // Шестой Национальный Суперкомпьютерный Форум, (НСКФ-2017), ИПС им. А.К. Айламазяна РАН, г. Переславль-Залесский, Россия, 28 ноября–01 декабря 2017 г. – http://2017.nscf.ru/in-fo/Schedule.pdf.
12. *Левин И.И., Дордопуло А.И., Федоров А.М.* Реконфигурируемые вычислительные системы сверхвысокой производительности с жидкостной системой охлаждения // Матер. Всероссийской научно-практической конференции с международным участием. Технические науки «Аспекты развития науки, образования и модернизации промышленности». Таганрог, 20–21 апреля 2017 г., ДГТУ. – Ростов-на-Дону: ДГТУ, 2017. – С. 67-73. – ISBN 978-5-7890-1329-8.

13. *Левин И.И., Дордопуло А.И., Доронченко Ю.И., Федоров А.М.* Перспективные реконфигурируемые вычислительные системы с жидкостным охлаждением // Матер. 10-й Всероссийской мультиконференции по проблемам управления (МКПУ-2017), 11-16 сентября 2017 г. – Ростов-на-Дону: Изд-во ЮФУ, 2017. – Т. 3. – С. 148-150. – ISBN 978-5-9275-2463-1.
14. *Levin I., Dordopulo A., Fedorov A., Doronchenko Y.* Design of advanced reconfigurable computer systems with liquid cooling // In: Voevodin V., Sobolev S. (eds) Supercomputing. RuSCDays 2017. Communications in Computer and Information Science. Vol. 793. – Springer, Cham, 2017. – P. 442-455. – https://link.springer. com/chapter/10.1007/978-3-319-71255-0_36.
15. *Levin I.I., Dordopulo A.I., Fedorov A.M., Gulenok A.A.* Reconfigurable computer based on Virtex UltraScale+ FPGAs with immersion cooling system // XI международная научная конференция "Параллельные вычислительные технологии (ПаВТ) 2017 (Parallel computational technologies (PCT'2017) agora.guru.ru/pavt)", Казань, Республика Татарстан, Россия, 3-7 апреля 2017 г. – Челябинск: Изд. центр ЮУрГУ, 2017. – С. 27-41. – ISBN 978-5-696-04880-2. – URL: https://link.springer.com/chapter/10.1007/978-3-319-67035-5_3 (дата обращения: 10.10.2019).
16. *Левин И.И., Федоров А.М., Доронченко Ю.И., Раскладкин М.К.* Перспективные высокопроизводительные реконфигурируемые вычислители с иммерсионным охлаждением // Всероссийская научная конференция "Суперкомпьютерные технологии (СКТ) 2020": Тр. молодых ученых. 05-10 октября 2020 г.: Крым, Алушта, Россия. – Ростов-на-Дону; Таганрог: Изд-во ЮФУ, 2020. – С. 29-34.
17. *Левин Илья, Федоров Александр, Доронченко Юрий, Дордопуло Алексей, Раскладкин Максим.* Высокопроизводительный реконфигурируемый вычислительный блок на основе ПЛИС XILINX ULTRASCALE+ // 14th International Conference «Parallel Computational Technologies», PCT 2020, 31 марта-2 апреля 2020 г., Пермь. – P. 174-184. – http://omega.sp.susu.ru/pavt2020/short/079.pdf.
18. *Каляев И.А., Дордопуло А.И., Левин И.И., Федоров А.М.* Развитие отечественных многокристальных реконфигурируемых вычислительных систем: от воздушного к жидкостному охлаждению // Тр. СПИИРАН. – СПб.: Изд-во СПИИРАН ФГБУН Санкт-Петербургский институт информатики и автоматизации РАН, 2017. – № 1 (50). – С. 5-31. – Doi: 10.15622/sp.50.1 http://dx.doi.org/10.15622/sp.50 (дата обращения: 10.10.2019).
19. *Левин И.И., Федоров А.М., Доронченко Ю.И., Раскладкин М.К.* Высокопроизводительный реконфигурируемый вычислитель на основе ПЛИС XILINX ULTRASCALE+ для решения сильносвязанных задач // 12-я Мультиконференция по проблемам управления (МКПУ-2019), 23-28 сентября 2019 г., с. Дивноморское, Россия. – Ростов-на-Дону; Таганрог: Изд-во ЮФУ, 2019. – Т. 3. – С. 113-117. – ISBN 978-5-9275-3188-2.
20. *Levin I.I., Dordopulo A.I., Fedorov A.M., Gulenok A.A.* Reconfigurable computer based on Virtex UltraScale+FPGAs with immersion cooling system / In: Sokolinsky L., Zymbler M. (eds) // Parallel Computational Technologies. PCT 2017. Communications in Computer and Information Science. – Vol 753. – P. 27-41. – Springer, Cham, 2017. – URL: https://link.springer.com/chapter/10.1007/978-3-319-67035-5_3.
21. *Kalyaev I.A., Levin I.I., Dordopulo A.I., Slasten L.M.* FPGA – based Reconfigurable Computer Systems // Proc. of 2013 Science and Information Conference SAI-2013- Oct 9, 2013, London, UK. – P. 148-155.

REFERENCES

1. *Guzik V.F., Kalyaev I.A., Levin I.I.* Rekonfiguriruemye vychislitel'nye sistemy: ucheb. posobie [Reconfigurable computer systems: textbook], study guide end. by I.A. Kalyaeva. 2016. Rostov-on-Don: Izd-vo YuFU, 2016, 472 p. ISBN 978-5-9275-1980-7.
2. *Abramov Sergey, Amel'kin Sergey, Klyuev Leonid, CHichkovskiy Aleksandr*. Zhidkostnoe okhlazhdenie vychislitel'nykh kompleksov [Liquid cooling of computer complexes], *Radioelektronnye tekhnologii* [Radio-electronic technologies], 2017, No. 5, pp. 79-82.
3. *Klyuev Leonid, Khrebtovskiy Ivan.* Opyt postroeniya HPC klasterov IMMERS s ispol'zovaniem pogruzhnogo zhidkostnogo okhlazhdeniya [Design experience of IMMERS HPC clusters with immerse liquid cooling]. Available at: http://www2.sscc.ru/Seminars/paper/2015/2015-05-14_IMMERS.pdf (accessed 10 October 2019).

4. *Levin I.I., Dordopulo A.I., Doronchenko Y.I., Raskladkin M.K., Fedorov A.M., Kalyaev Z.V.* Immersion liquid cooling FPGA-based reconfigurable computer system, *14th IFAC International Conference on Programmable Devices and Embedded Systems (PDES 2016), Brno/Lednice, Czech Republic*, 2016, Vol. No. 3, Part No. 1, pp. 175-180. Available at: https://www.sciencedirect.com/science/article/pii/S2405896316327082.
5. *Levin I.I., Doronchenko Y.I., Kovalenko A.G., Dordopulo A.I.* Realization of Problem of Liquid Filtering in Porous Medium on Reconfigurable Computer System, *5th International Conference on Informatics, Electronics & Vision (ICIEV), 13~14 May, 2016, Dhaka, Bangladesh*, pp. 1117-1121. Doi: 20.1109/ICIEV/2016.7760173. Available at: https://ieeexplore.ieee.org/document/7760173.
6. *Levin I.I., Dordopulo A.I., Fedorov A.M., Doronchenko Yu.I.* Razvitie tekhnologii postroeniya rekonfiguriruemykh vychislitel'nykh sistem s zhidkostnym okhlazhdeniem [Development of the design technology of reconfigurable computer systems with liquid cooling], *Superkomp'yuternye tekhnologii (SKT-2018): Mater. 5-y Vserossiyskoy nauchno-tekhnicheskoy konferentsii* [Supercomputer technologies (SCT-2018): Proceedings of the 5th All-Russian scientific and technical conference]: in 2 vol. Vol. 1. Rostov-on-Don; Taganrog: Izd-vo YuFU, 2018, pp. 184-187. ISBN 978-5-9275-2834-9.
7. *Levin Ilya, Dordopulo Alexey, Fedorov Alexander, Doronchenko Yuriy.* High-Performance Reconfigurable Computer Systems with Immersion Cooling, *Parallel Computational Technologies (PCT 2018). 12th International Conference (Rostov-on-Don, Russia, April 2–6, 2018)*, pp. 62-76. Available at: https://link.springer.com/content/pdf/10.1007%2F978-3-319-99673-8_5.pdf (accessed 10 October 2019).
8. *Levin I.I., Fedorov A.M., Udovenko D.L.* Patent No. 2643173 for an invention No. 2016152777. Russian Federation. Immersionnaya sistema okhlazhdeniya dlya elektronnykh ustroystv [Immerse cooling system for electronic devices]. Applicant of invention and patent holder is OOO "SRC of supercomputers and neurocomputers"; registered on 31.01.2018.
9. *Levin I.I., Fedorov A.M., Udovenko D.L.* Patent No. 2683425 for an invention No. 2018104165. Russian Federation. Sistema teploobmena dlya zhidkostnogo okhlazhdeniya elektronnykh ustroystv (varianty) [Heat-transfer system for liquid cooling of electronic devices (variants)]. Applicant of invention and patent holder is OOO "SRC of supercomputers and neurocomputers"; registered on 28.03.2019.
10. *Levin Ilya, Dordopulo Alexey, Fedorov Alexander, Doronchenko Yuriy.* Design Technology for Reconfigurable Computer Systems with Immersion Cooling, In: Voevodin V., Sobolev S. (eds), *Supercomputing. RuSCDays 2018. Communications in Computer and Information Science*, Vol. 965. Springer, Cham, 2018, pp. 554-564. Available at: https://link. springer.com/chapter/10.1007/978-3-030-05807-4_47.
11. *Levin I.I., Dordopulo A.I., Doronchenko Yu.I., Fedorov A.M.* Postroenie perspektivnykh rekonfiguriruemykh vychislitel'nykh sistem s zhidkostnym okhlazhdeniem [Design of advanced reconfigurable computer systems with liquid cooling], *Shestoy Natsional'nyy Superkomp'yuternyy Forum, (NSKF-2017), IPS im. A.K. Aylamazyana RAN, g. Pereslavl'-Zalesskiy, Rossiya, 28 noyabrya–01 dekabrya 2017 g.* [6th National Supercomputer Forum, (NSCF-2017), Program Systems Institute of the RAS, Pereslavl-Zalesskiy, Russia, November 28 – December 01, 2017]. Available at: http://2017.nscf.ru/in-fo/Schedule.pdf.
12. *Levin I.I., Dordopulo A.I., Fedorov A.M.* Rekonfiguriruemye vychislitel'nye sistemy sverkhvysokoy proizvoditel'nosti s zhidkostnoy sistemoy okhlazhdeniya [Super-high performance reconfigurable computer systems with liquid cooling], *Mater. Vserossiyskoy nauchno-prakticheskoy konferentsii s mezhdunarodnym uchastiem. Tekhnicheskie nauki «Aspekty razvitiya nauki, obrazovaniya i modernizatsii promyshlennosti». Taganrog, 20–21 aprelya 2017 g., DGTU* [Proceedings of All-Russian scientific and practical conference with international participance. Technical sciences "Development aspects of science, education and modernization of industry". Taganrog, April 20–21, 2017., DSTU]. Rostov-on-Don: DGTU, 2017, pp. 67-73. ISBN 978-5-7890-1329-8.
13. *Levin I.I., Dordopulo A.I., Doronchenko Yu.I., Fedorov A.M.* Perspektivnye rekonfiguriruemye vychislitel'nye sistemy s zhidkostnym okhlazhdeniem [Advanced reconfigurable computer systems with liquid cooling], *Mater. 10-y Vse-rossiyskoy mul'tikonferentsii po problemam upravleniya (MKPU-2017), 11-16 sentyabrya 2017 g.* [Proceedings of the 10th All-Russian multi-conference on problems of control (MCPC-2017), September 11-16, 201]. Rostov-on-Don: Izd-vo YuFU, 2017, Vol. 3, pp. 148-150. ISBN 978-5-9275-2463-1.

14. *Levin I., Dordopulo A., Fedorov A., Doronchenko Y.* Design of advanced reconfigurable computer systems with liquid cooling, *In: Voevodin V., Sobolev S. (eds) Supercomputing. RuSCDays 2017. Communications in Computer and Information Science*, Vol. 793. Springer, Cham, 2017, pp. 442-455. Available at: https://link.springer. com/chapter/10.1007/978-3-319-71255-0_36.
15. *Levin I.I., Dordopulo A.I., Fedorov A.M., Gulenok A.A.* Reconfigurable computer based on Virtex UltraScale+ FPGAs with immersion cooling system, *XI mezhdunarodnaya nauchnaya konferentsiya "Parallel'nye vychislitel'nye tekhnologii (PaVT) 2017 (Parallel computational technologies (PCT'2017) agora.guru.ru/pavt)", Kazan', Respublika Tatarstan, Rossiya, 3-7 aprelya 2017 g*. [XI international scientific conference "Parallel computational technologies (PCT'2017) agora.guru.ru/pavt)", Kazan, Republic of Tatarstan, Russia, April 3-7, 2017]. Chelyabinsk: Izd. tsentr YuUrGU, 2017, pp. 27-41. ISBN 978-5-696-04880-2. Available at: https://link.springer.com/chapter/10.1007/978-3-319-67035-5_3 (accessed 10 October 2019).
16. *Levin I.I., Fedorov A.M., Doronchenko Yu.I., Raskladkin M.K.* Perspektivnye vysokoproizvoditel'nye rekonfiguriruemye vychisliteli s immersionnym okhlazhdeniem [Advanced high-performance reconfigurable computers with immersion cooling], *Vserossiyskaya nauchnaya konferentsiya "Superkomp'yuternye tekhnologii (SKT) 2020": Tr. molodykh uchenykh. 05-10 oktyabrya 2020 g.: Krym, Alushta, Rossiya* [All-Russian scientific conference "Supercomputer technologies (SCT) 2020". Proceedings of young scientists. October 05-10, 2020: Crimea, Alushta, Russia]. Rostov-on-Don; Taganrog: Izd-vo YuFU, 2020, pp. 29-34.
17. *Levin Il'ya, Fedorov Aleksandr, Doronchenko Yuriy, Dordopulo Aleksey, Raskladkin Maksim.* Vysokoproizvoditel'nyy rekonfiguriruemyy vychislitel'nyy blok na osnove PLIS XILINX ULTRASCALE+ [High-performance reconfigurable computational block based on XILINX ULTRASCALE+ FPGAs], *14th International Conference «Parallel Computational Technologies», PCT 2020, 31 marta-2 aprelya 2020 g., Perm'* [14th International Conference "Parallel Computational Technologies", PCT 2020, March 31 – April 2, 2020, Perm], pp. 174-184. Available at: http://omega.sp.susu.ru/pavt2020/short/079.pdf.
18. *Kalyaev I.A., Dordopulo A.I., Levin I.I., Fedorov A.M.* Razvitie otechestvennykh mnogokristal'nykh rekonfiguriruemykh vychislitel'nykh sistem: ot vozdushnogo k zhidkostnomu okhlazhdeniyu [Development of domestic multichip reconfigurable computer systems: from air to liquid cooling], *Tr. SPIIRAN* [Proceedings of SPIIRAS]. Saint Petersburg: Izd-vo SPIIRAN FGBUN Sankt-Peterburgskiy institut informatiki i avtomatizatsii RAN, 2017, No. 1 (50), pp. 5-31. Doi: 10.15622/sp.50.1. Available at: http://dx.doi.org/10.15622/sp.50 (accessed 10 October 2019).
19. *Levin I.I., Fedorov A.M., Doronchenko Yu.I., Raskladkin M.K.* Vysokoproizvoditel'nyy rekonfiguriruemyy vychislitel' na osnove PLIS XILINX ULTRASCALE+ dlya resheniya sil'nosvyazannykh zadach [High-performance reconfigurable computer based on XILINX ULTRASCALE+ FPGAs for tightly-coupled tasks], *12-ya Mul'tikonferentsiya po problemam upravleniya (MKPU-2019), 23-28 sentyabrya 2019 g., s. Divnomorskoe, Rossiya* [12th Multiconference on problems of control (MCPC-2019), September 23-28, 2019, Divnomoskoye, Russia]. Rostov-on-Don; Taganrog: Izd-vo YuFU, 2019, Vol. 3, pp. 113-117. ISBN 978-5-9275-3188-2.
20. *Levin I.I., Dordopulo A.I., Fedorov A.M., Gulenok A.A.* Reconfigurable computer based on Virtex UltraScale+FPGAs with immersion cooling system, In: Sokolinsky L., Zymbler M. (eds), *Parallel Computational Technologies. PCT 2017. Communications in Computer and Information Science*, Vol 753, pp. 27-41. Springer, Cham, 2017. Available at:https://link.springer.com/chapter/10.1007/978-3-319-67035-5_3.
21. *Kalyaev I.A., Levin I.I., Dordopulo A.I., Slasten L.M.* FPGA – based Reconfigurable Computer Systems, *Proc. of 2013 Science and Information Conference SAI-2013- Oct 9, 2013, London, UK,* pp. 148-155.

Статью рекомендовал к опубликованию д.т.н. Э.В. Мельник.

Левин Илья Израилевич – Общество с ограниченной ответственностью «НИЦ супер-ЭВМ и нейрокомпьютеров»; e-mail: levin@superevm.ru; 347900, г. Таганрог, пер. Итальянский, 106; тел.: +78634612111; директор; д.т.н.; профессор.

Федоров Александр Михайлович – e-mail: fedorov@superevm.ru**;** начальник отдела.

Доронченко Юрий Иванович – e-mail: doronchenko@superevm.ru; технический директор; к.т.н.

Раскладкин Максим Константинович – e-mail: raskladkin@superevm.ru; начальник отдела; к.т.н.

Levin Ilya Izrailevitch – OOO “SRC of supercomputers and neurocomputers”; e-mail: levin@superevm.ru; 106, Italyanskiy street, Taganrog, 347900, Russia; phone: +78634612111; director; dr. of eng. sc.; professor.

Fedorov Alexander Mikhailovitch – e-mail: fedorov@superevm.ru; head of department.

Doronchenko Yuriy Ivanovitch – e-mail: doronchenko@superevm.ru; technical director; cand. of eng. sc.

Raskladkin Maxim Konstantinovitch – e-mail: raskladkin@superevm.ru; head of department; cand. of eng. sc.

УДК 681.3+004.32 DOI 10.18522/2311-3103-2020-7-19-34

Е.А. Титенко, Е.В. Талдыкин

КОММУТАЦИОННАЯ МОДЕЛЬ ПАРАЛЛЕЛЬНЫХ СРАВНЕНИЙ ЭЛЕМЕНТОВ ДЛЯ ПРОДУКЦИОННЫХ СИСТЕМ, УПРАВЛЯЕМЫХ ПОТОКОМ ДАННЫХ

В статье достигается цель – сокращение временных затрат на генерацию сочетаний элементов множества. Элементы множества формируются из образцов (левых частей) продукционных правил. Основная задача заключается в построении эффективных по времени схем (алгоритмов) параллельной генерации сочетаний элементов массива. Применительно к продукционным системам такие схемы необходимы для активации подмножества продукций, применимых к символьным данным на текущем шаге. За основу взят и развит известный алгоритм параллельного пузырька. Схема коммутации «параллельный пузырек» состоит из двух чередующихся вариантов коммутации элементов в пары. Эти коммутации основаны на локальном объединении в пары элементов массива, имеющих смежные индексы. Такое локальное объединение элементов в пары приводит к «малым» перемещениям элементов по длине массива и регулярному характеру генерации пар. В каждой паре выполняется операция сравнения-обмена операндов. Для продукционных систем операция сравнения сводится к поиску пересечений образцов и формированию списка конфликтных слов. Сокращение времени генерации сочетаний основывается на построении вариантов коммутации с распределенным объединением элементов в пары с шагом, равным 4. Разработанная схема коммутации содержит на нечетных шагах коммутации с локальным объединением элементов в пары. На четных шагах выполняется коммутация-ускоритель с распределенным объединением элементов в пары. Моделирование работы разработанной схемы коммутации осуществлялось на типовых задачах сортировки и полного перебора пар элементов. Установлено сокращение временных затрат по сравнению с четно-нечетной сортиовкой на 15–18 %. В работе определена линейная зависимость времени сортировки с углом наклона меньше 1. Это позволяет использовать схему коммутации для продукционных систем большого размера. Локальные и распределенные связи в схеме коммутации сохраняют свойство регулярности. Эта особенность определяет аппаратную реализацию схемы в виде параллельного коммутатора с естественным масштабированием. Данная схема может использоваться в специализированном продукционном устройстве для декомпозиции продукционной системы на независимые подмножества продукций.

Параллельная сортировка; схема коммутаций; однородное устройство.

E.A. Titenko, E.V Taldykin

SWITCHING MODEL OF PARALLEL COMPARISONS FOR A DATA FLOW RULE BASED SYSTEMS

The article is show the reduction of time spent on generating combinations of elements of the set. The elements of the set are formed from samples (left parts) of the production rules. The main task is to build time-efficient schemes (algorithms) for parallel generation of combinations of array elements. With regard to production systems, such schemes are necessary for the activation of a subset of products applicable to character data in the current step. The basis is taken and developed the well-known algorithm of the parallel bubble. The switching circuit "parallel bubble" consists of two alternating variants of switching elements in pairs. These commutations are based on local union into pairs of array elements with adjacent indices. Such a local combination of elements into pairs leads to "small" displacements of elements along the length of the array and the regular nature of the generation of pairs. In each pair, the operation of comparison-exchange of operands is performed. For production systems, the comparison operation is reduced to the search for sample intersections and the formation of a list of conflicting words. The reduction in the generation time of combinations is based on the construction of switching options with distributed combining of elements in pairs with a step equal to 4. The developed switching scheme contains on odd switching steps with a local combination of elements in pairs. In even-numbered steps, a switching accelerator is performed with a distributed combination of elements in pairs. The simulation of the work of the developed switching scheme was carried out on typical tasks of sorting and complete enumeration of pairs of elements. The reduction of time costs compared with the scheme "parallel bubble" by 15-18%. A linear dependence of the sorting time with a slope angle less than 1 was determined. This allows the use of a switching circuit for large-scale production systems. Local and distributed communications in the switching scheme preserve the property of regularity. This feature determines the hardware implementation of the circuit in the form of a parallel switch with natural scaling. This scheme can be used in a specialized production device for decomposing a production system into independent subsets of products.

Parallel sorting; commutation circuit; multi unit.

Введение. Однородные вычислительные устройства, состоящие из регулярно соединенных ячеек, модулей или блоков, с единым устройством управления представляют перспективный путь создания высокопроизводительных вычислительных устройств (ВУ), совмещающих в операционной части функциональные блоки и коммутационную схему передачи промежуточных результатов [1–3].

Как правило, такие ВУ эффективно применимы для проблемно-поисковых, комбинаторных задач принятия решений, задач обработки символьной информации (ОСИ) с генерацией множества решений [4–6]. В них массово значимыми являются операции множественной генерации и оценки вариантов, тасования, преобразований «бабочка», сортировки, приоритетного выбора, множественного сравнения данных, ассоциативного поиска, поиска и модификации по образцу (шаблону) и др. [7–9]. Такие операции основаны на множественных обменах промежуточных данных, нерегулярных соединениях операндов между собой в информационном графе задачи [10].

Достаточно эффективным подходом обеспечения производительности ВУ на таких задачах является структурное соответствие информационного графа задачи и графа операционной части ВУ с реконфигурацией связей между вершинами графа ВУ [3, 7]. Однако такой путь проектирования ВУ приводит к их специализации, что ограничивает область применения.

Структурное несоответствие графов приводит к функционально и аппаратно избыточным техническим схемам с низкими однородностью и масштабированием. Как следствие, время решения таких задач имеет полиномиальную, степенную сложности, поэтому последовательные алгоритмы и аппаратные решения, разрабатываемые для микропроцессоров и систем общего назначения [11, 12], неэффективны для них.

Обработка символьной информации (ОСИ), связанная с генерацией, анализом и отбором вариантов на основе исчислительных продукционных моделей, является значимым классом проблемно-поисковых задач, использующих схему вычислений «условие→действие» [13]. Главное отличие таких продукционных систем (ПС) или моделей – это естественная ориентация на управление потоком данных [13]. Управление потоком данных основывается на параллельном исполнении такого подмножества правил (продукций), для которых выполнено условие их активации (положительный поиск по образцу) [14–16]. Вторым условием параллельного исполнения потока продукций является наличие достаточного количества функциональных блоков (ячейки, блоки, модули) в операционной части [17, 18].

Распараллеливание потока данных для исчислительных продукционных моделей имеет особенности, связанные с обеспечением полного или направленного перебора и анализа продукций, входящих в ПС, на предмет их систематизации, перестроения, т.е. предобработки ПС. [19, 20]. Сущность управления в такой ПС сводится к заданию ветвящихся вычислительных процессов, отображаемых в виде графа вывода. Недетерминированный характер правил из ПС, управляемых потоком данных, уточняется как возможность их объединения и параллельного выполнения. Это требование объединения набора правил из ПС в текущую активацию обусловливает высокую гибкость коммутационных схем, комбинаторно реализующих выборку правил на выполнение. Основная проблемная ситуация связана с сокращением возвратно-переборных шагов на синтез и выполнение подмножеств активированных продукций.

В связи с этим задача создания эффективных по времени аппаратно-ориентированных коммутационных схем в рамках продукционной парадигмы вычислений является общезначимой задачей теоретической информатики и вычислительной техники.

Постановка задачи. Пусть заданы $n \in$ N, рабочий алфавит A, *метасимволы* $\rightarrow \notin A$, $* \notin A$, слова $O_i, P_i \in A^*$, $i=1\text{-}n$. Пусть задано обрабатываемое слово $S \in A^*$.

Продукция – это правило в алфавите $A \cup \{\rightarrow\}$ вида [9, 13]

$$O \rightarrow P. \tag{1}$$

где O – слово-образец в алфавите A; P – слово-модификатор в алфавите A.

Продукционная система U задается как система-тройка:

$$U = \{ A, \Theta, C \}, \tag{2}$$

где C – схема (стратегия) управления продукциями, Θ – определяющее множество продукций.

Упрощенно, ПС представляется множеством продукций, объединенных на решение задачи стратегией Θ управления потоком данных (потоком активированных продукций).

$$\Theta : \begin{cases} O_1 \rightarrow P_1 \\ \quad \ldots \\ O_i \rightarrow P_i \\ \quad \ldots \\ O_n \rightarrow P_n \end{cases}. \tag{3}$$

Для снижения ресурсных затрат на работу ПС (объем памяти, время перебора продукций) предлагается ее декомпозиция на обособленные подмножества, замкнутые относительно операций алгебры продукций. В этом случае работа ПС разбивается на автономные вычисления по каждому подмножеству с сохранением корректности конечного результата [19, 26].

Декомпозиция ПС основана на алгебре продукций, включающей операции конструктивной логики пересечения, объединения, дополнения слов и др. Эти операции попарно устанавливают свойства структурной связи между образцами и подстановками продукций вида (1). Над ПС выполняются подготовительные вычислительные действия, позволяющие исследовать связи между продукциями и разбить их на обособленные подмножества. Тем не менее характер подготовительных действий имеет комбинаторный характер, а их вычислительная сложность – полиномиальную зависимость, что определяет востребованность специализированных устройств или функциональных узлов, поддерживающих как базовые операции поиска вхождений, пересечений, объединений слов (образцы, подстановки), так и комбинирования образцов и подстановок в пары.

Количество проверок на парные пересечения слов определяется как число сочетаний $C_n^2 = \frac{n(n-1)}{2}$. Квадратичная сложность выражения C_n^2, при больших n или интенсивных модификациях ПС является достаточной критичной для работы ПС.

В связи с этим требуются аппаратные средства обеспечения параллельных продукционных вычислений в виде схем параллельных коммутаций для перебора пар образцов (образцов и подстановок) ПС с линейной временной сложностью. Далее задача перебора рассматривается в рамках задачи сортировки массива [21].

Метод решения. Задача сортировки является классической benchmark-задачей, на которой проверяется эффективность алгоритмов перебора, сравнения и обмена сортируемых элементов, а также работоспособность схемотехнических решений, реализующих последовательные или параллельные вычисления. Структуры данных, применяемые операции и схемы коммутаций элементов, образующие промежуточные данные, составляют основу алгоритмов сортировки.

Пусть задан алфавит $A = \{a_1, a_2, \ldots, a_t\}$, мощностью $|A| = n$ букв. На алфавите A задана функция отношения $F(x_1, x_2)$, $x_1 \in A$, $x_2 \in A$, определяющая порядок пары любых элементов алфавита A:

$$F(x_1, x_2) = \begin{cases} 1, \text{если } (x_1, x_2) \text{ упорядоченная пара} \\ 0, \text{если } (x_2, x_1) \text{ упорядоченная пара} \end{cases} \quad (1)$$

Функция (1) характеризуется свойствами:

- транзитивности – если $F(x_1, x_2) = 1$ и $F(x_2, x_3) = 1$, то $F(x_1, x_3) = 1$;
- рефлективности – $F(x_1, x_1) = 1$;
- антисимметричности – если $F(x_1, x_2) = F(x_2, x_1)$, то $x_1 = x_2$.

В качестве функции отношения будут использованы строгие неравенства вида «<» или «>». Эти неравенства обеспечивают возможность сортировки массива данных по возрастанию или убыванию.

Упорядоченным (отсортированным) массивом $M_{SORT} = \{m_1, m_2, \ldots, m_n\}$, $|M|=n$, $m_i \in A$, $i=1..n$ является массив, в котором для любой пары (m_i, m_j) в случае $i \leq j$ выполняется равенство $F(m_i, m_j)=1$. Алгоритм сортировки преобразует исходный массив $M' = \{m'_1, m'_2, \ldots, m'_n\}$ в отсортированный массив M_{SORT} [21].

Среди всего многообразия алгоритмов сортировки в данном исследовании выделяются алгоритмы сортировки с генерацией множества параллельно выполняемых элементарных процессов сравнения над простейшими массивами - парами элементов. Образование пар элементов основано на динамически изменяемых связях в схемах коммутаций. Это свойство что приводит к образованию наборов пар элементов массива, над сформированными парами элементов массива выполняется элементарное упорядочение согласно (1), а схемы коммутаций обеспечивают перебор всевозможных пар и перемещают элементы в соответствии с отношением упорядочения букв алфавита.

В качестве потенциально возможных алгоритмов, основанных на элементарных процессах сравнения-обмена элементов, рассматриваются алгоритмы [21-24]:

- четно-нечетных перестановок;
- параллельный вариант алгоритма Шелла;
- алгоритм параллельной сортировки Бэтчера.

Также задача сортировки как частный случай перестановки также решается на основе оригинальных настраиваемых коммутационные сетей. Среди наиболее приспособленных под аппаратную сортировку выделяются сеть Каутца, сеть Стоуна [8].

В качестве показателей сравнения алгоритмов принимаются следующие показатели:

- аппаратная реализация;
- однородность загрузки;
- масштабируемость схемы коммутаций;
- степень распараллеливания;
- устойчивость сортировки;
- адаптивность сортировки;
- количество инверсий перестановок;
- требование дополнительной памяти;
- сложность управления.

Под однородностью загрузки понимается количество образуемых пар элементов по итерациям работы алгоритма. С аппаратной точки зрения легче обеспечить управление постоянным количеством коммутируемых пар. Переменное количество коммутируемых пар требует большего количества управляющих сигналов. Масштабируемость схемы основана на однородности схемы коммутаций и заключается в естественном наращивании схемы при увеличении размера исходных данных. Устойчивость – показатель качества сортировки, заключающийся в сохранении отношений следования между одноименными буквами в отсортированном множестве. Адаптивность – показатель, устанавливающий зависимость количества шагов алгоритма от состава исходных данных. В этом смысле используется еще термин чувствительность (чувствительность к данным) алгоритма. Инверсия перестановок – показатель, показывающий количество выполняемых перестановок для расположения элементов в отсортированные позиции. Общеизвестно, что среднее число инверсий перестановок в равно $(n^2 - n)/4$ [22]. Соответственно, чем меньше шагов алгоритм тратит на инверсии, тем он эффективнее. Сложность управления оценивается как количество управляющих элементов (бит), приходящихся на 1 информационный элемент.

В табл. 1 приведены качественные оценки алгоритмов в шкале значений {Низкая, Средняя, Высокая}.

Таблица 1

Качественные оценки алгоритмов сортировки

№	Показатели	Алгоритмы сортировки		
		Четно-нечетная	Параллельная Шелл	Бэтчер
1	Аппаратная реализация	Высокая	Низкая	Средняя
2	Однородность загрузки	Высокая	Средняя	Средняя
3	Масштабируемость схемы	Высокая	Низкая	Средняя
4	Степень распараллеливания	Высокая	Средняя	Высокая
5	Устойчивость	Высокая	Низкая	Низкая

6	Адаптивность	Низкая	Низкая	Высокая
7	Количество инверсий	Низкая	Среднее	Высокая
8	Дополнительная память	Низкая	Средняя	Низкая
9	Сложность управления	Низкая	Высокая	Высокая

В табл. 1 первые 7 качественных показателей являются стимулирующими, последние 2 показателя регрессивные. Анализ трех алгоритмов сортировки выявил, что предпочтительным для аппаратной реализации является алгоритм четно-нечетных сортировки как имеющий большинство лучших значений в предлагаемом 9-мерном пространстве показателей.

Тем не менее, низкое значение показателя «Количество инверсий» определяет необходимость модификации схемы четно-нечетных перестановок для сокращения промежуточных шагов на устранение инверсий в образуемых парах.

Разработка модифицированной схемы коммутации. Классическая схема четно-нечетной сортировки состоит из чередующих коммутаций К1, К2, в которых соседние элементы массива *M* объединяются в пары (рис. 1). Массив элементов понимается как линейная структура.

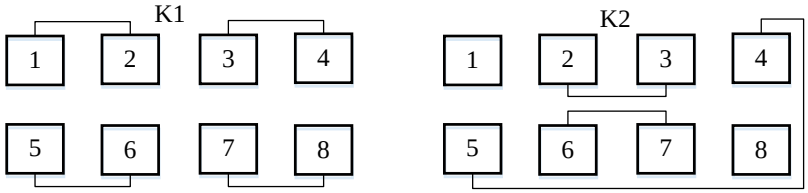


Рис. 1. Коммутации К1, К2 для четно-нечетной сортировки для n=8

Пусть в ПС имеется $n=2w$ правил, где n=$\in$**N**. Пусть каждый элемент массива ассоциируется с правилом вида (1). Тогда правила коммутации К1, К2 обеспечивают перебор пар продукций ПС для проверки истинности конструктивной дизъюнкции пересечения образцов [10] и последующей декомпозиции ПС на обособленные подмножества продукций.

Правила коммутации К1:

$$\forall i \forall j((O_i^H = O_j^K) \vee (O_j^H = O_i^K) \vee (O_i \subset O_j) \vee (O_j \subset O_i))|(i = 1,3,\dots n-1, j = i+1). \quad (3)$$

Правила коммутации К2:

$$\forall i \forall j((O_i^H = O_j^K) \vee (O_j^H = O_i^K) \vee (O_i \subset O_j) \vee (O_j \subset O_i))|(i = 2,4,\dots n-2, j = i+1). \quad (4)$$

Коммутация К1 формирует n/2 пар элементов массива *M*. Коммутация К2 формирует (n/2-1) пар элементов массива *M*.

Образуемые по коммутациям К1, К2 пары элементов (i,j) имеют адреса, отличающиеся между собой на 1, что определяет связность с соседними парами (i-1,i) и (j,j+1). Вместе с тем такая связность уменьшает количество инверсий между элементами не более, чем на 1, что отражается в последовательных перемещениях элементов в необходимые позиции. Как следствие, наибольшее количество шагов для устранения инверсий коммутациями (3) и (4) равно n.

На рис. 2 показана четно-нечетная сортировка по убыванию массива, упорядоченного по возрастанию (один из «худших» наборов).

Для уменьшения количества шагов (сравнений-обменов) четно-нечетной сортировки предлагается модификация с кольцевой трактовкой массива М. В этом случае из граничных элементов массива формируется дополнительная пара элементов, способствующая сокращению количества инверсий (рис. 3).

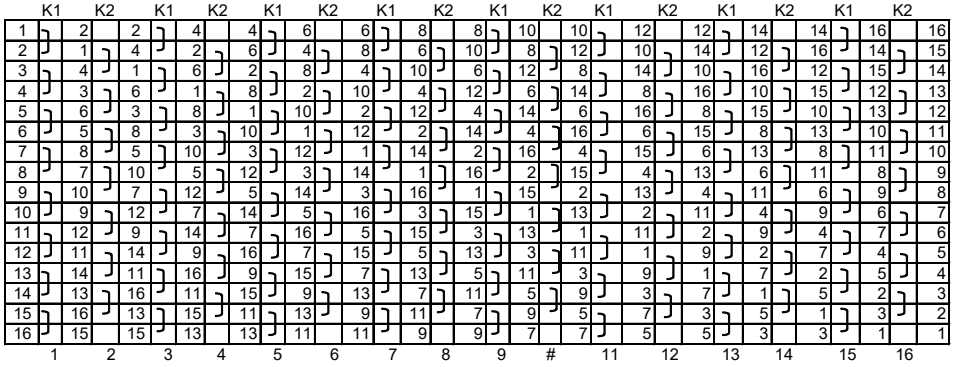


Рис. 2. Пример параллельной четно-нечетной сортировки

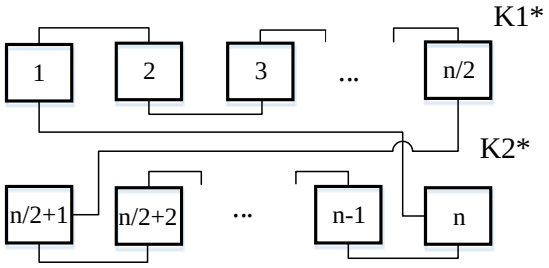


Рис. 3. Модифицированная схема четно-нечетных коммутаций

Теперь коммутации K1* и K2* обеспечивают образование *n*/2 пар элементов каждая.

Правила модифицированной четно-нечетной коммутации имеют вид

$$\text{K1*: } \forall i \forall j\,(i,j) | i = 1,3,\dots n-1, j = (i+1)_{\mathrm{mod}\, n} \tag{5}$$

$$\text{K2*:} \forall i \forall j\,(i,j) | i = 2,4\dots n, j = (i+1)_{\mathrm{mod}\, n}. \tag{6}$$

Сравнение двух схем четно-нечетных коммутаций (рис. 1 и 3) при «матричном» представлении массива *M* выявило свойство замыкания элементов массива относительно операции обмена при добавлении пары (*n*,1). За счет дополнительной пары элементов реализуется обход массива с любой позиции, например, {8, 7, 6, 5, 4, 3, 2, 1}. Кроме того, модифицированная схема четно-нечетных коммутаций формирует параллельные пары элементов, над которыми бесконфликтно реализуются операции «сравнение-обмен»:

K1*: {(1,2), (3,4), (5,6), (7,8)},

K2*: {(2,3), (4,5), (6,7), (8,1)}.

Разработка специализированного устройства модифицированной четно-нечетной сортировки. Разрабатываемое специализированное устройство сортировки (СУС) строится на основе прямого отображения информационного графа задачи в граф функциональных узлов.

Информационный граф задачи описывается модифицированной схемой четно-нечетных перестановок на *n* элементов с правилами коммутации (5), (6). Операционная часть СУС состоит из *n* блоков памяти элементов (БПЭ), соединения которых образуют кольцевую структуру исходного массива *M. (*рис. 4). На рис. 4

соединения соседних БПЭ имеют перекрестный характер, он нужен для реализации параллельных обменов элементов в пределах каждой пары, включая пару ($БПЭ_n$, $БПЭ_1$).

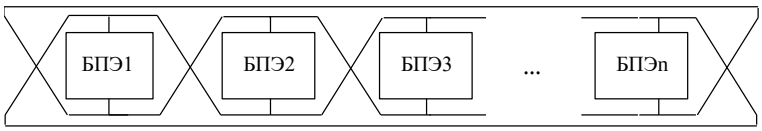


Рис. 4. Общая схема операционной части устройства сортировки

Общая структурная схема СУС (рис. 5) состоит из операционной части, блока управления (БУ) операциями «сравнение-обмен» и БУ коммутациями. БУ «сравнение-обмен» получает от операционной части осведомительные сигналы **X**, выдавая в ответ сигналы **Y**, управляющие микрооперациями записи, чтения, адресации, сравнения и др. БУ коммутациями получает от операционной части осведомительные коммутационные сигналы **KX**, выдавая в ответ управляющие коммутационные сигналы **KY** для проверки и реализации коммутационных соединений БПЭ в пары.

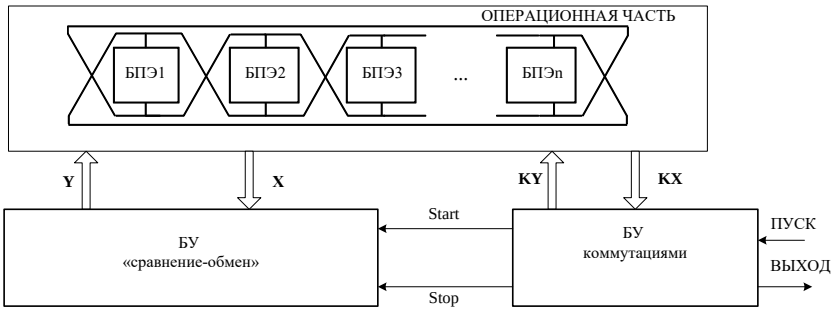


Рис. 5. Общая структурная схема СУС

Наличие двух управляющих автоматов (блоков управления) является отличительной особенностью специализированных устройств данного класса. Блок управления коммутациями является главным, он принимает внешний сигнал ПУСК из внешней среды (от хост-машины) и выдает обратно сигнал окончания работы ВЫХОД. Также БУ коммутациями:

- запускает БУ операциями «сравнение-обмен» (сигнал Start), получая от него, при необходимости, осведомительные сигналы о состоянии процессов коммутации;
- завершает работу БУ «сравнение-обмен» (сигнал Stop).

Детализированная структурная схема специализированного устройства сортировки представлена на рис. 6. Однородный состав СУС и регулярные соединения между блоками памяти элементов позволяют получить общую картину о составе операционной части устройства, его информационных, коммутационных и управляющих входах и выходах.

Каждый $БПЭ_k$ (k=1-n) содержит регистр RG_k для хранения k-го элемента из массива M и буферный регистр $BuffRG_k$ для реализации обмена. Обмен реализуется на основе мультиплексора МХ с однобитовым адресным входом А. Информационные входы мультиплексора (D_0, D_1) подключены к соседним БПЭ. В целом, сортировка четно-нечетными коммутациями обеспечивается подачей чередующейся последовательности на вход А в виде {0,1,0,1 ….} и выполнением последовательности микроопераций до тех пор, пока не будет получен отсортированный массив.

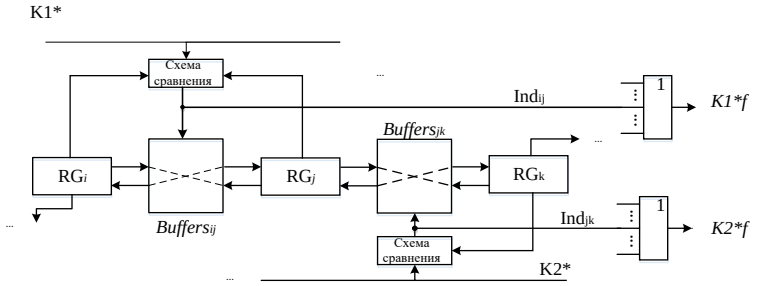


Рис. 6. Структурная схема СУС

Здесь $Buffers_{ij}$, $Buffers_{jk}$ – буферные пары регистров для промежуточного хранения обмениваемых элементов из RG_i, RG_j RG_k соответственно. Выходы Ind_{ij} Ind_{jk} необходимы для проверки условия завершения сортировки. Это условие формируется на основе двух выходных сигналов *K1*f* и *K2*f*, подаваемых в БУ коммутациями. БУ коммутациями прекращает сортировку в соответствии с условием

$$STOP = K1^* f \,\&\, K2^* f = 1. \qquad (7)$$

Выходные сигналы завершения коммутаций *K1**, *K2** формируются на основе многовходовых элементов ИЛИ, которые проверяют нулевые значения от выходов схем сравнения – индикаторы Ind.

На рис. 7 приведена функциональная схема *ij*-й пары БПЭ.

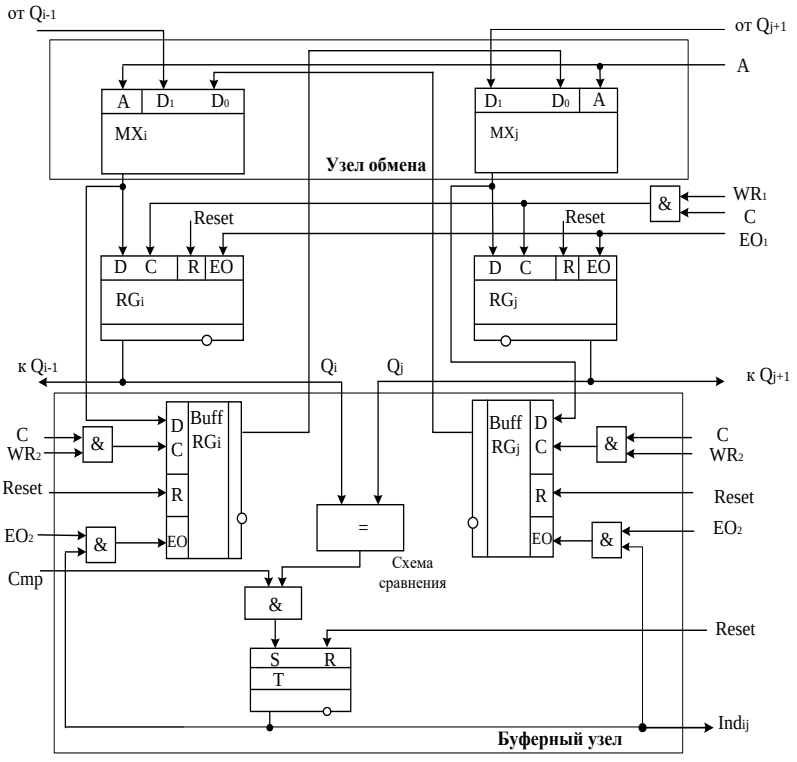


Рис. 7. Функциональная схема пары БПЭ специализированного устройства сортировки

Схема состоит из регистров RG_i и RG_j с высоко импендансными выходами Q_i, Q_j, буферного узла, включая схему сравнения (компаратор), и узла обмена пар элементов. Буферный узел содержит пару буферных регистров $BuffRG_i$ $BuffRG_j$ и асинхронный RS-триггер для фиксации результата сравнения элементов. Схема содержит информационные входы и выходы от соседних БПЭ – Q_{i-1}, Q_{j+1} соответственно.

Адресный вход А, управляемый коммутационными входами К1* и К2*, обеспечивает в зависимости от варианта коммутации (5) или (6) подключение на вход регистров RG_i и RG_j одной из пар значений:

$$(RG_i, RG_j) = \begin{cases} (RG_j, RG_i), \text{если } A = 0 \\ (RG_{i-1}, RG_i) \& (RG_{j+1}, RG_i), \text{если } A = 1 \end{cases}. \quad (8)$$

Функциональная схема (рис. 7) содержит следующие управляющие входы:

- Reset- вход сброса;
- WR_1, WR_2 – входы записи в регистры и буферные регистры;
- EO_1, EO_2 – входы разрешения (подключения) выходов регистров;
- С – вход синхронизации;
- Cmp – вход разрешения обменной записи Q_j и Q_i в буферные регистры.

Аппаратная поддержка микроопераций коммутации, сравнения пар элементов, обменных записей и проверки условия окончания работы позволяет синтезировать алгоритм работы БУ «сравнение-обмен». Алгоритм представляет собой цикл с постусловием, в нем самая длительная операция связана с обменной записью элементов в буферные регистры (функциональные схемы начальной загрузки значений $Q_1 – Q_n$ в БПЭ1 – БПЭn не показаны).

Логическое условие разбиения БПЭ на пары (обозначается как «!») формируется в БУ коммутациями на основе комбинации коммутационных сигналов К1* и К2*:

$$! = K1^* \& \overline{K2}^*. \quad (9)$$

На рис. 8 показана блок-схема алгоритма работы БУ «сравнение-обмен». Фактически по управляющему сигналу вида (9) реализуется одна из коммутаций - К1* или К2*, объединяющая в пары элементы в четно-нечетных позициях массива и при необходимости реализующая перестановки в регистрах БПЭ1 – БПЭn. Алгоритм блока управления «сравнение-обмен» представляет собой цикл с постусловием. Условие выхода из цикла (Stop=1) формируется в БУ коммутациями.

Для корректности подсчета количества шагов сортировки данный алгоритм синхронизируется с алгоритмом управления коммутациями по переменной CountRes. Начальное значение CountRes=0 устанавливается в алгоритме управления коммутациями, а инкрементируется эта переменная в алгоритме «сравнение-обмен». Такое распределение по синхронизируемой переменной CountRes позволяет контролировать состояние работы 2-х блоков управления. При CountRes=2 алгоритм «сравнение-обмен» приостанавливается для проверки условия окончания работы СУС. Проверка выполняется в БУ коммутациями.

Алгоритм управления коммутациями (рис. 9) проверяет значение логической переменной *flag*. Если выполнена хотя бы 1 микрооперация обмена по коммутациям К1* или К2*, то остается *flag*=0. Тогда сбрасывается переменная CountRes=0, и работа алгоритма «сравнение-обмен» продолжается с сохранением Stop=0.

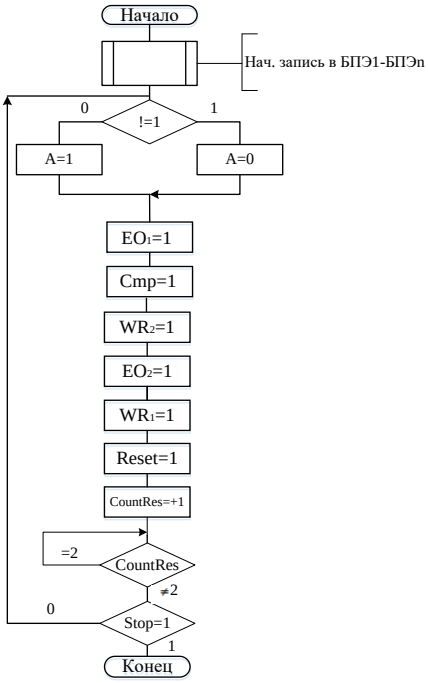


Рис. 9. Блок-схема алгоритма БУ «сравнение-обмен»

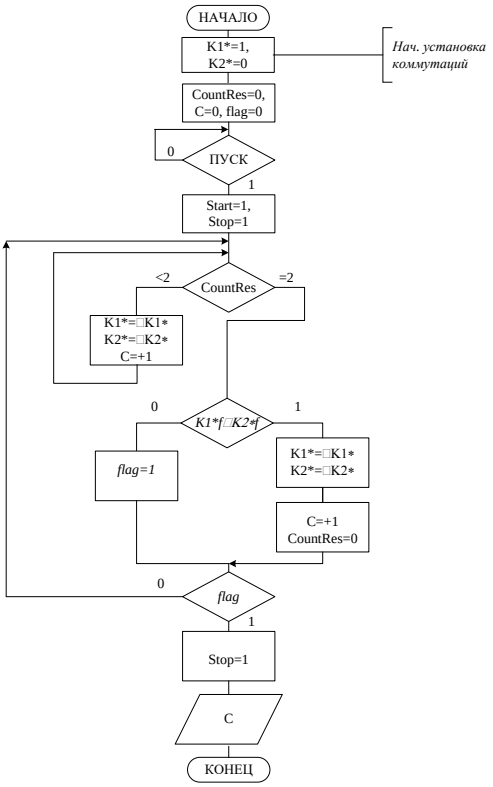


Рис. 9. Алгоритм управления коммутациями

Если на 2-х подряд чередующихся коммутациях K1* и K2* не было ни одной микрооперации обмена ($K1^* \vee K2^*$), тогда устанавливается *flag*=1, далее формируется Stop=1. Значение Stop=1 завершает работу алгоритма «сравнение-обмен». Алгоритм управления коммутациями формирует количество шагов сортировки в переменной *C*.

Итоговая структурная схема СУС представлена на рис. 10.

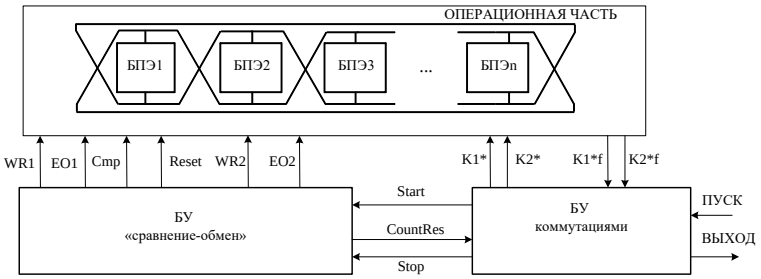


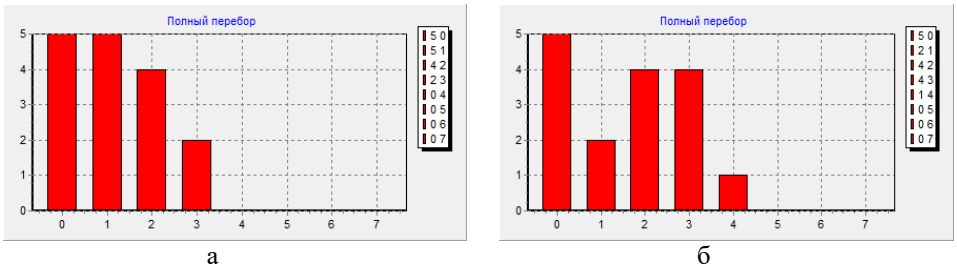
Рис. 10. Итоговая структурная схема СУС

Структурная схема СУС (рис. 10) показывает, что операционная часть устройства состоит из *n* БПЭ, из которых образуется *n/2* пар соседних блоков, что на аппаратном уровне позволяет реализовать одну из коммутаций параллельно. Граф вычислительного процесса представляется итерационным переключением двух коммутаций до тех пор, пока на двух подряд коммутациях выполняется хотя бы одна микрооперация обмена.

Моделирование работы. Непосредственное отображение информационного графа задачи в граф операционной части устройства позволило достичь структурного соответствия и выполнять параллельную обработку пар элементов за 1 шаг работы устройства. При этом операция «сравнение-обмен» является аппаратно-ориентированной, она реализуется на буферных регистрах с перекрестными связями. Управление буферными регистрами осуществляется схем сравнения, значения «1» или «0» на выходах которых разрешают или блокируют обменную перезапись элементов соответственно.

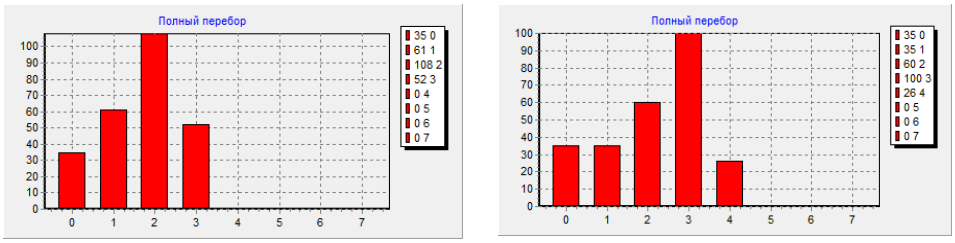
Аппаратная поддержка базовых операций «сравнение-обмен» позволяет достичь линейной зависимости времени сортировки. Это подтверждается моделированием работы алгоритмов четно-нечетной сортировки по схемам (3), (4) и (5), (6) соответственно.

На рис. 11 и 12 приведены гистограммы и средние значения шагов работы алгоритмов четно-нечетной сортировки для массива размером *n*=4 элементов и мощности алфавитов A_1={0,1), A_2={0,1,2,3) на полный перебор всех комбинаций соответственно.



а б

Рис. 11. Гистограммы четно-нечетной сортировки для кольцевой (а) и линейной (б) структуры массива М, A_1={0,1)



а б

Рис. 12. Гистограммы четно-нечетной сортировки для кольцевой (а) и линейной (б) структуры массива M, A_1={0,1,2,3)

Средние значения шагов сортировки по рис. 11 и 12 показаны в табл. 2.

Таблица 2

Среднее количество шагов сортировки

Размер массива *n* (структура данных)	Алфавит	
	{0,1}	{0,1,2,3}
4 (кольцо)	1,20	1,69
4 (строка)	1,63	2,20
8 (кольцо)	3,96	4,59
8 (строка)	4,59	5,23

Соотношения среднего времени для фиксированного размера и заданной мощности алфавита показывает преимущество модифицированной схемы сортировки на 15–18 % перед стандартной четно-нечетной сортировкой.

Выводы:

1. Для обеспечения декомпозиции продукционной системы на обособленные подмножества создана модифицированная четно-нечетная схема коммутаций, состоящая из двух чередующихся вариантов коммутаций. Для оценки эффективности разработанной схемы коммутаций продукционная система отождествлена с массивом, количество элементов которого соотвествует количеству продукций в системе. Эта редукция позволяет заменить операцию пересечения слов на типовую операцию сравнения чисел и исследовать коммутационные процессы.

2. Сравнение классической и модифицированной схем коммутаций показало преимущество кольцевой организации массива, в которой пары из граничных элементов позволяют сократить количество инверсий при сортировке. Схема коммутаций позволяет выполнить перебор всех пар за время с линейной зависимостью от количества продукций в системе. Аппаратная поддержка операций «сравнение-обмен» и динамического образования элементов в пары позволила для модифицированной схемы четно-нечетных коммутаций получить среднее время, меньшее по классической четно-нечетной коммутации на 15-18% в зависимости от размера массива и мощности алфавита

3. В рамках теории структурно-процедурных вычислений разработано специализированное устройство сортировки, обеспечивающее декомпозицию продукционной системы. Оригинальность устройства определяется наличием двух управляющих автоматов, отвечающих за параллельные сравнения-обмены и коммутации элементов и имеющих общую синхронизируемую переменную, что обеспечивает корректность работы устройства.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Бетелин В.Б., Кушниренко А.Г., Райко Г.О.* Проблемы обеспечения роста производительности отечественных суперЭВМ в период до 2020 года // Информационные технологии и вычислительные системы. – 2010. – № 3. – С. 15-18.
2. *Корнеев В.В.* Следующее поколение суперкомпьютеров // Открытые системы. – 2008. – № 8. – С. 14-19.
3. *Гузик В.Ф., Каляев И.А., Левин И.И.* Реконфигурируемые вычислительные системы: учеб. пособие / под общ. ред. И.А. Каляева. – Таганрог: Изд-во ЮФУ, 2016. – 472 с.
4. *Воеводин, В.В., Воеводин Вл.В.* Параллельные вычисления. – СПб.: БХВ-Петербург, 2002. – 608 с.
5. *Огнев И.В., Борисов В.В., Сутула Н.А*. Ассоциативная память, среды, системы. – М.: Горячая линия – Телеком, 2016. – 416 с.
6. *King A.* Distributed Parallel Symbolic Execution. – B.S., Kansas State University, 2005. – P. 181-197.
7. *Каляев А.В., Левин И.И.* Модульно-наращиваемые многопроцессорные системы со структурно-процедурной организацией вычислений. – М.: Изд-во Янус-К, 2003. – 380 с.
8. *Фет Я.И.* Параллельные процессоры для управляющих систем. – М.: Энергоатомиздат, 1981. – 160 с.
9. *Ва Б.У., Лоурай М.Б., Ли Гоцзе*. ЭВМ для обработки символьной информации // ТИИЭР. – 1989. – Т. 77, № 4. – С. 5-40.
10. *Артамонов Д.С., Путря М.Г.* Метод оптимизации вычислительного процесса на реконфигурируемых вычислительных средах // Информационные технологии вычислительные системы. – 2010. – № 3. – С. 19-26.
11. *Путря Ф.М.* Архитектурные особенности процессоров с большим числом вычислительных ядер // Информационные технологии. – 2009. – № 4. – С. 2-7.
12. *Бурцев В.С.* Параллелизм вычислительных процессов и развитие архитектуры суперЭВМ. – М.: ТОРУС ПРЕСС, 2006. – 416 с.
13. *Титенко Е.А., Евсюков В.С.* Продукционные системы и теорема о конфликтных словах // Известия Тульского государственного университет. – 2006. – Вып. 15. – С. 92-98.
14. *Титенко Е.А.* Общая структурная схема реконфигурируемого мультипроцессора // Вестник Воронежского государственного технического университета. – 2011. – Т. 7, № 9. – С. 53-57.
15. *Титенко Е.А.* Метод и однородное вычислительное устройство k-приблизительного поиска вхождений по образцу // Вестник Воронежского государственного технического университета. – 2011. – Т. 7, № 7. – С. 70-78.
16. *Титенко Е.А.* Метод параллельного поиска по образцу и матричное устройство для его реализации // Информационные системы и технологии. – 2011. – № 4 (66). – С. 24-30.
17. *Wichert A.* Artificial intelligence and a universal quantum computer // AI Communications. – 2016. – Vol. 29, Issue 4. – P. 537-543.
18. *Titenko E.A., Degtyarev S.V.* Approximate search in the sample on the basis manber-wu method // Journal of Fundamental and Applied Sciences. – 2017. – Vol. 9, No. 2. – P. 914-918.
19. *Титенко Е.А., Зерин И.С.* Исчислительная система продукций и процедура распознавания конфликтов данных // Вестник компьютерных и информационных технологий. – 2012. – № 2. – С. 24-29.
20. *Люгер Дж.Ф.* Искусственный интеллект: стратегии и методы решения сложных проблем. – М.: Изд. дом «Вильямс». 2003. – 864 с.
21. *Титенко Е.А., Крипачев, Марухленко А.Л.* Коммутационная схема параллельных парных перестановок для специализированного продукционного устройства // Известия ЮФУ. Технические науки. – 2018. – № 8 (203). – С. 29-38.
22. *Макконелл Дж.* Основы современных алгоритмов. – 2-е изд. доп.: пер. с англ. / под ред. С.К. Ландо. – М.: Техносфера, 2004. – 386 с.
23. *Кнут Д.* Сортировка и поиск. Т. 3. – М.: Вильямс, 2000. – 500 с.
24. *Batcher K.E.* Sorting Networks and their Applications // Proc. AFIPS Spring Joint Comput. Conf. – 1968. – Vol. 32. – P. 307314.
25. *Писаренко И.В., Алексеев К.Н., Мельников А.К.* Ресурсонезависимое представление сортирующих сетей на языке программирования Set@l // Вестник компьютерных и информационных технологий. – 2019. – № 9. – С. 3-10.

REFERENCES

1. *Betelin V.B., Kushnirenko A.G., Rayko G.O.* Problemy obespecheniya rosta proizvoditel'nosti otechestvennykh superEVM v period do 2020 goda [Problems of ensuring the productivity growth of domestic supercomputers in the period up to 2020], *Informatsionnye tekhnologii i vychislitel'nye sistemy* [Information technologies and computing systems], 2010, No. 3, pp. 15-18.
2. *Korneev V.V.* Sleduyushchee pokolenie superkomp'yuterov [The next generation of supercomputers], *Otkrytye sistemy* [Open systems], 2008, No. 8, pp. 14-19.
3. *Guzik V.F., Kalyaev I.A., Levin I.I.* Rekonfiguriruemye vychislitel'nye sistemy: ucheb. posobie [Reconfigurable computing systems: textbook], under the general editorship of I.A. Kalyaeva. Taganrog: Izd-vo YuFU, 2016, 472 p.
4. *Voevodin, V.V., Voevodin Vl.V.* Parallel'nye vychisleniya [Parallel computing]. St. Petersburg: BKhV-Peterburg, 2002, 608 p.
5. *Ognev I.V., Borisov V.V., Sutula N.A.* Assotsiativnaya pamyat', sredy, sistemy [Associative memory, environments, and systems]. Moscow: Goryachaya liniya – Telekom, 2016, 416 p.
6. *King A.* Distributed Parallel Symbolic Execution. B.S., Kansas State University, 2005, pp. 181-197.
7. *Kalyaev A.V., Levin I.I.* Modul'no-narashchivaemye mnogoprotsessornye sistemy so strukturno-protsedurnoy organizatsiey vychisleniy [Modular-stackable multiprocessor systems with structural and procedural organization of calculations]. Moscow: Izd-vo Yanus-K, 2003, 380 p.
8. *Fet Ya.I.* Parallel'nye protsessory dlya upravlyayushchikh system [Parallel processors for control systems]. Moscow: Energoatomizdat, 1981, 160 p.
9. *Va B.U., Louray M.B., Li Gotsze.* EVM dlya obrabotki simvol'noy informatsii [Computer for processing symbolic information], *TIIER* [Proceedings of the Institute of Electrical and Radio Electronics Engineers], 1989, Vol. 77, No. 4, pp. 5-40.
10. *Artamonov D.S., Putrya M.G.* Metod optimizatsii vychislitel'nogo protsessa na rekonfiguriruemykh vychislitel'nykh sredakh [Method of optimization of the computational process on reconfigurable computing environments], *Informatsionnye tekhnologii vychislitel'nye sistemy* [Information Technologies computing systems], 2010, No. 3, pp. 19-26.
11. *Putrya F.M.* Arkhitekturnye osobennosti protsessorov s bol'shim chislom vychislitel'nykh yader [Architectural features of processors with a large number of computing cores], *Informatsionnye tekhnologii* [Information technologies], 2009, No. 4, pp. 2-7.
12. *Burtsev V.S.* Parallelizm vychislitel'nykh protsessov i razvitie arkhitektury superEVM [Parallelism of computing processes and the development of the architecture of supercomputers]. Moscow: TORUS PRESS, 2006, 416 p.
13. *Titenko E.A., Evsyukov V.S.* Produktsionnye sistemy i teorema o konfliktnykh slovakh [Productional systems and the theorem on conflict words], *Izvestiya Tul'skogo gosudarstvennogo universitet* [Izvestiya Tulskogo gosudarstvennogo universiteta], 2006, Issue 15, pp. 92-98.
14. *Titenko E.A.* Obshchaya strukturnaya skhema rekonfiguriruemogo mul'tiprotsessora [General block diagram of a reconfigurable multiprocessor], *Vestnik Voronezhskogo gosudarstvennogo tekhnicheskogo universiteta* [Bulletin of the Voronezh State Technical University], 2011, Vol. 7, No. 9, pp. 53-57.
15. *Titenko E.A.* Metod i odnorodnoe vychislitel'noe ustroystvo k-priblizitel'nogo poiska vkhozhdeniy po obraztsu [A method and a homogeneous computing device for k-approximate search for occurrences in a sample], *Vestnik Voronezhskogo gosudarstvennogo tekhnicheskogo universiteta* [Bulletin of the Voronezh State Technical University], 2011, Vol. 7, No. 7, pp. 70-78.
16. *Titenko E.A.* Metod parallel'nogo poiska po obraztsu i matrichnoe ustroystvo dlya ego realizatsii [Method of parallel search by sample and matrix device for its implementation], *Informatsionnye sistemy i tekhnologii* [Information systems and technologies], 2011, No. 4 (66), pp. 24-30.
17. *Wichert A.* Artificial intelligence and a universal quantum computer, *AI Communications*, 2016, Vol. 29, Issue 4, pp. 537-543.
18. *Titenko E.A., Degtyarev S.V.* Approximate search in the sample on the basis manber-wu method, *Journal of Fundamental and Applied Sciences*, 2017, Vol. 9, No. 2, pp. 914-918.
19. *Titenko E.A., Zerin I.S.* Ischislitel'naya sistema produktsiy i protsedura raspoznavaniya konfliktov dannykh [Calculative system of products and the procedure for recognizing data conflicts], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Bulletin of Computer and Information Technologies], 2012, No. 2, pp. 24-29.

20. *Lyuger Dzh.F.* Iskusstvennyy intellekt: strategii i metody resheniya slozhnykh problem [Artificial intelligence: strategies and methods for solving complex problems]. Moscow: Izd. dom «Vil'yams». 2003, 864 p.
21. *Titenko E.A., Kripachev, Marukhlenko A.L.* Kommutatsionnaya skhema parallel'nykh parnykh perestanovok dlya spetsializirovannogo produktsionnogo ustroystva [Switching scheme of parallel pair permutations for a specialized production device], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2018, No. 8 (203), pp. 29-38.
22. *Makkonell Dzh.* Osnovy sovremennykh algoritmov [Fundamentals of modern algorithms]. 2nd ed.: transl. from engl., ed. by S.K. Lando. Moscow: Tekhnosfera, 2004, 386 p.
23. *Knut D.* Sortirovka i poisk [Sorting and searching]. Vol. 3. Moscow: Vil'yams, 2000, 500 p.
24. *Batcher K.E.* Sorting Networks and their Applications, *Proc. AFIPS Spring Joint Comput. Conf.*, 1968, Vol. 32, pp. 307314.
25. *Pisarenko I.V., Alekseev K.N., Mel'nikov A.K.* Resursonezavisimoe predstavlenie sortiruyushchikh setey na yazyke programmirovaniya Set@l [Resource-independent representation of sorting networks in the programming language Set@l], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Bulletin of Computer and Information Technologies], 2019, No. 9, pp. 3-10.

Статью рекомендовал к опубликованию д.ф.-м.н., профессор В.П. Добрица.

Титенко Евгений Анатольевич – Юго-Западный государственный университет; e-mail: johntit@mail.ru; г. Курск, ул. 50 лет Октября, 94; тел.: +79051588904; доцент; в.н.с.

Талдыкин Евгений Владимирович – e-mail: taldykin@mail.ru; аспирант.

Titenko Evgeny Anatolyevich – South-West State University; e-mail: johntit@mail.ru; 94, st. 50 years of October, Kursk, Russia; phone: +79051588904; associate professor, leading researcher.

Taldykin Evgeny Vladimirovich – e-mail: taldykin@mail.ru; post-graduate student.

Раздел II. Математическое и системное программное обеспечение суперкомпьютеров

УДК 004.931 DOI 10.18522/2311-3103-2020-7-35-45

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

МНОГОЭТАПНЫЙ МЕТОД АВТОМАТИЧЕСКОЙ КОРРЕКЦИИ ИСКАЖЕННЫХ ТЕКСТОВ*

Одним из основных факторов, существенно затрудняющих понимание, перевод и анализ текстов, полученных при автоматическом распознавании речи или оптическом распознавании изображений текстов, являются содержащиеся в них искажения в виде ошибочных символов, слов и словосочетаний. Наиболее характерными ошибками систем распознавания являются: – 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 (в т.ч. предлогов и союзов). В результате распознавания получается текст, имеющий искажения и состоящий, в основном, из словарных слов, в том числе и в местах искажений. При большом количестве искажений тексты становятся практически нечитаемыми. Автоматическая обработка таких текстов весьма затруднительна, хотя эта задача является актуальной как для русского, так и для других распространенных языков. Программные средства коррекции, хорошо работающие при малых искажениях в тексте, в случае текстов с высоким уровнем искажений, вне зависимости от их происхождения, показывают неудовлетворительные результаты. Это делает необходимым разработку самостоятельных подходов к коррекции искаженных текстов. Предложен новый многоэтапный метод коррекции искаженных текстов, основанный на последовательном определении ошибок и исправлении искаженных текстов. Искаженными считаются несловарные словоформы и словоформы, вероятность появления которых в тексте в соответствии с выбранной вероятностной моделью меньше заданного порога. После установки признака искаженности для отдельных слов происходит распространение этого признака на их сочетания, т.е. выделяются искаженные фрагменты текста. Для них строится список возможных вариантов слов, в который попадают только те словоформы из словаря, которые находятся от исследуемого слова на определенном расстоянии Левенштейна. Скорректированный текст из вариантов слов получается в результате поиска наиболее вероятной цепочки словоформ. Метод коррекции состоит из нескольких этапов, на каждом этапе корректируются лишь те фрагменты текста, которые остались искаженными после предыдущего этапа коррекции. Метод позволяет заметно повысить качество (точность) коррекции. В проведенных экспериментах качество коррекции в терминах F1-меры для средне искаженных текстов повысилось на 9 %, а для сильно искаженных текстов – на 7.7 %.

Коррекция текста; искаженный текст; исправление ошибок в тексте; модель языка; расстояние Левенштейна; F1-мера.

* Исследование выполнено при частичной финансовой поддержке РФФИ в рамках научного проекта № 18-29-22104.

D.V. Vakhlakov, S.Yu. Melnikov, V.A. Peresypkin

MULTI-PASS METHOD FOR AUTOMATIC CORRECTION OF DISTORTED TEXTS

One of the main factors that significantly complicate the understanding, translation and analysis of texts obtained by automatic speech recognition or optical recognition of text images are the distortions contained in them in the form of erroneous characters, words and phrases. The most typical errors of recognition systems are: – replacement of a word with a similar sounding or graphic spelling; – replacing several words with one; – replacement of one word with several; – skipping words; – insertion or deletion of short words (including prepositions and conjunctions). As a result of recognition, a text is obtained that has distortions and consists mainly of dictionary words, including in places of distortion. With a large amount of distortion, the texts become almost unreadable. Automatic processing of such texts is very difficult, although this task is relevant both for Russian and for other common languages. Correction software that works well at low distortions in the text, in the case of texts with a high level of distortion, regardless of their origin, show unsatisfactory results. This makes it necessary to develop independent approaches to correcting distorted texts. A new multi-pass method for correction of distorted texts based on sequential error identification and correction of distorted texts is proposed. Non-dictionary word forms and word forms which occurrence probability in the text in accordance with the selected probabilistic model is less than a preset threshold are considered to be distorted. After setting of the distortion sign for individual words, this sign is spread to their combinations, i.e. distorted text fragments are extracted. A list of possible word variants which includes only those word forms from the dictionary that are located at a certain Levenshtein distance from the word under study is built for them. The corrected text from word variants is obtained by searching for the most probable chain of word forms. The correction method consists of several passes, at each pass only those fragments of the text are corrected that remained distorted after the previous pass of correction. The method allows to increase significantly the quality (accuracy) of the correction. In the carried out experiments the quality of correction in terms of the F1-measure for moderately distorted texts has been increased by 9 %, and for highly distorted texts – by 7.7 %.

Language model; automatic text correction; distorted text; noisy text; F1-measure; Levenshtein distance.

Введение. Актуальность задачи. Одним из основных факторов, существенно затрудняющих понимание, перевод и анализ текстов, полученных при автоматическом распознавании речи или оптическом распознавании изображений текстов, являются содержащиеся в них искажения в виде ошибочных символов, слов и словосочетаний ([1–3]). При большом количестве искажений тексты становятся практически нечитаемыми. Автоматическая обработка таких текстов весьма затруднительна, хотя эта задача является актуальной как для русского, так и для других распространенных языков ([4]).

В [5] показано, что распространенные программные средства коррекции ([6–9]), хорошо работающие при малых искажениях 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 самостоятельных подходов к коррекции сильно искаженных текстов.

Задача коррекции искаженных текстов, полученных теми или иными системами распознавания, в последние годы формируется как отдельное направление (пост-обработка) и привлекает значительное внимание исследователей. Так, 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 [10, 11]. 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.

Отметим, что близкие к описанным постановки рассматриваются также в биоинформационных задачах секвенирования и сборки больших геномов [12] для коррекции так называемых чтений, получаемых с помощью машин-секвенаторов. Эти чтения могут содержать ошибки, поскольку секвенирование основано на выполнении ряда химических реакций, исправление ошибок в наборе чтений является одним из необходимых этапов сборки генома.

В большинстве исследований (см. обзор в [13]) для коррекции используются статистики сочетаемости слов в текстах, т.н. вероятностные языковые модели, которые позволяют строить цепочки словоформ (слов) скорректированного текста из колонок вариантов слов для каждого искаженного фрагмента текста ([14]). В настоящей работе предложен новый многоэтапный метод автоматической коррекции искаженных текстов, обоснованы его преимущества с позиций точности распознавания и скорости работы на современных вычислительных средствах.

Многоэтапные методы коррекции и близкие подходы. Идеи многоэтапных алгоритмов коррекции ошибок использовались разными авторами, прежде всего при разработке систем распознавания речи. В [15] описано применение сложных лингвистических моделей для распознавания при сохранении баланса между сложностью и эффективностью. Предложенная структура состоит из трех этапов: начальное распознавание, обнаружение ошибок и исправление ошибок. Представлен и оценен прототип трехэтапного метода распознавания диктовки на мандаринском диалекте. В этом прототипе первый проход распознает речь с помощью хорошо обученного распознавателя, включающего эффективную языковую модель; второй проход обнаруживает ошибки распознавания с помощью процедуры обнаружения ошибок; третий проход исправляет ошибки, обнаруженные в полученных слабо искаженных текстах. Алгоритм исправления ошибок исправляет ошибки распознавания, сначала создавая списки кандидатов для ошибок, а затем повторно ранжируя кандидатов с помощью триграммной языковой модели.

Для построения множества слов–кандидатов на замену ошибочного, помимо расстояния Левенштейна, могут применяться также его модификации [16], позволяющие точнее корректировать случаи ошибочного разбиения слова на несколько или ошибочного соединения (склейки) рядом стоящих слов.

В работах [17] и [18] предложены усовершенствования этапа коррекции ошибок. Наряду с используемой вероятностной моделью текста, вводится эвристическая мера уверенности, названная «жизнеспособностью слова», которая позволяет исправлять некоторые синтаксические и семантические ошибки за счет учета информации о соседних словах. Идея метода состоит в переупорядочивании списка слов-кандидатов на замену ошибочного. Алгоритм исправления ошибок исправляет ошибки распознавания, сначала создавая список кандидатов для ошибок, а затем повторно ранжируя кандидатов с помощью комбинации оценки взаимной информации, модельной вероятности триграммы и введенной меры «жизнеспособности слова».

Выбор слов-кандидатов на замену ошибочного слова достаточно трудоемок, что связано с многочисленными вычислениями по языковой модели. В [19] предложен способ, ориентированный на снижение вычислительной трудоемкости процедуры исправления ошибок. При выборе кандидатов на замену используются не только слова, близкие по Левенштейну, но и слова, построенные из решеток распознавания, сгенерированных во время распознавания речи. Используется расширенная языковая модель на триграммах слов, которая учитывает их взаимную информацию, а также факторная модель языка, построенная на частях речи.

В работе [20] для повышения точности систем распознавания речи предложен метод исправления ошибок, использующий расширенные знания о предметной области распознаваемого текста. При ранжировании списка слов-кандидатов используется расстояние фонетического редактирования для выбора фонетически близких кандидатов, а для поиска окончательного результата применяется тематически-ориентированная языковая модель.

В [21] предложен подход к обнаружению ошибок в тексте после распознавания речи, основанный на машинном обучении. Для обучения используется большое число признаков, как фонетических, так и связанных со строением предложений текста.

В [22] используется идея привлечения внешних ресурсов для исправления ошибок. Результат распознавания речи проверяется с помощью программной подсказки правописания Bing для обнаружения и исправления неправильно распознанных слов. Текст разбивается на отрезки, состоящие из нескольких слов, и эти отрезки отправляются в качестве поисковых запросов в программную систему Bing. Возвращенный вариант написания означает, что запрос написан с ошибкой, он заменяется предлагаемым исправлением; в противном случае коррекция не выполняется, и алгоритм переходит к следующему отрезку до тех пор, пока не будут проверен и исправлен весь распознанный текст. Похожий подход предложен в [23]. Орфографические ошибки в словах текста, полученного в результате распознавания речи, корректируются на основе набора данных Microsoft N-Gram.

Многоэтапные алгоритмы обработки также используются в области оптического распознавания в задачах сегментации изображений на текстовые блоки ([24, 25]).

Коррекция искаженных текстов с одновременным поиском ошибок. Наиболее характерными ошибками систем распознавания речи и изображений текстов являются: – 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 (в т.ч. предлогов и союзов). В результате распознавания получается текст, имеющий искажения и состоящий, в основном, из словарных слов, в том числе и в местах искажений. В качестве единицы искаженного текста обычно 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 вычислятся с использованием расстояния Левенштейна, которое равно минимальному количеству изменений (вставок, замен и удалений) символов алфавита, необходимых для преобразования одного слова в другое.

Таким образом, для каждого слова исходного искаженного текста T подбираются возможные варианты близких по расстоянию Левенштейна слов с помощью их выбора из словаря словоформ языка. В получившихся колонках вариантов слов строятся всевозможные последовательности словоформ (цепочки слов) $S = s_1, s_2, ..., s_n$ и надо найти такую цепочку слов $\hat{S}$, для которой достигается максимум условной вероятности $P(S/T)$:

$$\hat{S} = \max_S P(S/T).$$

В соответствии с формулой Байеса справедливо равенство

$$\max_S P(S/T) = \max_S (P(S) \bullet P(T/S)). \quad (1)$$

Вероятность $P(S)$ определяется моделью языка. В частности, в рамках N-граммной словарной модели эта вероятность представима в виде произведения N условных вероятностей: $P(S) = \prod_{i=1}^{n} P(s_i / s_{i-N+1}, s_{i-N+2}, \ldots s_{i-1})$.

Условная вероятность $P(T/S)$ определяется распределением случайных искажений, которым подвергается текст. Если эти искажения возникают в результате процедуры распознавания, получить необходимые для вычислений по формуле (1) значения $P(T/S)$ практически невозможно. В этом случае можно использовать модели случайных искажений текста (в частности, анализировавшиеся в [26] и [5]).

Для вычисления максимума правой части формулы (1) используются те или иные алгоритмы дискретной оптимизации, например, алгоритмы динамического программирования.

Несмотря на адекватность описанного подхода к коррекции ошибок, он имеет недостатки: используемые на практике модели языка и модели искажений (особенно для систем распознавания) являются достаточно грубыми; алгоритмы дискретной оптимизации в условиях таких сложных оптимизируемых функций чрезвычайно трудоемки и могут обеспечивать поиск лишь локальных оптимумов.

Возможным компромиссом являются многоэтапные методы, обеспечивающие последовательные приближения к оптимальному решению.

Описание нового многоэтапного метода автоматической коррекции искаженных текстов. Предлагаемый метод основан на многоэтапном применении описанного выше подхода, причем на каждом этапе корректируются лишь те фрагменты текста, которые остались искаженными после предыдущего этапа коррекции.

Искаженными считаются несловарные словоформы и словоформы, вероятность появления которых в тексте в соответствии с выбранной вероятностной моделью меньше заданного порога. Словоформы определяются как непрерывные последовательности буквенных символов, отделённых друг от друга пробелами или знаками препинания.

После установки признака искаженности для отдельных слов происходит распространение этого признака на их сочетания, т.е. выделяются искаженные фрагменты текста (ИФТ). Опишем подробнее подход к определению ИФТ.

1. Искаженное слово *А* – это ИФТ.

2. Если слова *А* и *Б* являются ИФТ и между ними присутствует пробел или несколько пробелов, то конкатенация «*А Б*» – тоже ИФТ. Пример объединения участков приведен на рис. 1.

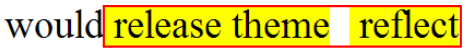


Рис. 1. Пример объединения двух ИФТ, между которыми находится знак пробела

3. Если участки *А* и *Б* являются ИФТ и между ними присутствует слово, в котором менее *d* символов (*d* – параметр), то конкатенация *«А W Б»* – тоже ИФТ (рис. 2).

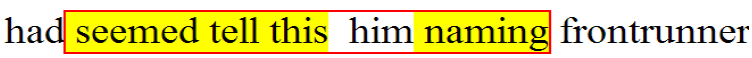


Рис. 2. Пример объединения двух ИФТ, между которыми слово из 3-х символов (для значения d>3)

Опишем предложенный метод коррекции искаженных текстов в виде *k*-этапной процедуры.

На вход поступает искаженный текст, модель языка и словарь словоформ языка.

В качестве параметров метода выступают: k – количество этапов, d – количество символов при определении ИФТ.

1 этап. В исходном тексте с искажениями после определения ИФТ для каждого входящего в них слова строится список возможных вариантов слов, в который попадают только те словоформы из словаря, которые находятся от исследуемого слова на расстоянии Левенштейна, равном 1. Скорректированный текст из вариантов слов получается в результате поиска наиболее вероятной цепочки словоформ в соответствии с (1).

2 этап. В скорректированном после 1 этапа тексте снова определяются ИФТ; для всех искаженных слов, входящих в них, строятся новые списки кандидатов, куда попадают слова, находящиеся на расстоянии Левенштейна, равном 2. Далее в построенных списках ищется наиболее вероятная цепочка словоформ, которая является скорректированным текстом.

…..

k-й этап. В скорректированном на (*k-1*)-м этапе тексте определяются ИФТ, для входящих в них слов строятся списки, куда попадают слова, находящиеся от искаженных слов на расстоянии Левенштейна, равном k. В построенных списках ищется наиболее вероятная цепочка словоформ.

Результатом работы является выход последнего этапа.

Таким образом, алгоритм коррекции состоит из нескольких этапов, на каждом из которых используется определенное значение расстояния Левенштейна, а входными данными являются результаты предыдущего этапа.

Описание и результаты экспериментов. Эксперименты по коррекции проводились с двумя группами текстов, со средними (70 текстов) и сильными искажениями (50 текстов). Процент искаженных слов в текстах первой группы составлял от 15 до 35%, в текстах второй группы от 36 до 50%. Длина текстов варьировалась в пределах от 3000 до 5000 символов. Искажения проводились с помощью программной процедуры, описанной в [5], и являлись комбинацией случайных словарных и символьных искажений.

В качестве языковых моделей использовались 4-граммные модели с модифицированным сглаживанием Кнессера-Нея, построенные на корпусе объемом 200 млн. слов.

Программная реализация описанного подхода тестировалась на вычислителе следующей конфигурации: процессор Intel (R) Xeon (R) CPU E5-2699 v4 @ 2.20GHz, 44 ядра, ОЗУ 250 ГБ, с установленной ОС Windows Server 2012 R2 Standard, x6.

Целью экспериментов являлось получение оценок меры качества и скорости коррекции при различных параметрах алгоритма.

Мерой качества коррекции служила $F1$-мера, которая вычисляется как гармоническое среднее точности A и полноты R коррекций искажённого текста с одинаковым весом, т.е.

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

Полнота коррекции R рассчитывалась как отношение количества верно скорректированных слов $W(T)$ к количеству слов в искажённых фрагментах $W(E)$,

$$R = \frac{W(T)}{W(E)}. \quad (3)$$

а точность коррекции A – через отношение количества слов $W(F)$ в неверных коррекциях к количеству слов в искажённых фрагментах $W(E)$,

$$A = 1 - \frac{W(F)}{W(E)}. \quad (4)$$

В табл. 1 представлены экспериментальные данные по качеству ($F1$-мера (2)) и скорости коррекции (количество скорректированных слов в секунду) программной реализации предложенного подхода.

Таблица 1

Качество и скорость коррекции

	Тексты 1 группы	Тексты 2 группы	Все тексты
Коррекция одноэтапным методом			
Качество ($F1$-мера)	69.3	54.3	63.1
Скорость (слов/сек)	9414.5	895.3	5864.8
Коррекция 4-х этапным методом			
Качество ($F1$-мера)	75.5	58.5	68.4
Скорость (слов/сек)	2007.8	374.4	1327.2

На рис. 3 и 4 представлены графики распределения значений $F1$-меры при коррекции средне и сильно искаженных текстов. Приведены отдельные графики для различного числа этапов (k = 1, 2, 3, 4) многоэтапного метода. Для одноэтапного варианта метода также приведены графики распределения $F1$-меры в четырех случаях, когда список слов-кандидатов составлялся из слов, находящихся на расстоянии Левенштейна L =1, 2, 3, 4 от корректируемого слова.

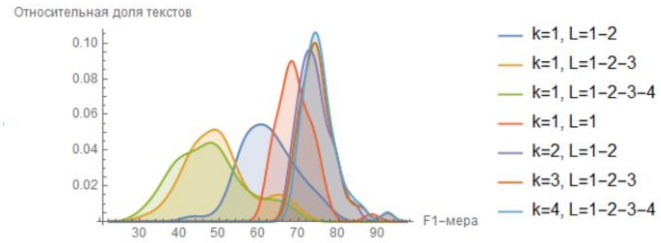


Рис. 3. Распределение значений $F1$-меры после коррекции средне искаженных текстов

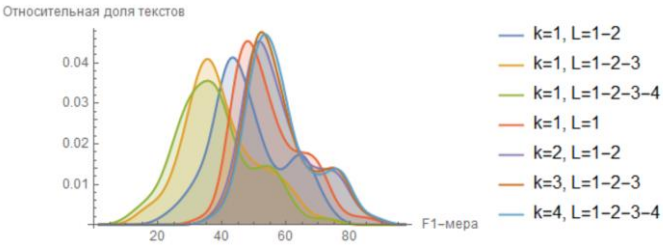


Рис. 4. Распределение значений $F1$-меры после коррекции сильно искаженных текстов

Выводы. Предложен новый многоэтапный метод коррекции искаженных текстов, основанный на последовательном определении ошибок и исправлении искаженных текстов.

Метод позволяет заметно повысить точность коррекции. В проведенных экспериментах качество коррекции в терминах F1-меры для средне искаженных текстов повысилось на 9 %, а для сильно искаженных текстов – на 7.7 %.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Петрова О. О., Булатов К. Б.* Методы пост-обработки результатов распознавания машиночитаемой зоны документов // Тр. ИСА РАН. Специальный выпуск. – 2018. –С. 43-50.
2. *Lee C., Wu S., Liu C., Lee H.* Spoken SQuAD: A Study of Mitigating the Impact of Speech Recognition Errors on Listening Comprehension // Proc. Interspeech. – 2018. – P. 3459-3463.
3. *Мещеряков Р.В.* Структура систем синтеза и распознавания речи // Известия Томского политехн. ун-та. – 2009. – Т. 315, № 5. – С. 127-132.
4. *Шакиров И.Ш., Калаков Б.А.* Автоматизация ручной корректировки ошибок оптического распознавания символов // Инженерные решения. – 2020. – № 3 (13). – С. 7-13.
5. *Бирин Д.А., Мельников С.Ю., Пересыпкин В.А., Писарев И.А., Цопкало Н.Н.* Об эффективности средств коррекции искаженных текстов в зависимости от характера искажений // Известия ЮФУ. Технические науки. – 2018. – № 8 (202). – С. 104-114.
6. Спеллер – Технологии Яндекса. – URL: https://tech.yandex.ru/speller/ (дата обращения: 08.11.2020).
7. AfterScan – post-OCR text proofing, advanced spell-checking, automatic correction. – URL: http://www.afterscan.com/ru/ (accessed: 08.11.2020).
8. *Турдаков Д. и др.* Texterra: инфраструктура для анализа текстов // Тр. Института системного программирования РАН. – 2014. – Т. 26. – Вып. 1. – С. 421-438.
9. Microsoft Cognitive Services – API Bing проверки орфографии. – URL: https://www.microsoft.com/en-us/bing/apis/bing-spell-check-api (accessed: 08.11.2020).
10. *Chiron G., Doucet A., Coustaty M., Moreux J.P.* ICDAR 2017 competition on post-OCR text correction // 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR). – 2017. – Vol. 1. – P. 1423-1428.
11. *Rigaud C., Doucet A., Coustaty M., Moreux J.P.* ICDAR 2019 Competition on Post-OCR Text Correction // International Conference on Document Analysis and Recognition. – 2019. – P. 1588-1593.
12. *Das A.K., Goswami S., Lee K., Park S.J.* A hybrid and scalable error correction algorithm for indel and substitution errors of long reads // BMC Genomics. – 2019. – Vol. 20 (Suppl 11). – P. 1-15.
13. *Германович А.В., Мельников С.Ю., Пересыпкин В.А., Сидоров Е.С., Цопкало Н.Н.* Информационные измерения языка. Программная система оценки читаемости искаженных текстов // Известия ЮФУ. Технические науки. – 2019. – № 8. – С. 6-18.
14. *Мельников С.Ю., Пересыпкин В.А.* О применении вероятностных моделей языка для обнаружения ошибок в искаженных текстах // Вестник компьютерных и информационных технологий. – 2016. – № 5. – С. 29-34.
15. *Zhou Z., Meng H., Lo W.* A multi-pass error detection and correction framework for Mandarin LVCSR // In: Proceedings of the International Conference on Spoken Language Processing (ICSLP). – 2006. – P. 1646–1649.
16. *Nguyen T.-T.-H., Coustaty M., Doucet A., Jatowt A., Nguyen N.-V.* Adaptive Edit-Distance and Regression Approach for Post-OCR Text Correction / In: Dobreva M., Hinze A., Žumer M. (eds) // Maturity and Innovation in Digital Libraries. ICADL 2018: Lecture Notes in Computer Science. – Vol. 11279. – P. 278-289.
17. *Zukerman I., Partovi A.* Improving the understanding of spoken referring expressions through syntactic-semantic and contextual-phonetic error-correction // Computer Speech & Language. – 2017. – Vol. 46. – P. 284-310.
18. *Li B., Chang F., Liu G.* Speech Recognition error correction by using combinational measures // 3rd IEEE International Conference on Network Infrastructure and Digital Content, Beijing, 2012. – P. 375-379.

19. *Zhou Z.* An error detection and correction framework to improve large vocabulary continuous speech recognition. PhD Thesis, HK, 2009.
20. *Ning Y., Xing C., Zhang L.* Domain Knowledge Enhanced Error Correction Service for Intelligent Speech Interaction / In: Wang D., Zhang LJ. (eds) // Artificial Intelligence and Mobile Services – AIMS 2019: Lecture Notes in Computer Science. – Vol. 11516. – P. 179-187.
21. *Zavareh F., Zukerman I., Kim S., Kleinbauer T.* Error Detection in Automatic Speech Recognition // In Proceedings of Australasian Language Technology Association Workshop. – 2013. – P. 101-105.
22. *Bassil Y., Alwani M.* Post-Editing Error Correction Algorithm for Speech Recognition using Bing Spelling Suggestion // International Journal of Advanced Computer Science and Applications. – 2012. – Vol. 3, No. 2. – P. 95-101.
23. *Bassil Y., Semaan P.* ASR Context-Sensitive Error Correction Based on Microsoft N-Gram Dataset // Journal of Computing. – January 2012. – Vol. 4, I.1. – P. 34-42.
24. *Abuhaiba I.* Skew Correction of Textural Documents // Journal of King Saud University – Computer and Information Sciences. – 2003. – Vol. 15. – P. 73-93.
25. *Cao H., Prasad R., Natarajan P., MacRostie E.* Robust page segmentation based on smearing and error correction unifying top-down and bottom-up approaches // In: Ninth Internat. Conf. on Document Analysis and Recognition, Curitiba, Brazil, 2007. – P. 392-396.
26. *Белозеров А.А., Вахлаков Д.В., Мельников С.Ю., Пересыпкин В.А., Скавинская Д.В.* Использование эволюционных методов дискретной оптимизации для коррекции искаженных текстов // Вестник компьютерных и информационных технологий. – 2018. – № 12. – С. 3-10.

REFERNCES

1. *Petrova O. O., Bulatov K. B.* Metody post-obrabotki rezul'tatov raspoznavaniya mashinochitaemoy zony dokumentov [Methods of post-processing of results of recognition of the machine-readable zone of documents], *Tr. ISA RAN. Spetsial'nyy vypusk* [Proceedings of the ISA RAS. Special issue], 2018, pp. 43-50.
2. *Lee C., Wu S., Liu C., Lee H.* Spoken SQuAD: A Study of Mitigating the Impact of Speech Recognition Errors on Listening Comprehension, *Proc. Interspeech.*, 2018, pp. 3459-3463.
3. *Meshcheryakov R.V.* Struktura sistem sinteza i raspoznavaniya rechi [Structure of systems synthesis and speech recognition], *Izvestiya Tomskogo politekhn. un-ta* [Proceedings of the Tomsk Polytechnic University], 2009, Vol. 315, No. 5, pp. 127-132.
4. *Shakirov I.Sh., Kalakov B.A.* Avtomatizatsiya ruchnoy korrektirovki oshibok opticheskogo raspoznavaniya simvolov [Automating manual correction of optical character recognition errors], *Inzhenernye resheniya* [Engineering solutions], 2020, No. 3 (13), pp. 7-13.
5. *Birin D.A., Mel'nikov S.Yu., Peresypkin V.A., Pisarev I.A., Tsopkalo N.N.* Ob effektivnosti sredstv korrektsii iskazhennykh tekstov v zavisimosti ot kharaktera iskazheniy [On the effectiveness of correction tools for distorted texts depending on the nature of the distortion], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2018, No. 8 (202), pp. 104-114.
6. Speller – Tekhnologii Yandeksa [Speller-Yandex Technologies]. Available at: https://tech.yandex.ru/speller/ (accessed 08 November 2020).
7. AfterScan – post-OCR text proofing, advanced spell-checking, automatic correction. Available at: http://www.afterscan.com/ru/ (accessed 08 November 2020).
8. *Turdakov D. i dr.* Texterra: infrastruktura dlya analiza tekstov [Texterra: infrastructure for text analysis], *Tr. Instituta sistemnogo programmirovaniya RAN* [Proceedings of the Institute of System Programming of the Russian Academy of Sciences], 2014, Vol. 26, Issue 1, pp. 421-438.
9. Microsoft Cognitive Services – API Bing проверки орфографии. Available at: https://www.microsoft.com/en-us/bing/apis/bing-spell-check-api (accessed 08 November 2020).
10. *Chiron G., Doucet A., Coustaty M., Moreux J.P.* ICDAR 2017 competition on post-OCR text correction, *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, 2017, Vol. 1, pp. 1423-1428.
11. *Rigaud C., Doucet A., Coustaty M., Moreux J.P.* ICDAR 2019 Competition on Post-OCR Text Correction, *International Conference on Document Analysis and Recognition*, 2019, pp. 1588-1593.
12. *Das A.K., Goswami S., Lee K., Park S.J.* A hybrid and scalable error correction algorithm for indel and substitution errors of long reads, *BMC Genomics*, 2019. Vol. 20 (Suppl 11), pp. 1-15.

13. *Germanovich A.V., Mel'nikov S.Yu., Peresypkin V.A., Sidorov E.S., Tsopkalo N.N.* Informatsionnye izmereniya yazyka. Programmnaya sistema otsenki chitaemosti iskazhennykh tekstov [Information dimensions of the language. Software system for evaluating the readability of distorted texts], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2019, No. 8, pp. 6-18.
14. *Mel'nikov S.Yu., Peresypkin V.A.* O primenenii veroyatnostnykh modeley yazyka dlya obnaruzheniya oshibok v iskazhennykh tekstakh [On the application of probabilistic language models for detecting errors in distorted texts], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Bulletin of Computer and Information Technologies], 2016, No. 5, pp. 29-34.
15. *Zhou Z., Meng H., Lo W.* A multi-pass error detection and correction framework for Mandarin LVCSR, I*n: Proceedings of the International Conference on Spoken Language Processing (ICSLP)*, 2006, pp. 1646-1649.
16. *Nguyen T.-T.-H., Coustaty M., Doucet A., Jatowt A., Nguyen N.-V.* Adaptive Edit-Distance and Regression Approach for Post-OCR Text Correction, In: Dobreva M., Hinze A., Žumer M. (eds), *Maturity and Innovation in Digital Libraries. ICADL 2018: Lecture Notes in Computer Science*, Vol. 11279, pp. 278-289.
17. *Zukerman I., Partovi A.* Improving the understanding of spoken referring expressions through syntactic-semantic and contextual-phonetic error-correction, *Computer Speech & Language*, 2017, Vol. 46, pp. 284-310.
18. *Li B., Chang F., Liu G.* Speech Recognition error correction by using combinational measures, *3rd IEEE International Conference on Network Infrastructure and Digital Content, Beijing, 2012*, pp. 375-379.
19. *Zhou Z.* An error detection and correction framework to improve large vocabulary continuous speech recognition. PhD Thesis, HK, 2009.
20. *Ning Y., Xing C., Zhang L.* Domain Knowledge Enhanced Error Correction Service for Intelligent Speech Interaction, In: Wang D., Zhang LJ. (eds), *Artificial Intelligence and Mobile Services – AIMS 2019: Lecture Notes in Computer Science*, Vol. 11516, pp. 179-187.
21. *Zavareh F., Zukerman I., Kim S., Kleinbauer T.* Error Detection in Automatic Speech Recognition, *In Proceedings of Australasian Language Technology Association Workshop*, 2013, pp. 101-105.
22. *Bassil Y., Alwani M.* Post-Editing Error Correction Algorithm for Speech Recognition using Bing Spelling Suggestion, *International Journal of Advanced Computer Science and Applications*, 2012, Vol. 3, No. 2, pp. 95-101.
23. *Bassil Y., Semaan P.* ASR Context-Sensitive Error Correction Based on Microsoft N-Gram Dataset, *Journal of Computing*, January 2012, Vol. 4, I.1, pp. 34-42.
24. *Abuhaiba I.* Skew Correction of Textural Documents, *Journal of King Saud University – Computer and Information Sciences*, 2003, Vol. 15, pp. 73-93.
25. *Cao H., Prasad R., Natarajan P., MacRostie E.* Robust page segmentation based on smearing and error correction unifying top-down and bottom-up approaches, *In: Ninth Internat. Conf. on Document Analysis and Recognition, Curitiba, Brazil, 2007*, pp. 392-396.
26. *Belozerov A.A., Vakhlakov D.V., Mel'nikov S.Yu., Peresypkin V.A., Skavinskaya D.V.* Ispol'zovanie evolyutsionnykh metodov diskretnoy optimizatsii dlya korrektsii iskazhennykh tekstov [The use of evolutionary methods of discrete optimization for correction of distorted texts], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Bulletin of Computer and Information Technologies], 2018, No. 12, pp. 3-10.

Статью рекомендовал к опубликованию д.т.н., профессор Р.В. Мещеряков.

Вахлаков Дмитрий Владимирович – ФГУП «НТЦ «Орион»; e-mail: melnikov@linfotech.ru; 127018, г. Москва, ул. Образцова, д. 38, стр. 1; тел/факс: +74952499053.

Пересыпкин Владимир Анатольевич – к.т.н.

Мельников Сергей Юрьевич – ООО «Линфо»; e-mail: melnikov@linfotech.ru; 127018, г. Москва, ул. Образцова, д. 38, стр. 1; тел/факс: +74952499053, моб.: +79037222824; к.ф.-м.н.; инженер-математик.

Vakhlakov Dmitriy Vladimirivich – FGUP "NTC "Orion"; e-mail: melnikov@linfotech.ru; 38, Obraztsova street, build. 1, Moscow, 127018, Russia; phone/fax: +74952499053.

Peresypkin Vladimir Anatolyevich – cand. of eng. sc.

Melnikov Sergey Yurievich – Linfo LLC; e-mail: melnikov@linfotech.ru; 38, Obraztsova street, build. 1, Moscow, 127018, Russia; phone/fax: +74952499053, mobile: +79037222824; cand. of eng. sc.; mathematical engineer.

УДК 004.056.55 DOI 10.18522/2311-3103-2020-7-45-52

Е.И. Духнич, А.Г. Чефранов

АППАРАТУРНО-ОРИЕНТИРОВАННЫЙ АЛГОРИТМ ДЛЯ БЫСТРОГО УМНОЖЕНИЯ КРОНЕКЕРОВА ПРОИЗВЕДЕНИЯ МАТРИЦ НА ВЕКТОР

В статье на основе использования свойств произведения Кронекера (КП) матриц предлагается новый алгоритм для повышения эффективности выполнения операции умножения КП на вектор. Указанная операция широко применяется при решении задач обработки сигналов, изображений, криптографии и т.п., где выполняется формирование матриц большого размера с заданными свойствами с помощью КП матриц малого размера. При этом используются матрицы со следующими свойствами: ортогональные (унитарные), обращаемые, инволютивные. Умножение квадратной матрицы размера $n \times n$ на вектор имеет вычислительную сложность $O(n^2)$. Поэтому при росте количества элементарных матриц-сомножителей размер результирующей матрицы КП и сложность умножения ее на вектор растут экспоненциально. Это обстоятельство существенно повышает время решения прикладных задач. Целью предлагаемой работы является построение алгоритма, ориентированного на аппаратную реализацию и ускоряющего процессы формирования КП и умножения вектора на него. Предлагается совместить во времени эти процедуры. Таким образом матрица КП в явном виде фактически не рассчитывается. Вместо этого матрицы-сомножители КП итеративно умножаются на компоненты вектора за время $O(n\log_2 n)$ и требуют линейной сложности памяти. Приведена схема вычислений с топологией гиперкуба для возможной аппаратной реализации предлагаемого алгоритма, которая легко поддается конвейеризации. В разделе 1 приведены определения и свойства КП, используемые при синтезе предлагаемого алгоритма. В разделе 2 рассмотрен иллюстрирующий предлагаемый алгоритм пример с $n = 8$, на основе которого в разделе 3 предложена аппаратурно-ориентированная структура его реализации для произвольного n.

Алгоритм, произведение Кронекера; элементарная матрица; сложность вычислений; конвейерная реализация.

E.I. Dukhnich, A.G. Chefranov

HARDWARE-ORIENTED ALGORITHM FOR FAST MULTIPLICATION OF A VECTOR BY A MATRIX KRONECKER PRODUCT

The article discusses new algorithm to increase the efficiency of the operation of multiplying a matrix Kronecker product (KP) by a vector. It is based on the use of the KP properties. This operation is widely used in solving problems of processing signals, images, cryptography, etc., where the formation of large matrices with specified properties is performed using small size matrices. In this case, matrices with the following properties are used: orthogonal (unitary), invertible, involutive. Multiplying an $n \times n$ square matrix by a vector has a computational complexity of $O(n^2)$. Therefore, with an increase in the number of elementary matrix factors, the size of the resulting KP matrix and the complexity of multiplying it by a vector grow exponentially. This circumstance significantly increases the time for solving applied problems. The aim of the proposed work is to construct an algorithm that accelerates the processes of forming the KP and multiplying

the vector by it. It is proposed to combine the process of multiplication with the process of forming the KP. Thus, the KP matrix is not actually calculated explicitly. Instead, the KP factor matrices are iteratively multiplied by the vector components in $O(n\log_2 n)$ time with linear memory complexity. The computational scheme with the hypercube topology for the possible hardware implementation of the proposed algorithm is presented. It can be easily pipelined. Section 1 presents the definitions and properties of the KP used in the synthesis of the proposed algorithm. Section 2 presents an example with n = 8 illustrating the proposed algorithm, on the basis of which, in Section 3, a hardware-oriented structure of its implementation for arbitrary n is proposed.

Algorithm; Kronecker product; elementary matrix; computational complexity; pipeline implementation.

Введение. Кронекерово произведение (КП) матриц сходно с тензорным произведением и широко используется во многих приложениях, в том числе, при обработке сигналов и изображений [1–9]. Формирование матриц большого размера с заданными свойствами с помощью КП матриц малого размера может использоваться при разработке новых вычислительных алгоритмов. Для обработки сигналов и блочных криптографических алгоритмов используются матрицы со следующими свойствами: ортогональные (унитарные) [10], обращаемые [11], инволютивные [12, 13]. Умножение матрицы на вектор широко используется во многих приложениях, например, вращение объектов в мультимедиа-системах [14]. Очень важным направлением использования матричного произведения является шифрование информации [11, 13] и, в частности, медицинских DICOM изображений для телемедицины [15, 16].

Умножение квадратной матрицы размера $n \times n$ на вектор имеет вычислительную сложность $O(n^2)$ [17]. Размер КП элементарных матриц (ЭМ) может быть значительным, например, КП десяти ЭМ с $n = 10$ равен n^{10}, соответственно, умножение на вектор имеет сложность порядка 10^{20}. Известны эффективные алгоритмы выполнения умножения вектора на КП произвольных матриц [4-6], которые оптимизируют и организацию использования памяти системы. В [6] предложен эффективный алгоритм умножения КП произвольных матриц на вектор (далее, УКПВ), рассмотрена задача о выборе размера матриц, участвующих в КП, минимизирующего вычислительную сложность УКПВ, и показано, что минимальная сложность $O(n\log_2 n)$ достигается при использовании в КП элементарных квадратных матриц размера $n = 2$ [11]. На его основе в [11] предложен алгоритм быстрого УКПВ (АБУКПВ) вычисления УКПВ умножением очередной элементарной матрицы на части входного или промежуточного вектора. АБУКПВ допускает его параллельную реализацию за время $O(log_2 n)$.

В статье приведены определения и свойства КП, используемые АБУКПВ. Показан иллюстрирующий АБУКПВ пример с $n = 8$, на основе которого предложена аппаратурно-ориентированная структура его реализации для произвольного n, являющаяся гиперкубом [19, 20].

1. Кронекерово произведение матриц и его свойства. КП матриц $\boldsymbol{A}(\boldsymbol{m},\boldsymbol{n})$ и $\boldsymbol{B}(\boldsymbol{p},\boldsymbol{r})$ представляет собой блочную матрицу $(\boldsymbol{m}\cdot\boldsymbol{p},\boldsymbol{n}\cdot\boldsymbol{r})$ размера такую, что

$$(A \otimes B)_{ij} = A_{\lfloor (i-1)/p \rfloor+1,\,(\lfloor j-1)/r \rfloor+1} B_{(i-1)\bmod p+1,\,(j-1)\bmod r+1}, \quad (1)$$

$$i = \overline{1,mp}, j = \overline{1,nr},$$

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

Из (1), если $m=n=p=r=2$, следует:

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B \\ a_{21}B & a_{22}B \end{pmatrix} = \\ = \begin{pmatrix} a_{11}b_{11} & a_{11}b_{12} & a_{12}b_{11} & a_{12}b_{12} \\ a_{11}b_{21} & a_{11}b_{22} & a_{12}b_{21} & a_{12}b_{22} \\ a_{21}b_{11} & a_{21}b_{12} & a_{22}b_{11} & a_{22}b_{12} \\ a_{21}b_{21} & a_{21}b_{22} & a_{22}b_{21} & a_{22}b_{22} \end{pmatrix}$$

КП порядка K это произведение K элементарных матриц, $\boldsymbol{P}_j$. $j = \overrightarrow{1,K}$:

$$\boldsymbol{P} = \boldsymbol{P_1}\otimes(\boldsymbol{P_2}\otimes\ldots(\boldsymbol{P_{K-1}}\otimes\boldsymbol{P_K})\ldots) = \bigotimes_{j=1}^{K} P_j, \quad (2)$$

при этом размерности матриц:

$$sizeof(P_i) = (m_i, n_i),\ i = \overline{1,l}, \quad sizeof(P) = (\prod_{i=1}^{l} m_i, \prod_{i=1}^{l} n_i).$$

Свойство 1. Если A^{-1} и B^{-1} существуют, то

$$(A\otimes B)^{-1}=A^{-1}\otimes B^{-1}. \quad (3)$$

Свойство 2. Пусть $C(mp,nr) = A(m,n)\otimes B(p,r)$. Тогда сложность матрично-векторного умножения

$$Y(mp) = C(mp,nr)X(nr) \quad (4)$$

равна *O(mnpr)*, но может быть уменьшено до *O(npr + pmn)*, если использовать структуру КП матрицы *C*.

Доказательство: Рассмотрим алгоритм вычисления $Y = CX$, используя структуру КП. Из (1)**,**

$$Y_i = Y_{(i_1-1)p+i_2} = \sum_{j=1}^{nr} C_{ij}X_j = \sum_{j_1=1}^{n}\sum_{j_2=1}^{r} A_{i_1j_1}B_{i_2j_2}X_{(j_1-1)r+j_2} = \\ \sum_{j_1=1}^{n} A_{i_1j_1}Y^1_{(j_1-1)r+i_2}, \quad (5)$$

где

$$i_1 = \lfloor (i-1)/p \rfloor + 1, i_2 = (i-1) \bmod p + 1,$$
$$j_1 = \lfloor (j-1)/r \rfloor + 1,\ j_2 = (j-1) \bmod r + 1$$
$$Y^1_{(j_1-1)r+i_2} = \sum_{j_2=1}^{r} B_{i_2j_2}X_{(j_1-1)r+j_2},\ i = \overrightarrow{1,mp}, j = \overrightarrow{1,nr}. \quad (6)$$

Из (5), (6) следует, что сложность вычисления (6) равна *O(nrp)*, а сложность вычисления (5), с помощью Y^1 из (6), равна *O(mnp)*. Таким образом, сложность вычисления (4) равна *O(nrp)+O(mnp)=O(pn(m+r))*, ЧТД.

Из Свойства 2 следует

*Следствие 1***.** Вычислительная сложность УКПВ (4) *K=2* при *m=n=p=r* равна

$$BC\_KП(2,m)=O(2m^3). \quad (7)$$

2. Пример КП порядка $\boldsymbol{n = 8, K = log_2 n = 3}$. АБУКПВ основан на использовании Свойства 2. Рассмотрим вычисление

$$\boldsymbol{Y = (A\otimes B\otimes C)\cdot X = (A\otimes(B\otimes C))\cdot X}. \quad (8)$$

При размерности матриц-сомножителей $m \times m$, где *m*=2, выражение (8) можно представить как

$$(y1, y2, y3, y4, y5, y6, y7, y8)^{Tr} =$$

$$= \begin{pmatrix} a11\begin{pmatrix} b11\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x1 \\ x2 \end{pmatrix} + & b12\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x3 \\ x4 \end{pmatrix} \\ b21\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x1 \\ x2 \end{pmatrix} + & b22\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x3 \\ x4 \end{pmatrix} \end{pmatrix} + \\ +a12\begin{pmatrix} b11\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x5 \\ x6 \end{pmatrix} + & b12\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x7 \\ x8 \end{pmatrix} \\ b21\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x5 \\ x6 \end{pmatrix} + & b22\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x7 \\ x8 \end{pmatrix} \end{pmatrix} \\ a21\begin{pmatrix} b11\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x1 \\ x2 \end{pmatrix} + & b12\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x3 \\ x4 \end{pmatrix} \\ b21\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x1 \\ x2 \end{pmatrix} + & b22\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x3 \\ x4 \end{pmatrix} \end{pmatrix} + \\ +a22\begin{pmatrix} b11\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x5 \\ x6 \end{pmatrix} + & b12\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x7 \\ x8 \end{pmatrix} \\ b21\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x5 \\ x6 \end{pmatrix} + & b22\begin{pmatrix} c11 & c12 \\ c21 & c22 \end{pmatrix}\cdot\begin{pmatrix} x7 \\ x8 \end{pmatrix} \end{pmatrix} \end{pmatrix} X \quad (9)$$

Видим, что в правой части (9) каждая матрица С четыре раза умножается на одну и ту же часть вектора X (из четырех). Умножая, получаем:

$$\begin{pmatrix} a11\begin{pmatrix} b11\begin{pmatrix} y1^3 \\ y2^3 \end{pmatrix} + & b12\begin{pmatrix} y3^3 \\ y4^3 \end{pmatrix} \\ b21\begin{pmatrix} y1^3 \\ y2^3 \end{pmatrix} + & b22\begin{pmatrix} y3^3 \\ y4^3 \end{pmatrix} \end{pmatrix} + a12\begin{pmatrix} b11\begin{pmatrix} y5^3 \\ y6^3 \end{pmatrix} + & b12\begin{pmatrix} y7^3 \\ y8^3 \end{pmatrix} \\ b21\begin{pmatrix} y5^3 \\ y6^3 \end{pmatrix} + & b22\begin{pmatrix} y7^3 \\ y8^3 \end{pmatrix} \end{pmatrix} \\ a21\begin{pmatrix} b11\begin{pmatrix} y1^3 \\ y2^3 \end{pmatrix} + & b12\begin{pmatrix} y3^3 \\ y4^3 \end{pmatrix} \\ b21\begin{pmatrix} y1^3 \\ y2^3 \end{pmatrix} + & b22\begin{pmatrix} y3^3 \\ y4^3 \end{pmatrix} \end{pmatrix} + a22\begin{pmatrix} b11\begin{pmatrix} y5^3 \\ y6^3 \end{pmatrix} + & b12\begin{pmatrix} y7^3 \\ y8^3 \end{pmatrix} \\ b21\begin{pmatrix} y5^3 \\ y6^3 \end{pmatrix} + & b22\begin{pmatrix} y7^3 \\ y8^3 \end{pmatrix} \end{pmatrix} \end{pmatrix} = \begin{pmatrix} y1 \\ y2 \\ y3 \\ y4 \\ y5 \\ y6 \\ y7 \\ y8 \end{pmatrix}, \quad (10)$$

где

$$y1^3 = c11 \cdot x1 + c12 \cdot x2,\ y2^3 = c21 \cdot x1 + c22 \cdot x2,\ y3^3 = c11 \cdot x3 + \\ + c12 \cdot x4,\ y4^3 = c21 \cdot x3 + c22 \cdot x4,\ y5^3 = c11 \cdot x5 + c12 \cdot x6,\ y6^3 = \\ = c21 \cdot x5 + c22 \cdot x6,\ y7^3 = c11 \cdot x7 + c12 \cdot x8, y8^3 = c21 \cdot x7 + c22 \cdot x8.$$

Переставляя ряды в левой части и элементы вектора в правой части (10), получаем

$$\begin{pmatrix} a11\begin{pmatrix} \begin{pmatrix} b11 & b12 \\ b21 & b22 \end{pmatrix}\begin{pmatrix} y1^3 \\ y3^3 \end{pmatrix} \\ \begin{pmatrix} b11 & b12 \\ b21 & b22 \end{pmatrix}\begin{pmatrix} y2^3 \\ y4^3 \end{pmatrix} \end{pmatrix} + a12\begin{pmatrix} \begin{pmatrix} b11 & b12 \\ b21 & b22 \end{pmatrix}\begin{pmatrix} y5^3 \\ y7^3 \end{pmatrix} \\ \begin{pmatrix} b11 & b12 \\ b21 & b22 \end{pmatrix}\begin{pmatrix} y6^3 \\ y8^3 \end{pmatrix} \end{pmatrix} \\ a21\begin{pmatrix} \begin{pmatrix} b11 & b12 \\ b21 & b22 \end{pmatrix}\begin{pmatrix} y1^3 \\ y3^3 \end{pmatrix} \\ \begin{pmatrix} b11 & b12 \\ b21 & b22 \end{pmatrix}\begin{pmatrix} y2^3 \\ y4^3 \end{pmatrix} \end{pmatrix} + a22\begin{pmatrix} \begin{pmatrix} b11 & b12 \\ b21 & b22 \end{pmatrix}\begin{pmatrix} y5^3 \\ y7^3 \end{pmatrix} \\ \begin{pmatrix} b11 & b12 \\ b21 & b22 \end{pmatrix}\begin{pmatrix} y6^3 \\ y8^3 \end{pmatrix} \end{pmatrix} \end{pmatrix} = \begin{pmatrix} y1 \\ y3 \\ y2 \\ y4 \\ y5 \\ y7 \\ y6 \\ y8 \end{pmatrix} \quad (11)$$

В (11) каждая матрица В умножается два раза на одну и ту же часть вектора X. Умножая, получаем:

$$\begin{pmatrix} a11\begin{pmatrix} \begin{pmatrix} y1^2 \\ y2^2 \end{pmatrix} \\ \begin{pmatrix} y3^2 \\ y4^2 \end{pmatrix} \end{pmatrix} + a12\begin{pmatrix} \begin{pmatrix} y5^2 \\ y6^2 \end{pmatrix} \\ \begin{pmatrix} y7^2 \\ y8^2 \end{pmatrix} \end{pmatrix} \\ a21\begin{pmatrix} \begin{pmatrix} y1^2 \\ y2^2 \end{pmatrix} \\ \begin{pmatrix} y3^2 \\ y4^2 \end{pmatrix} \end{pmatrix} + a22\begin{pmatrix} \begin{pmatrix} y5^2 \\ y6^2 \end{pmatrix} \\ \begin{pmatrix} y7^2 \\ y8^2 \end{pmatrix} \end{pmatrix} \end{pmatrix} = \begin{pmatrix} y1 \\ y3 \\ y2 \\ y4 \\ y5 \\ y7 \\ y6 \\ y8 \end{pmatrix}, \quad (12)$$

где

$$y1^2 = b11 \cdot y1^3 + b12 \cdot y3^3,\ y2^2 = b21 \cdot y1^3 + b22 \cdot y3^3,\ y3^2 =$$
$$= b11 \cdot y2^3 + b12 \cdot y4^3,\ y4^2 = b21 \cdot y2^3 + b22 \cdot y4^3,\ y5^2 = b11 \cdot y5^3 + +b12 \cdot y7^3,$$
$$y6^2 = b21 \cdot y5^3 + b22 \cdot y7^3,\ y7^2 = b11 \cdot y6^3 + b12 \cdot y8^3,\ y8^2 = b21 \cdot y6^3 + b22 \cdot y8^3.$$

Переставляя ряды матрицы в левой части и соответствующие элементы вектора в правой части (12), получаем

$$\begin{pmatrix} \begin{pmatrix} a11 & a12 \\ a21 & a22 \end{pmatrix} \cdot \begin{pmatrix} y1^2 \\ y5^2 \end{pmatrix} \\ \begin{pmatrix} a11 & a12 \\ a21 & a22 \end{pmatrix} \cdot \begin{pmatrix} y2^2 \\ y6^2 \end{pmatrix} \\ \begin{pmatrix} a11 & a12 \\ a21 & a22 \end{pmatrix} \cdot \begin{pmatrix} y3^2 \\ y7^2 \end{pmatrix} \\ \begin{pmatrix} a11 & a12 \\ a21 & a22 \end{pmatrix} \cdot \begin{pmatrix} y4^2 \\ y8^2 \end{pmatrix} \end{pmatrix} = \begin{pmatrix} y1^1 \\ y2^1 \\ y3^1 \\ y4^1 \\ y5^1 \\ y6^1 \\ y7^1 \\ y8^1 \end{pmatrix} = \begin{pmatrix} y1 \\ y5 \\ y2 \\ y6 \\ y3 \\ y7 \\ y4 \\ y8 \end{pmatrix}. \quad (13)$$

В (13) каждая матрица А умножается теперь однократно на каждую из четырех частей вектора X.

3. Вычислительная схема для реализации АБУКПВ. Преобразования (9)–(13) можно представить в виде схемы на рис. 1, где все матрицы A, B, C обозначены прямоугольниками и для каждого из четырех экземпляров каждой из этих матриц указаны соответствующие входные и выходные данные.

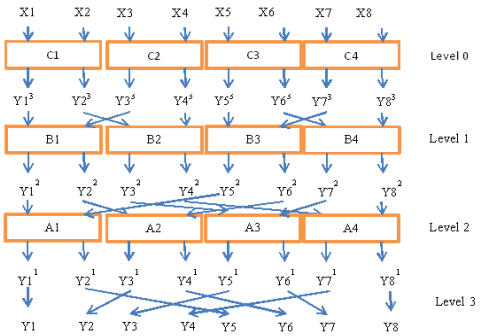


Рис. 1. Схема вычислений

На рис. 2 представлена эта же схема УКПВ, но уже с двоичными номерами входов и выходов каждого блока умножения.

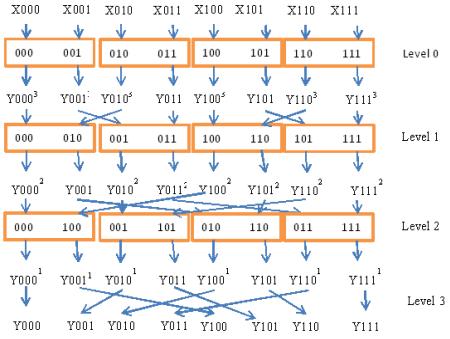


Рис. 2. Схема УКПВ с указанием двоичных номеров входов и выходов блоков умножения

Можно видеть, что схема УКПВ имеет $log_2 n = 3$ уровня, в каждом по $\frac{n}{2} = 4$ схемы матричного 2×2 умножения. Внутри прямоугольника каждой схемы проставлены два кода, которые показывают, какие компоненты выходного вектора предыдущего уровня соединены с этими входами. Можно видеть, что на рис. 2 соединены входы-выходы соседних уровней с одинаковыми номерами. На уровне 3 показаны окончательные значения результирующего вектора *Y*.

Схема УКПВ для $n = 2^K$ имеет $log_2 n = K$ уровней с $\frac{n}{2}$ матричными 2×2 умножителями, пронумерованными в каждом уровне $l = 0..K-1$ от 0 до $\frac{n}{2} - 1$ $(K-1)$-битными двоичными числами. При этом входы/выходы схемы с номером $i_{K-2}..i_0$ уровня l отличаются значением l-го бита с остальными битами со значениями бит из номера схемы, причем взаимный порядок этих бит в номере входа/выхода тот же, что и в номере схемы. Соединены между собой входы и выходы схем соседних уровней с одинаковыми номерами. Входы схем уровня 0 соединены с элементами входного вектора с теми же номерами. Выходы схем последнего уровня соединены с элеменами результирующего вектора с теми же номерами. Такой порядок соединений характерен для гиперкубов. При использовании физически параллельных схем матричного умножения время УКПВ имеет порядок $O(K) = O(log_2 n)$. При использовании схемы УКПВ в конвейерном режиме результаты выдаются с задержкой порядка $O(1)$ после заполнения конвейера.

Заключение. На основе предложенного метода ускоренного умножения кронекерова произведения матриц на вектор представлены аппаратурно-ориентированные вычислительные структуры, которые существенно повышают его эффективность за счет исключения предварительного отдельного вычисления КП и отсутствия необходимости хранения в памяти его результирующей матрицы.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Van Loan C.F*. The ubiquitous Kronecker product // Journal of Computational and Applied Mathematics. – 2000. – No. 123. – P. 85-100.
2. *Jemderson H.V., Pukelsheim F., and Searle S.R.* On the history of the Kronecker product // Linear and Multilinear Algebra. – 1983. – Vol. 14, No. 2. – P. 113-120.
3. *Graham A.* Kronecker Products and Matrix Calculus with Applications (Dover Books on Mathematics), Dover Publications, 2018.
4. *Buchholz P., Ciardo G., Donatelli S., Kemper P.* Complexity of memory-efficient Kronecker operations with applications to the solution of Markov models // INFORMS J. Comput. – 2000. – P. 203-222.
5. *Tadonki C., Philippe B.* Parallel multiplication of a vector by a Kronecker product of matrices // Journal of Parallel and Distributed Computing and Practices. – 2000. – No. 3 (3). – P. 1-11.
6. *Tadonki C.* Large-scale Kronecker product on supercomputers // 2011 Second Workshop on Architecture and Multi-Core Applications (WAMCA 2011). 26-27 Oct. 2011. – Victoria, Brazil. IEEE. – P. 1-4.
7. *Doukhnitch E., Strelnikov O., Andreev A.,* Application of Kronecker Matrix Product for the Synthesis of Hardware-oriented DSP Algorithms // in Proc. of Intern. Conf. on Signal Proc. “DSPA-99”, Moscow, Sept. 1999. – P. 78-83.
8. *Doukhnitch E., Salamah M., Andreev A.* Effective Processor for Matrix Decomposition // Arabian Journal for Science and Engineering. – March 2014. – Vol. 39, Issue 3. – P. 1797-1804.
9. *Dayar T., Orhan M.C.* On vector-Kronecker product multiplication with rectangular factors // SIAM J. Sci. Comput. – 2015. – Vol. 37 (5). – P. S526-S543.
10. *Koukouvinos C., Lappas E., Simos D.E.* Encryption schemes using orthogonal arrays // Journal of Discrete Mathematical Sciences and Cryptography. – 2009. – No. 12 (5). – P. 615-628.
11. *Chefranov A., Dukhnich E.* One-time Kronecker product-based Hill cipher modification // International Journal of Information Assurance and Security. – 2017. – No. 12 (3). – P. 94-103.
12. *Lancaster P. and Tismenetsky M.,* The Theory of Matrices. – 2nd ed. – Orlando, Florida: AcademicPress, 1985.

13. *Chefranov A., Dukhnich E., Shapel A.* One-Time Involutory Matrix-Based Hill Cipher Modification // International Journal of Information Assurance and Security (JIAS). – 2020. – Vol. 15, No. 4. – P. 165-174.
14. *Джамбруно М.* Трехмерная (3D) графика и анимация. – М.: Вильямс, 2002. – 640 с.
15. *Dzwonkowski M., Rykaczewski R.* Secure quaternion Feistel cipher (S-QFC) for DICOM images // IEEE Transactions on Image Processing. – 2019. – Vol. 28, No. 1. – P. 371-380.
16. *Dzwonkowski M., Papaj M., Rykac R.* A new quaternion-based encryption method for DICOM Images // IEEE Transactions on Image Processing. – 2015. – Vol. 24, No. 11. – P. 4614-4522.
17. *Гантмахер Ф.Р.* Теория матриц. – М.: Главная редакция физико-математической литературы издательства "Наука", 1966.
18. *Blair Jeffrey S.* The Biomedical Engineering handbook. CRC Press Taylor & Francis Group, New York, 2006. – P. 42-4–42-5.
19. *Deng Y., Guo M., Ramos A.F., et al.* Optimal low-latency network topologies for cluster performance enhancement // J. Supercomput. Published online 02 March 2020. – https://doi.org/10.1007/s11227-020-03216-y.
20. *Hayes J.P., Mudge T.* Hypercube supercomputers // Proceedings of the IEEE. – 1989. – Vol. 77 (12). – P. 1829-1841.

REFERENCES

1. *Van Loan C.F.* The ubiquitous Kronecker product, *Journal of Computational and Applied Mathematics*, 2000, No. 123, pp. 85-100.
2. *Jemderson H.V., Pukelsheim F., and Searle S.R.* On the history of the Kronecker product, *Linear and Multilinear Algebra*, 1983, Vol. 14, No. 2, pp. 113-120.
3. *Graham A.* Kronecker Products and Matrix Calculus with Applications (Dover Books on Mathematics), Dover Publications, 2018.
4. *Buchholz P., Ciardo G., Donatelli S., Kemper P.* Complexity of memory-efficient Kronecker operations with applications to the solution of Markov models, *INFORMS J. Comput*., 2000, pp. 203-222.
5. *Tadonki C., Philippe B.* Parallel multiplication of a vector by a Kronecker product of matrices, *Journal of Parallel and Distributed Computing and Practices*, 2000, No. 3 (3), pp. 1-11.
6. *Tadonki C.* Large-scale Kronecker product on supercomputers, *2011 Second Workshop on Architecture and Multi-Core Applications (WAMCA 2011). 26-27 Oct. 2011*. Victoria, Brazil. IEEE, pp. 1-4.
7. *Doukhnitch E., Strelnikov O., Andreev A.,* Application of Kronecker Matrix Product for the Synthesis of Hardware-oriented DSP Algorithms, *in Proc. of Intern. Conf. on Signal Proc. "DSPA-99", Moscow, Sept. 1999*, pp. 78-83.
8. *Doukhnitch E., Salamah M., Andreev A.* Effective Processor for Matrix Decomposition, *Arabian Journal for Science and Engineering*, March 2014, Vol. 39, Issue 3, pp. 1797-1804.
9. *Dayar T., Orhan M.C.* On vector-Kronecker product multiplication with rectangular factors, *SIAM J. Sci. Comput*., 2015, Vol. 37 (5), p. S526-S543.
10. *Koukouvinos C., Lappas E., Simos D.E.* Encryption schemes using orthogonal arrays, *Journal of Discrete Mathematical Sciences and Cryptography*, 2009, No. 12 (5), pp. 615-628.
11. *Chefranov A., Dukhnich E.* One-time Kronecker product-based Hill cipher modification, *International Journal of Information Assurance and Security*, 2017, No. 12 (3), pp. 94-103.
12. *Lancaster P. and Tismenetsky M.,* The Theory of Matrices. 2nd ed. Orlando, Florida: AcademicPress, 1985.
13. *Chefranov A., Dukhnich E., Shapel A.* One-Time Involutory Matrix-Based Hill Cipher Modification, *International Journal of Information Assurance and Security (JIAS)*, 2020, Vol. 15, No. 4, pp. 165-174.
14. *Dzhambruno M.* Trekhmernaya (3D) grafika i animatsiya [3D Graphics &Animation]. Moscow: Vil'yams, 2002, 640 p.
15. *Dzwonkowski M., Rykaczewski R.* Secure quaternion Feistel cipher (S-QFC) for DICOM images, *IEEE Transactions on Image Processing*, 2019, Vol. 28, No. 1, pp. 371-380.
16. *Dzwonkowski M., Papaj M., Rykac R.* A new quaternion-based encryption method for DICOM Images, *IEEE Transactions on Image Processing*, 2015, Vol. 24, No. 11, pp. 4614-4522.
17. *Gantmakher F.R.* Teoriya matrits [The theory of matrices]. Moscow: Glavnaya redaktsiya fiziko-matematicheskoy literatury izdatel'stva "Nauka", 1966.

18. *Blair Jeffrey S.* The Biomedical Engineering handbook. CRC Press Taylor & Francis Group, New York, 2006, pp. 42-4–42-5.
19. *Deng Y., Guo M., Ramos A.F., et al.* Optimal low-latency network topologies for cluster performance enhancement, *J. Supercomput. Published online 02 March 2020*. Available at: https://doi.org/10.1007/s11227-020-03216-y.
20. *Hayes J.P., Mudge T.* Hypercube supercomputers, *Proceedings of the IEEE,* 1989, Vol. 77 (12), pp. 1829-1841.

Статью рекомендовал к опубликованию д.т.н., профессор Ю.М. Вишняков.

Духнич Евгений Иванович – Государственный Морской университет имени адмирала Ф.Ф. Ушакова; e-mail: evgenydukhnich@gmail.com; г. Новороссийск, Россия; тел.: +79184907411; кафедра радиоэлектроники и информационных технологий; д.т.н.; профессор.

Чефранов Александр Гергиевич – Восточный Средиземноморский университет; e-mail: alexander.chefranov@emu.edu.tr; г. Фамагуста, С. Кипр; тел.: +05338673790; кафедра компьютерных технологий; д.т.н.; профессор.

Dukhnich Evgeny Ivanovich – Novorossiysk State Maritime University; e-mail: evgenydukhnich@gmail.com; Novorossiysk, Russia; phone: +79184907411; the department of radio-electronics and information technologies; dr. of eng. sc.; professor.

Chefranov Alexander Georgievich – Eastern Mediterranean University, e-mail: alexander.chefranov@emu.edu.tr; t. Famagusta, N. Cyprus; phone: +05338673790; the department of computer engineering; dr. of eng. sc.; professor.

УДК 519.224.22 DOI 10.18522/2311-3103-2020-7-52-67

А.К. Мельников

АЛГОРИТМИЧЕСКАЯ СЛОЖНОСТЬ РАСЧЕТА ТОЧНЫХ ПРИБЛИЖЕНИЙ РАСПРЕДЕЛЕНИЙ ВЕРОЯТНОСТЕЙ ЗНАЧЕНИЙ СТАТИСТИК МЕТОДОМ РЕШЕНИЯ УРАВНЕНИЯ ПЕРВОЙ КРАТНОСТИ ТИПОВ

Рассматривается алгоритмическая сложность расчета точных распределений вероятностей значений статистик и их точных приближений методом решения уравнения первой кратности. В качестве точных приближений распределений вероятностей значений статистик рассматриваются их Δ–точные распределения, отличающиеся от точных распределений не более чем на заранее заданную, сколь угодно малую величину Δ. Показывается, что основой метода расчета точных распределений вероятностей значений статистик является перечисление элементов области поиска решений линейного уравнения кратности типов, составленной из векторов кратности типов, каждый элемент которого представляет собой число вхождений элементов определенного типа (какого-либо знака алфавита) в рассматриваемую выборку. Одновременно показывается, что для расчета точных приближений распределения вероятностей значений статистик применяется метод ограничения области поиска решений. Приводится выражение определяющее алгоритмическую сложность вычисления точных распределений методом решения уравнения первой кратности. Приведенное выражение является конечным и позволяет для каждого значения мощности алфавита определить максимальный объем выборки, для которой при использовании ограниченного вычислительного ресурса методом решения уравнения первой кратности могут быть рассчитаны точные распределения. Определена область параметров, представляемых объемом выборок и мощностью алфавита, для которых при ограниченном вычислительном ресурсе могут быть рассчитаны точные распределения. Для оценки алгоритмической сложности расчета точных приближений распределений приводится, впервые полученное, выражение для числа решений уравнения первой кратности с ограничением на значения координат векторов решений. Приводится выражение

определяющее алгоритмическую сложность вычисления точных приближений методом решения уравнения первой кратности с ограничением на значения координат векторов решений. В качестве параметра ограничения координат векторов решений используется значение статистики максимальной частоты, вероятность превышения которого меньше заранее заданной, сколь угодно малой величины Δ, что позволяет рассчитывать точные приближения распределений, отличающиеся от их точных распределений не более чем на выбранную величину Δ. Приведенное выражение является конечным и позволяет для каждого значения алфавита определить максимальный объем выборки, для которой при использовании ограниченного вычислительного ресурса методом решения уравнения первой кратности при ограничениях задаваемых с помощью величины Δ могут быть рассчитаны точные приближения. Приводятся результаты вычислений максимальных объемов выборок для которых могут быть рассчитаны точные приближения. Показывается, что алгоритмическая сложность расчета точных распределений на много порядков превосходит сложность расчета их точных приближений. Показано, что применение метода первой кратности для расчета точных приближений позволяет при одинаковых значениях мощности алфавита увеличить, по сравнению с расчетом точных распределений, объём выборок в два и более раз.

Вероятность; статистика; точное распределение; точное приближение; вектор кратности типов; линейное уравнение; алгоритмическая сложность.

A.K. Melnikov

ALGORITHMIC COMPLEXITY OF CALCULATING EXACT APPROXIMATIONS OF PROBABILITY DISTRIBUTIONS OF STATISTICAL VALUES BY SOLVING THE EQUATION OF THE FIRST MULTIPLICITY OF TYPES

We consider the algorithmic complexity of calculating the exact probability distributions of statistical values and their exact approximations by solving the first multiplicity equation. As exact approximations of probability distributions of statistical values, we consider their Δ−exact distributions that differ from the exact distributions by no more than a predetermined, arbitrarily small valueΔ. It is shown that the basis of the method for calculating the exact probability distributions of statistical values is the enumeration of elements of the search area for solutions to a linear equation of multiplicity of types, composed of vectors of multiplicity of types, each element of which is the number of occurrences of elements of a certain type (any sign of the alphabet) in the sample under consideration. At the same time, it is shown that the method of limiting the search area for solutions is used to calculate exact approximations of the probability distribution of statistical values. An expression is given that defines the algorithmic complexity of calculating exact distributions by solving the first multiplicity equation. The given expression is finite and allows for each value of the alphabet power to determine the maximum sample size for which, using a limited computational resource, exact distributions can be calculated by solving the first multiplicity equation. The range of parameters represented by the sample size and alphabet power for which exact distributions can be calculated with a limited computing resource is defined. To estimate the algorithmic complexity of calculating exact approximations of distributions, we present an expression for the first time obtained for the number of solutions to the equation of the first multiplicity with a restriction on the coordinate values of the solution vectors. An expression is given that defines the algorithmic complexity of calculating exact approximations by solving the first multiplicity equation with a restriction on the coordinate values of the solution vectors. As a parameter for limiting the coordinates of solution vectors, the maximum frequency statistic value is used, the probability of exceeding it is less than a pre-set, arbitrarily small valueΔ, which allows calculating exact approximations of distributions that differ from their exact distributions by no more than the selected value Δ. The given expression is finite and allows for each value of the alphabet to determine the maximum sample size for which, when using a limited computational resource, exact approximations can be calculated by solving the equation of the first multiplicity under the restrictions set using the valueΔ. The results of calculations of the maximum sample volumes for which exact approximations can be calculated are presented. It is shown that the algorithmic complexity of calculating exact distributions exceeds the complexity of calculating their exact ap-

proximations by many orders of magnitude. It is shown that the use of the first multiplicity method for calculating exact approximations allows for the same values of the alphabet power to increase the sample volume by two or more times compared to the calculation of exact distributions.

Probability; statistics; exact distribution; an accurate approximation of the vector of multiplicity of types; the linear equation algorithmic complexity.

Введение. Применение в критериях согласия с равновероятным распределением в качестве эталонного распределения точных распределений вероятностей значений статистик при заданном уровне значимости позволяет строить критерии с наибольшей относительной эффективностью [1]. Однако расчет точных распределений для многих значений параметров мощности алфавита N и объема выборки n (длины последовательности) не всегда возможен [2, 3], что, для решения задачи по статистическому анализу последовательностей (текстов) [4], заставляет пользоваться точными или предельными приближениями данных распределений [5, 6]. Применение точных приближений (Δ-точных распределений [7]) предпочтительнее перед применением предельных приближений, так как дает возможность строить критерии с большей относительной эффективностью [8]. При заданном уровне значимости и невозможности рассчитать точное распределение большую эффективность дает обработка текстов, построенная на критерии использующем точное приближение эталонного распределения, чем применение предельного распределения . Рассмотрение относительной алгоритмической сложности методов расчета точных распределений и их точных приближений показывает, что расчет точных приближений во много порядков проще расчета точных распределений [9]. Но для расчета значений параметров мощности алфавита и объема выборки, для которых могут быть рассчитаны точные приближения, учитывая используемый вычислительный ресурс, необходимо уметь рассчитывать алгоритмическую сложность методов расчета точных распределений и их точных приближений. Одним из таких методов [10] является метод решения уравнения первой кратности типов [11]. Данная работа посвящена оценке алгоритмической сложности расчета точных распределений вероятностей значений статистик и их точных приближений методом решения уравнения первой кратности типов.

Постановка задачи. Рассмотрим статистику $S(N,n) = f(h_1, ..., h_N)$, где h_i – частота встречаемости знака (исхода) a_i, n – длина текста (объем выборки), N – число исходов полиномиальной схемы (мощность алфавита A_N=$\{a_1, a_2, \dots, a_N\}$) и p_i – вероятность a_i -го исхода.

Одним из методов расчета точных значений распределения вероятности значений статистики $S(N,n)$ – $P\{ S(N,n) \geq c \}$ является расчет её значений для всех решений $(h_1^{(\nu)}, h_2^{(\nu)}, ..., h_N^{(\nu)})$ в неотрицательных целых числах линейного уравнения

$$h_1 + \dots + h_N = n, \qquad (1)$$

называемого ещё уравнением кратности типов или уравнением первой кратности типов. Поэтому данный метод справедливо можно назвать методом решения уравнения первой кратности типов или сокращенно *методом первой кратности* (МПК).

Решения уравнения (1) находятся путем перебора в лексикографическом порядке области поиска решений $R_{n,n}^N = \{h_i \mid i = \overline{1,N},\ h_i \in \mathrm{N},\ 0 \leq h_i \leq n\}$. При этом для расчета точных распределений $P_T\{ S(N,n) \geq c \}$ перебор координат вектора $(h_1^{(\nu)}, h_2^{(\nu)}, ..., h_N^{(\nu)})$ возможных решений производится в условии

$$\{h_i \mid i = \overline{1,N},\ h_i \in \mathrm{N},\ 0 \le h_i \le n\}. \tag{1.1}$$

Основой методики расчета точных приближений $P_\Delta\{\ S(N,n) \ge c\ \}$ [12] является ограничение области поиска решений до

$$R_{n,r}^{N} = \{h_i \mid i = \overline{1,N},\ h_i \in \mathrm{N},\ 0 \le h_i \le r\},$$

где $r < n$ является параметром ограничения. Соответственно для расчета точных приближений перебор координат вектора $(h_1^{(\nu)}, h_2^{(\nu)}, \ldots, h_N^{(\nu)})$ возможных решений уравнения (1) производится в условии

$$\{h_i \mid i = \overline{1,N},\ h_i \in \mathrm{N},\ 0 \le h_i \le r\}. \tag{1.2}$$

Для оценка и сравнение между собой сложностей вычисления, числа обобщенных операций (алгоритмической сложности), точного распределения $P_{ext}\{\ S(N,n) \ge c\ \}$ – $C(P_{ext}\{\ S(N,n) \ge c\ \})$ и его точного приближения $P_\Delta\{\ S(N,n) \ge c\ \}$ – $C(P_\Delta\{\ S(N,n) \ge c\ \})$ необходимо получить их аналитические выражения.

Алгоритмическая сложность расчета точных распределений вероятностей значений статистик методом решения уравнения первой кратности типов. Количество векторов $(h_1^{(\nu)}, h_2^{(\nu)}, \ldots, h_N^{(\nu)})$ – $K_h(N,n,n)$, равное равно числу целочисленных неотрицательных решений уравнения (1) в условиях (1.1), равное числу сочетаний с повторениями $\overline{C}_N^n$, связанному с числом сочетаний следующим соотношением [11] (стр. 127)

$$K_h(N,n,n) = \overline{C}_N^n = C_{N+n-1}^n = \frac{(N+n-1)!}{n!\cdot(N-1)!}. \tag{2}$$

Тогда алгоритмическая сложность расчета точных распределений методом первой кратности может быть выражена следующим образом

$$C_{МПК}(P_T\{S_{N,n} \ge c\}) = N^n \cdot C(N,n,n,\overrightarrow{h(N,n)}^{(\nu)}) + \tag{3}$$

$$+ C_{N+n-1}^n \cdot (C(S_{N,n}(\overrightarrow{h(N,n)})) + C(P(S_{N,n} = S_{N,n}(\overrightarrow{h(N,n)}^{(i)})) +$$

$$+ C(P\{S_{N,n} \ge c\}, \{S_{N,n}^{(i)}, P_{N,n}^{(i)} \mid i = 1,\ldots, C_{N+n-1}^n\}),$$

где $C(N,n,n,\overrightarrow{h(N,n)}^{(\nu)})$ есть алгоритмическая сложность генерации очередного, отличающегося от уже сгенерированных до этого, вектора частот $\overrightarrow{h(N,n)}$ без ограничения на координаты (N,n,n) и его проверки на удовлетворение уравнению (1), а $C(S_{N,n}(h(N,n))$ есть алгоритмическая сложность вычисления значения статистики $S_{N,n}$ от вектора первой кратности типов $\overrightarrow{h(N,n)}^{(i)}$ от первой (частотной) маркировки, $C(P(S_{N,n} = S_{N,n}(\overrightarrow{h(N,n)}^{(i)})$ сложность вычисление вероятности, с которой статистика $S_{N,n}$ примет значений $S_{N,n} = S_{N,n}(\overrightarrow{h(N,n)}^{(i)})$

$$P_{N,n}^{(i)} = P(S_{N,n} = S_{N,n}(\overrightarrow{h(N,n)}^{(i)}) = \frac{n!}{h_1^{(i)}!\cdot h_2^{(i)}!\cdot \ldots \cdot h_N^{(i)}}$$

и $C(P\{S_{N,n} \ge c\}, \{S^{(i)}_{h(N,n)}, P^{S(i)}_{h(N,n)} \mid i = 1,...,C^n_{N+n-1}\})$ есть алгоритмическая сложность расчета распределения вероятностей $P_T\{S_{N,n} \ge c\}$ (таблицы распределения) используя рассчитанные значения статистики и их вероятности $\{S^{(i)}_{h(N,n)}, P^{S(i)}_{h(N,n)} \mid i = 1,..., C^n_{N+n-1}\})$.

Примем следующие ограничения. Будем считать, что:

♦ для генерации в лексикографическом порядке очередного, отличающегося от уже сгенерированных до этого векторов, вектора частот $\overrightarrow{h(N,n)}$ длины *N*, у которого каждая координата изменяется от 0 до *n*, без ограничения на координаты (N,n,n), и его проверки на удовлетворение уравнению (1) потребуется не более 3*N* операций

$$C(N,n,n,\overrightarrow{h(N,n)}^{(\nu)}) \le 3N;$$

♦ не нарушая общности можно рассмотреть в качестве статистики $S_{N,n}$ статистику хи-квадрат $f = \chi_{N,n}$, имеющую вид [13]

$$\chi_{N,n} = \chi_n(h_1,h_2,...,h_N) = \sum_{i=1}^{N} \frac{(h_i - np_i)^2}{np_i}.$$

Тогда сложность расчета значения статистики $S_{N,n} = S_{N,n}(\overrightarrow{h(N,n)}^{(i)})$ от вектора первой кратности типов $\overrightarrow{h(N,n)}^{(i)}$ и записи её значения в результирующий массив $R_{h(N,n)}[1:K_h(N,n,n)]$ потребуется не более 6*N* операций

$$C(S_{N,n}(\overrightarrow{h(N,n)})) \le 6N;$$

♦ для вычисления вероятности $P^{S(i)}_{h(N,n)}$, с которым статистика $S_{h(N,n)}$ примет значение $S_{N,n}(\overrightarrow{h(N,n)}^{(i)})$ и записи её значения в результирующий массив потребуется не более 3*N* операций с учетом того, что значения *n*! и N^n будут вычислены заранее, а значения частот $h^{(i)}_j$ распределены равномерно и можно считать, что $h^{(i)}_j \approx n/N$.

Будем считать, что для вычисления $h^{(i)}_j!$ в условиях $h^{(i)}_j \approx n/N$ требуется не более *n*/*N* операций. Тогда для вычисления всех *N* значений $\{h^{(i)}_j! \mid j = \overline{1,N}\}$ с учетом того, что все $h^{(i)}_j$ удовлетворяют условию (1) потребуется не менее (*N*+*n*) операций, а для вычисления их произведения $h^{(i)}_1! \cdot h^{(i)}_2! \cdot \ldots \cdot h^{(i)}_N!$ потребуется еще не более *N* операций. Следовательно для вычисления $P^{S(i)}_{h(N,n)}$ в соответствии с формулой (3) при заранее рассчитанных значениях $n!$ и $1/N^n$ потребуется не более (2*N*+*n*+3) операций

$$C(P(S_{h(N,n)} = S_{N,n}(\overrightarrow{h(N,n)}^{(i)})) \le 2 \cdot N + n + 3 \cdot$$

♦ для вычисления распределения $P\{S_{N,n} \ge c\}$ по массиву $R_{h(N,n)}[1 : K_h(N,n,n)]$ необходимо вначале его отсортировать, а потом однопроходным суммированием по формуле

$$P_T(S_{N,n} \ge c_i) = 1 - \sum_{j=1}^{K_h(N,n,n)} (P^{S(j)}_{h(N,n)}, \textit{если } S^{(j)}_{h(N,n)} < c_i)$$

получить все составляющие $P_T(S_{N,n} \ge c_i) \cdot$ Для сортировки методом Шелла [14] по возрастанию массива $R_{h(n,N)}$, состоящего из $K_h(N,n,n)$ двойных элементов $(S^{(i)}_{h(N,n)}, P^{S(i)}_{h(N,n)})$ по ключу $S^{(i)}_{h(N,n)}$ необходимо порядка $2 \times K_h(N,n,n) \cdot \log_2 K_h(N,n,n)$ операций. Для однопроходного просмотра отсортированного массива $R_{h(n,N)}$ на сравнение $S^{(i)}_{h(N,n)}$ с c_j и при необходимости суммирования и получения всех $P_T(S_{N,n} \ge c_j)$ для $c_j = \overline{1,100}$ необходимо $2 \times K_h(N,n,n)$ операций.

Теперь в соответствии с принятыми ограничениями, выясненным фактом $K_h(N,n,n) = C^n_{N+n-1}$ и (3) поучаем значение для ограничения алгоритмической сложности метода первой кратности

$$C_{МПК}(P_T\{S_{N,n} \ge c\}) \le (n+1)^N \cdot 3N + C^n_{N+n-1} \cdot (6 \cdot N + 2 \cdot N + n + 3) +$$
$$+ 2 \times C^n_{N+n-1} \cdot \log_2 C^n_{N+n-1} + 2 \times C^n_{N+n-1} =$$
$$= (n+1)^N \cdot 3N + C^n_{N+n-1} \cdot (8 \cdot N + n + 2 \cdot \log_2 C^n_{N+n-1} + 5) \cdot \qquad (4)$$

Выражение (4) позволяет определить алгоритмическую сложность вычисления точных распределений вероятностей значений статистик методом решения уравнения первой кратности для любых значений длин последовательностей *n* и мощностей алфавита *N*.

Так же выражение (4) может быть использовано для определения значений параметров распределений при фиксированном вычислительном ресурсе, выделенном для его расчета.

Оценка максимальных значений параметров выборок для расчета точных распределений на современной вычислительной технике. Пусть $P_{МВС}$ доступная производительность в операциях в секунду многопроцессорной вычислительной системы (МВС) и *t* предоставляемое время в секундах её использования для поведения расчетов. Тогда предоставляемый вычислительный ресурс $R_{МВС}$ равен

$$R_{МВС} = P_{МВС} \cdot t$$

и значения параметров *N* и *n* вычисляются из соотношений

$$C_{МПК}(P_T\{S_{N,n} \ge c\}) \le R_{МВС} \cdot \qquad (5)$$

Анализ 55-й редакции от 22 июня 2020 года списка Top500 [15] 500 наиболее мощных компьютеров мира показывает, что пиковая производительность находящегося на первом месте списка японского суперкомпьютера Fugaku производства фирмы Fujitsu на базе процессоров ARM A64FX, установленного в RIKEN Center for Computational Science (R-CCS), равна 513.9 PFlop/s, (0,514 х 10^{18} оп./сек.) а производительность на тесте Linpack - 415.5 PFlop/s (0,41614 х 10^{18} оп./сек.).

Следовательно предположение, что нам может быть доступна МВС производительностью 10^{18} операций в секунду обоснован. Это могут быть МВС как кластерного типа, основанные на взаимодействии универсальных процессоров с классической фон-неймановской архитектурой [16, 17] (предложенной ещё в 1945 году Джо фон Нейманом), так и реконфигурируемые [18], основанные на программируемых логических схемах (ПЛИС). Пусть этот доступ может осуществляться в течении 1 (одного) месяца (2 592 000 секунды), тогда предоставленный вычислительный ресурс МВС R_{MBC} имеет значение

$$R_{MBC} = 2{,}592 \cdot 10^{24}$$

и параметры N и n, для которых могут быть вычислены точные распределения вычисляются из соотношений

$$C_{МПК}(P_T\{S_{N,n} \geq c\}) \leq 2{,}592 \cdot 10^{24}.$$

Значения параметров N и n частично приведены в таблице 1, область их значений показана на рис. 1.

Таблица 1

Параметры N и n, для которых при использовании МВС производительностью 10^{18} оп/сек. за один месяц методом первой кратности могут быть рассчитаны точные распределения (ВР=2,592 х 10^{24})

№ п/п	Параметры		$C_{МПК}(P_T\{S_{N,n} \geq c\})$	$C_{МПК}(P_T\{S_{N,n+1} \geq c\})$
	N	n		
1.	12	79	2,4739E+24	2,87159E+24
2.	14	41	2,23223E+24	3,10319E+24
3.	16	25	2,09322E+24	3,82879E+24
4.	18	19	1,41558E+25	3,40676E+25
5.	20	12	1,1403E+24	5,0201E+24
6.	22	9	6,60000E+23	5,37258E+24
7.	24	7	3,4001E+23	5,74318E+24
8.	26	6	7,32223E+23	2,35741E+25
9.	28	5	2,35741E+25	3,86389E+25
10.	30	4	8,3819E+22	1,98967E+25
11.	32	4	2,23517E+24	7,64031E+26
12.	34	3	3,01051E+22	5,93718E+25
13.	36	3	5,10016E+23	1,57161E+27
14.	38	2	1,53997E+20	8,6136E+24
15.	40	2	1,45892E+21	1,45071E+26
16.	46	2	1,22309E+24	6,83343E+29
17.	48	1	4,05324E+16	1,14864E+25
18.	50	1	1,68885E+17	1,07685E+26
19.	58	1	5,01521E+19	8,19562E+29
20.	64	1	3,54177E+21	6,59267E+32
21.	68	1	6,02102E+22	5,67382E+34
22.	72	1	1,02003E+24	4,86613E+36
23.	74	0	4,19346E+24	4,50117E+37

Рис. 1. Область параметров выборок, для которых методом первой кратности могут быть посчитаны точные распределения

Анализ значений параметров N и n, приведенных в табл. 1 и области их изменений, показанных на рис. 1, говорит о том, для значений параметров N>46, метод первой кратности не позволяет рассчитывать точные распределения вероятностей значений статистик. Поэтому перейдем к рассмотрению алгоритмической сложности расчета точных приближений распределения вероятностей значений статистик методом первой кратности, в качестве которых рассматриваются Δ-точные распределения [7].

Алгоритмическая сложность расчета точных приближений распределения методом решения уравнения первой кратности. При ограничении области поиска решений уравнения (1) до

$$R_{n,r}^{N} = \{h_i \mid i = \overline{1,N},\ h_i \in \mathrm{N},\ 0 \le h_i \le r\},$$

где $r < n$ возникает вопрос об оценке числа этих решений $K_h(N,n,r)$.

Опираясь на результаты Сачкова В.Н., определяющие число композиций [19] (выражение 3.23 стр. 215) и используя известный метод установления взаимно однозначного соответствия, получаем выражение определяющее число целочисленных неотрицательных решений уравнения (1) в условиях (1.2), равное

$$K_h(N,n,r) = \sum_{i=0}^{\min(N,\frac{n}{r+1})} (-1)^i \binom{N}{i} \binom{N+n-1-(r+1)i}{N-1}. \qquad (6)$$

Обращаем внимание, что выражение (6), в прямой постановке, получено автором впервые. Опираясь на результат (6) получаем выражение для алгоритмической сложности расчета точных приближений распределений методом МПК с ограничениями r.

$$C_{МПК}(P_\Delta\{S_{N,n} \ge c\}) = (r+1)^N \cdot C(N,n,r,\overrightarrow{h(N,n)}^{(\nu)}) + \qquad (7)$$

$$+ \cdot K_h(N,n,r) \cdot (C(S_{N,n}(\overrightarrow{h(N,n)}^{(i)}) + C(P(S_{N,n} = S_{N,n}(\overrightarrow{h(N,n)}^{(i)}))) +$$

$$+ C(P\{S_{N,n} \ge c\}, \{S_{N,n}^{(i)}, P_{N,n}^{(i)} \mid i = 1,\ldots,K_h(N,n,r)\})$$

где $\cdot C(N,n,r,\overrightarrow{h(N,n)}^{(\nu)})$ есть алгоритмическая сложность генерации очередного, отличающегося от уже сгенерированных до этого, вектора частот $\overrightarrow{h(N,n)}$ с ограничениями на координаты (N,n,r) и его проверки на удовлетворение уравнению (1), а остальные обозначения аналогичны обозначениям используемым в (3).

Как было показано в [7], для того чтобы точное приближение отличалось от точного значения распределения не более чем на сколь угодно малую величину Δ

$$| P_T\{S_{N,n} \ge c\} - P_T\{S_{N,n} \ge c\} | \le \Delta \quad (8)$$

необходимо, чтобы $r = m(N,n,\Delta)$, удовлетворяло условию

$$P\{M_n > m(N,n,\Delta)\} \le 1-\Delta,$$

где M_n статистика максимальной частоты, а рекуррентная формула для вычисления значения её вероятности приведена в [8]. Тогда используя (7) и принятые ограничения получаем аналитическое выражение для $C_{МПК}(P_\Delta\{S_{N,n} \ge c\})$ алгоритмической сложности вычисления точного приближения распределений $P_\Delta\{S_{N,n} \ge c\}$ методом МПК

$$C_{МПК}(P_\Delta\{S_{N,n} \ge c\}) = (r+1)^N \cdot 3N + \quad (9)$$

$$+ K_h(N,n,r)\cdot(8\cdot N + n + 2\cdot \log_2 K_h(N,n,r) + 5)\cdot$$

Учитывая то, что для выполнения условия (8) r обязательно должно выбираться из условия $r = m(N,n,\Delta)$ выражение (9) должно быть преобразовано в выражение

$$C_{МПК}(P_\Delta\{S_{N,n} \ge c\}) = (m(N,n,\Delta)+1)^N \cdot 3N + \quad (10)$$

$$+ K_h(N,n,m(N,n,\Delta))\cdot(8\cdot N + n + 2\cdot \log_2 K_h(N,n,m(N,n,\Delta)) + 5)\cdot$$

Полученное выражение (10), определяет алгоритмическую сложность расчета точных приближений распределений вероятностей значений статистик методом первой кратности, конечно и может быть использовано для вычисления их величины при любых конкретных значениях параметров N и n и задаваемой пользователем метода точности Δ.

Для вычисления значений алгоритмической сложности $C_{МПК}(P_\Delta\{S_{N,n} \ge c\})$ также необходимо знать значения $m(N,n,\Delta)$, которые были рассчитаны с помощью программы для ПЭВМ [20], результаты докладывались в [21] и частично приведены в таблице 2 и на рис. 2.

Таблица 2

Значения *m* статистики максимальной частоты M_n, принимаемые с вероятностью $P\{M_n > m\} < \Delta$ для $\Delta = 10^{-5}$

		n = 5*N* объём выборки (длина текста)													
		10	**15**	**50**	**100**	**150**	**180**	**200**	**250**	**300**	**350**	**400**	**450**	**500**	**1280**
N мощность алфавита	**2**	10													
	3	10	14												
	8	9	11												
	10	8	10	18											
	26	7	8	12	17										
	32	6	7	12	16	19									
	36	6	7	11	15	18	20								
	64	6	6	9	12	14	16	18	20						
	128	5	6	8	10	11	13	14	15	16	17	18	19	19	
	256	5	5	7	8	9	10	11	12	12	13	14	14	14	22

На рис. 2 визуально показано отличие значений n в выражении (4) и значений $m(N,n,\Delta)$ в выражении (10), определяющих основное различие в значениях алгоритмической сложности расчета точных распределений и их точных приближений методом первой кратности.

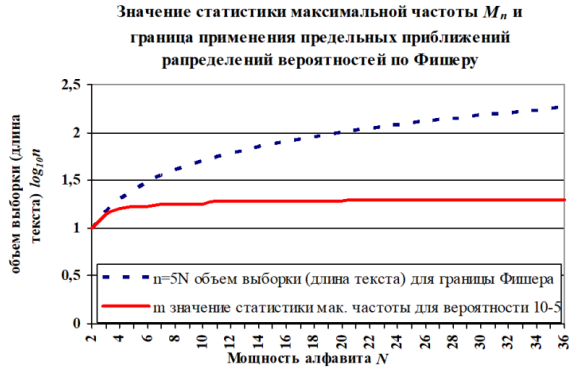


Рис. 2. График значений статистики максимальной частоты для параметров мощности алфавита и объема выборки на границе Фишера

В соответствии со значениями $m(N,n,\Delta)$ и (10) из соотношения

$$C_{МПК}(P_{\Delta}\{S_{N,n} \geq c\}) \leq 2{,}592 \cdot 10^{24}$$

были рассчитаны значения параметров N и n, для которых могут быть вычислены точные приближения распределений. Эти значения частично приведены в таблице 3, область их значений показана на рис. 3.

Таблица 3

Параметры N и n, для которых при использовании МВС производительностью 10^{18} оп/сек. за один месяц методом первой кратности могут быть рассчитаны точные приближения распределения (ВР=2,592 х 10^{24})

№ п/п	Параметры		$C_{МПК}(P_{\Delta}\{S_{N,n} \geq c\})$	$C_{МПК}(P_{\Delta}\{S_{N,n+1} \geq c\})$
	N	n		
1.	18	80	2,12470604E+24	5,62287692E+24
2.	20	41	1,14029783E+24	5,02009533E+24
3.	22	24	6,60000000E+23	5,37258146E+24
4.	24	14	3,40010387E+23	5,74318390E+24
5.	26	10	7,32223466E+23	2,35740535E+25
6.	28	7	5,15839146E+23	3,86388691E+25
7.	30	6	1,98966528E+25	2,02854063E+27
8.	32	5	2,23517418E+24	7,64031467E+26
9.	34	3	3,01050863E+22	5,93718141E+25
10.	36	3	5,10015580E+23	1,57160684E+27
11.	38	2	1,53997096E+20	8,61359646E+24
12.	40	2	1,45891986E+21	1,45071098E+26
13.	42	2	1,37867926E+22	2,43719445E+27
14.	46	2	1,22308546E+24	6,83342902E+29
15.	48	1	4,05323966E+16	1,14863678E+25
16.	50	1	1,68884986E+17	1,07684698E+26
17.	66	1	1,46098213E+22	6,11882457E+33
18.	72	1	1,02003116E+24	4,86613430E+36
19.	74	0	4,19346144E+24	4,50117423E+37

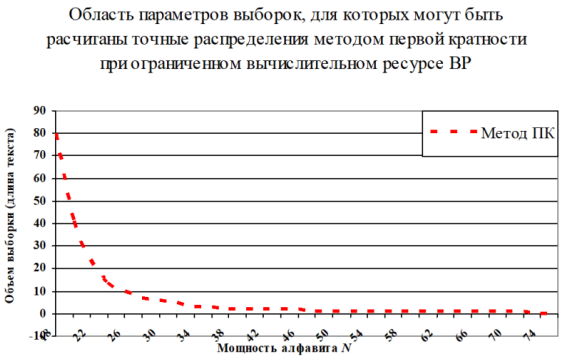


Рис. 3. Область параметров выборок, для которых методом первой кратности могут быть посчитаны точные приближения распределений

Анализ значений параметров N и n, приведенных в табл. 3 и области их изменений, показанных на рис. 3, говорит о том, для значений параметров N>46, метод первой кратности не позволяет рассчитывать точные приближения распределений вероятностей значений статистик.

Перейдем к сравнению алгоритмических сложностей расчета точных распределений и их точных приближений производимых методом первой кратности.

Оценка применения метода первой кратности для расчета точных распределений и их точных приближений. Для сравнения результатов применения метода первой кратности к расчету точных распределений и их точных приближений сравним полученные выражения для алгоритмических сложностей (4) и (10), для этого оценим их разность

$$C_{МПК}(P_T\{S_{N,n} \ge c\}) - C_{МПК}(P_\Delta\{S_{N,n} \ge c\}) =$$

$$= (n+1)^N \cdot 3N + C^n_{N+n-1} \cdot (8 \cdot N + n + 2 \cdot \log_2 C^n_{N+n-1} + 5) -$$

$$- (m(N,n,\Delta)+1)^N \cdot 3N -$$

$$- K_h(N,n,m(N,n,\Delta) \cdot (8 \cdot N + n + 2 \cdot \log_2 K_h(N,n,m(N,n,\Delta)) + 5) \cdot$$

Учитывая факт того, что $K_h(N,n,m(N,n,\Delta)) < \cdot K_h(N,n,n)$, рассматриваемая разность будет больше или равна при замене $K_h(N,n,m(N,n,\Delta))$ на $K_h(N,n,n) = C^n_{N+n-1}$. Тогда

$$C_{МПК}(P_T\{S_{N,n} \ge c\}) - C_{МПК}(P_\Delta\{S_{N,n} \ge c\}) \ge$$

$$\ge 3N \cdot ((n+1)^N - (m(N,n,\Delta)+1)^N)$$

и учитывая результаты значений $m(N,n,\Delta)$, полученных с помощью программы для ПЭВМ [10] и частично приведенных в табл. 3 видно что,

$$m(N,n,\Delta) << n$$

и

$$C_{МПК}(P_T\{S_{N,n} \ge c\}) - C_{МПК}(P_\Delta\{S_{N,n} \ge c\}) >> 0 \cdot$$

Оценка разности алгоритмических сложностей вычислений методом первой кратности точных распределений и их точных приближений показывает, что для большинства значений параметров распределений N и n сложность вычисления

точных приближений намного меньше сложности вычисления самих точных распределений. Относительная алгоритмическая сложность вычислений точных распределений и их точных приближений рассматривалась в работах [4, 11], где показано, что вычисление точных приближений во много порядков проще расчета самих точных распределений.

Для проведения анализа результатов применения метода МПК к расчету точных распределений и их точных приближений данные о максимальных значениях параметров N и n, из табл. 1 и 2 были сведены в сравнительную табл. 4.

Таблица 4

Сравнительная таблица максимальных значений параметров N и n, для которых при использовании МВС производительностью 10^{18} оп/сек. за один месяц (ВР=2,592 х 10^{24}) методом первой кратности могут быть рассчитаны точные распределения и их точные приближения

№ п/п	Мощность алфавита N	Максимальный объем выборки n для которой при ограниченном ВР методом МПК рассчитывается	
		точное приближение $P_{\Delta}\{S_{N,n} \geq c\}$- $n_{\Delta}(N)$	точное распределение $P_{T}\{S_{N,n} \geq c\}\}$ - $n_{T}(N)$
1.	18	80	19
2.	20	41	12
3.	22	24	9
4.	24	14	7
5.	26	10	6
6.	28	7	5
7.	30	6	4
8.	32	5	4
9.	34	3	3
10.	36	3	3
11.	38	2	2
12.	40	2	2
13.	42	2	2
14.	44	2	2
15.	46	2	2
16.	48	1	1
17.	72	1	1

Области расположения параметров выборок, для которых при ограниченном вычислительном ресурсе ВР методом первой кратности могут быть рассчитаны точные распределения и их точные приближения показаны на рис. 4.

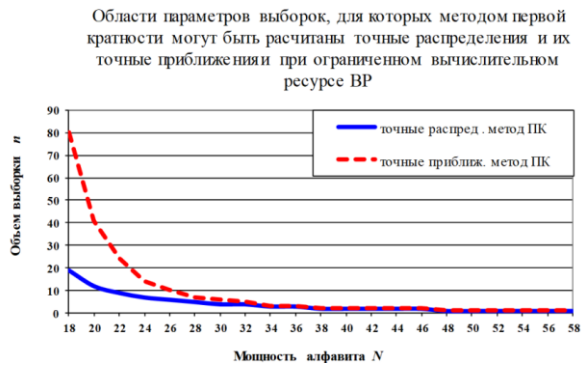


Рис. 4. Область параметров выборок, для которых методом первой кратности могут быть рассчитаны точные распределения и их точные приближения

Анализ данных таблицы 4, представленных на рис. 4 показывает, что при одинаковом выделенном вычислительном ресурсе объемом в $2{,}592 \cdot 10^{24}$ операций:

♦ методом первой кратности могут быть рассчитаны точные распределения и их точные приближения для мощности алфавита $N \leq 46$, при этом объём выборок $n \geq 2$;

♦ применение метода первой кратности не позволяет рассчитывать точные распределения и их точных приближения для значений мощности алфавита $N > 46$, при этом объем выборок $n \leq 1$;

♦ применение метода первой кратности для расчета точных приближений распределений позволяет при одинаковых значениях мощности алфавита N увеличить, по сравнению с расчетом точных распределений, объём выборок n, для которых могут быть рассчитаны точные приближения распределений;

♦ для мощности алфавита $N \leq 46$ объём выборок $n_{\Delta}(N)$, для которых могут быть рассчитаны точные приближения распределений, для каждого значения N превышает объём выборок $n_{T}(N)$, для которых могут быть рассчитаны точные распределения $n_{\Delta}(N) > n_{T}(N)$;

♦ для мощности алфавита $N \leq 24$ объём выборок $n_{\Delta}(N)$, для которых могут быть рассчитаны точные приближения распределений, в 2 и более раз превышает объём выборок $n_{T}(N)$, для которых могут быть рассчитаны точные распределения $(n_{\Delta}(N)/n_{T}(N)) \geq 2$.

Заключение и выводы. В работе приведены аналитические выражения позволяющие для любых значений параметров распределений мощности алфавита N и объёма выборки n рассчитывать алгоритмическую сложность вычисления точных распределений вероятностей значений статистик и их точных приближений методом решения уравнения первой кратности, учитывающие не только область перебора возможных решений но и число этих решений. В качестве точных приближений распределений рассматриваются Δ–точные распределения, отличающиеся от точных распределений не более чем на заданную заранее величину Δ.

Анализ аналитических выражений, полученных для алгоритмических сложностей расчета распределений, показал, что сложность вычисления точных приближений распределений на много меньше сложности вычисления самих точных распределений и позволил провести вычисление максимальных значений параметров распределений N и n, для которых при фиксированном вычислительном ресурсе, ограниченным современным уровнем развития вычислительных средств, методом решения уравнения первой кратности могут быть рассчитаны точные распределения и их точные приближения.

Анализ вычисленных максимальных значений параметров распределений показал, что методом решения уравнения первой кратности нельзя вычислить точные распределения и их точные приближения для мощностей алфавита N превышающих 46.

Сравнительный анализ вычисленных максимальных значений параметров распределений показал, что применение метода первой кратности для расчета точных приближений распределений позволяет при одинаковых значениях мощности алфавита N увеличить, по сравнению с расчетом точных распределений, объём выборок n, для которых могут быть рассчитаны точные приближения распределений, так для мощностей алфавита меньше 24 это увеличение составляет два и более раз.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Мельников А.К.* Сравнение эффективности обработки текстов при применении в статистических критериях точных и предельных приближений базовых распределений вероятностей значений тестовых статистик // Обозрение прикладной и промышленной математики. – 2018. – Т. 25. – Вып. 4. – С. 375-378. – ISSN 0869-8325. – https://tvp.ru/conferen/vsppmXIX/repso114.pdf (дата обращения: 14.01.2019).

2. *Зелюкин Н.Б., Мельников А.К.* Сложность расчета точных распределений вероятности значений статистик и область применения предельных распределений // Электронные средства и системы управления: Матер. докладов XIII Междунар. науч.-практ. конф. (29 ноября – 1 декабря 2017 г.): в 2 ч. Ч. 2. – Томск: В-Спектр, 2017. – С. 84-90. – https://storage.tusur.ru/files/115115/2017-2.pdf (дата обращения: 13.07.2018).
3. *Мельников А.К.* Сложность расчета точных распределений вероятности симметричных аддитивно разделяемых статистик и область применения предельных распределений // Доклады ТУСУР. – Томск, 2017. – Т. 20, № 4. – С. 126-130. – ISSN 1818-0442.
4. *Чеповский А.М.* Информационные модели в задачах обработки текстов на естественных языках. – М.: Национальный открытый университет «ИНТУИТ», 2015. – 228 с. – ISBN 978-5-9556-0176-2.
5. *Мельников А.К.* Анализ точных и предельных приближений распределений вероятностей значений статистик // Суперкомпьютерные технологии (СКТ-2018): Матер. 5-й Всероссийской научно-технической конференции: в 2 т. Т. 1. – Ростов-на-Дону; Таганрог: Изд-во ЮФУ, 2018. – С. 100-104. – ISBN 978-9275-2834-9.
6. *Мельников А.К.* Применение точных распределений в процедуре двухэтапной обработки текстов // Обозрение прикладной и промышленной математики. – 2018. – Т. 25. – Вып. 2. – С. 89-95. – ISSN 0869-8325. – https://tvp.ru/conferen/vsppmXIX/repso051.pdf (дата обращения: 24.01.2019).
7. *Мельников А.К., Ронжин А.Ф.* Обобщенный статистический метод анализа текстов, основанный на расчете распределений вероятности значений статистик // Информатика и её применения. – 2016. – Т. 10. – Вып. 4. – С. 91-97. – ISSN 1992-2264.
8. *Мельников А.К.* Применение точных и предельных приближений распределений вероятностей значений статистик при решении задачи обработке текстов // Известия ЮФУ. Технические науки. – 2018. – № 8 (202). – С. 114-135.
9. *Мельников А.К.* Относительная алгоритмическая сложность расчета точных приближений распределений вероятностей значений статистик // Известия ЮФУ. Технические науки. – 2019. – № 7 (209). – С. 35-45.
10. *Мельников А.К.* Направление модернизации частотного метода расчета точных распределений вероятностей значений статистик // Обозрение прикладной и промышленной математики. – 201. – Т. 24. – Вып. 5. – С. 324-325. – https://tvp.ru/conferen/vsppmXVIII/kisso073.pdf (дата обращения: 12.07.2018).
11. *Зуев Ю.А*. Современная дискретная математика: От перечислительной комбинаторики до криптографии XXI века. – М.: ЛЕНАРД, 2019. – 720 с. (Основы защиты информации № 17). – ISBN 978-5-9710-5660-7.
12. *Мельников А.К.* Методика расчета распределения вероятностей значений симметричных аддитивно разделяемых статистик, приближенных к их точному распределению // Научный вестник НГТУ. – 2018. – № 1 (70). – С. 153-166. – ISBN 1814-1196. – Doi: 10.17212/1814-1196-2018-1-153-166.
13. *Крамер Г.* Математические методы статистики. – М.: Мир, 1975. – 648 с.
14. *Кнут Д.Э.* Искусство программирования для ЭВМ. Т. 3. Сортировка и поиск. – М.: Мир, 1978. – 844 с.
15. Список TOP 500 55-й редакция от 22 июня 2020 года.
16. *Воеводин В.В., Воеводин Вл.В.* Параллельные вычисления. – СПб.: БХВ-Петербург, 2002. – 608 с. – ISBN 5-94157-160-7.
17. *Муравьёв С.А.* История отечественной электронной вычислительной техники / под общей редакцией директора Департамента радиоэлектронной промышленности Минпромторга России С.В. Хохлова. – М.: ООО "Издательский дом "Столичная энциклопедия", 2017. – 680 с. – ISBN 978-5-903989-34-8.
18. *Гузик В.Ф., Каляев И.А., Левин И.И.* Реконфигурируемые вычислительные системы / под общ. ред. И.А. Каляева. – Таганрог: Изд-во ЮФУ, 2016. – 472 с. – ISBN 978-5-9275-1918-7.
19. *Сачков В.Н.* Введение в комбинаторные методы дискретной математики. – М.: Наука. Гл. ред. физ.-мат. лит., 1982. – 384 с.

20. Свидетельство о государственной регистрации программы для ЭВМ № 2018664980. Расчет вероятностей значений статистики максимальной частоты. Правообладатель Мельников А.К. Автор: Мельников А.К. и Зелюкин Н.Б. Заявка № 2018662123. Дата поступления 01 ноября 2018 г. Дата государственной регистрации в Реестре программ для ЭВМ 27 ноября 2018 г.
21. *Мельников А.К.* Относительная алгоритмическая сложность расчета точных приближений распределений вероятностей значений статистик // Модели, методы и технологии интеллектуального управления (ИУ-2019): Матер. 12-й Мультиконференции по проблемам управления (МКПУ-2019): в 4 т. Т. 1. – Ростов-на-Дону; Таганрог: Изд-во ЮФУ, 2019.. – С. 108-112. – ISBN 978-5-9275-3189-9.

REFERENCES

1. *Mel'nikov A.K.* Sravnenie effektivnosti obrabotki tekstov pri primenenii v statisticheskikh kriteriyakh tochnykh i predel'nykh priblizheniy bazovykh raspredeleniy veroyatnostey znacheniy testovykh statistik [Comparison of the efficiency of text processing when applying exact and marginal approximations of the basic probability distributions of test statistics in statistical criteria], *Obozrenie prikladnoy i promyshlennoy matematiki* [Review of Applied and Industrial Mathematics], 2018, Vol. 25, Issue 4, pp. 375-378. ISSN 0869-8325. Available at: https://tvp.ru/conferen/vsppmXIX/repso114.pdf (accessed 14 January 2019).
2. *Zelyukin N.B., Mel'nikov A.K.* Slozhnost' rascheta tochnykh raspredeleniy veroyatnosti znacheniy statistik i oblast' primeneniya predel'nykh raspredeleniy [The complexity of calculating accurate probability distributions of statistical values and the scope of limit distributions], *Elektronnye sredstva i sistemy upravleniya: Mater. dokladov XIII Mezhdunar. nauch.-prakt. konf. (29 noyabrya – 1 dekabrya 2017 g.)* [Electronic means and control systems: Materials of reports of the XIII International Scientific and Practical Conference (November 29 – December 1, 2017)]: in 2 parts. Part 2. Tomsk: V-Spektr, 2017, pp. 84-90. Available at: https://storage.tusur.ru/files/115115/2017-2.pdf (accessed 13 July 2018).
3. *Mel'nikov A.K.* Slozhnost' rascheta tochnykh raspredeleniy veroyatnosti simmetrichnykh additivno razdelyaemykh statistik i oblast' primeneniya predel'nykh raspredeleniy [The complexity of calculating the exact probability distributions of symmetric additive-separated statistics and the application of limit distributions], *Doklady TUSUR* [Proceedings of Tomsk State University]. Tomsk, 2017, Vol. 20, No. 4, pp. 126-130. ISSN 1818-0442.
4. *Chepovskiy A.M.* Informatsionnye modeli v zadachakh obrabotki tekstov na estestvennykh yazykakh [Information models in tasks of processing of natural language texts]. Moscow: Natsional'nyy otkrytyy universitet «INTUIT», 2015, 228 p. ISBN 978-5-9556-0176-2.
5. *Mel'nikov A.K.* Analiz tochnykh i predel'nykh priblizheniy raspredeleniy veroyatnostey znacheniy statistik [Analysis of exact and marginal approximations of probability distributions of statistical values], *Superkomp'yuternye tekhnologii (SKT-2018): Mater. 5-y Vserossiyskoy nauchno-tekhnicheskoy konferentsii* [Supercomputer Technologies (SCT-2018): Proceedings of the 5th All-Russian Scientific and Technical Conference]: in 2 vol. Vol. 1. Rostov-on-Don; Taganrog: Izd-vo YuFU, 2018, pp. 100-104. ISBN 978-9275-2834-9.
6. *Mel'nikov A.K.* Primenenie tochnykh raspredeleniy v protsedure dvukhetapnoy obrabotki tekstov [Application of exact distributions in the procedure of two-step text processing], Obozrenie prikladnoy i promyshlennoy matematiki [Review of applied and industrial mathematics], 2018, Vol. 25, Issue 2, pp. 89-95. ISSN 0869-8325. Available at: https://tvp.ru/conferen/vsppmXIX/repso051.pdf (accessed 24 January 2019).
7. *Mel'nikov A.K., Ronzhin A.F.* Obobshchennyy statisticheskiy metod analiza tekstov, osnovannyy na raschete raspredeleniy veroyatnosti znacheniy statistik [Generalized statistical method of text analysis based on calculation of probability distribution of statistical values] *Informatika i ee primeneniya* [Informatics and its applications], 2016, Vol. 10, Issue 4, pp. 91-97. ISSN 1992-2264.
8. *Mel'nikov A.K.* Primenenie tochnykh i predel'nykh priblizheniy raspredeleniy veroyatnostey znacheniy statistik pri reshenii zadachi obrabotke tekstov [Application of exact and limit approximations of statistics probability distributions for the problem of text processing], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2018, No. 8 (202), pp. 114-135.
9. *Mel'nikov A.K.* Otnositel'naya algoritmicheskaya slozhnost' rascheta tochnykh priblizheniy raspredeleniy veroyatnostey znacheniy statistik [Relative algorithmic complexity of exact approximations for probability distributions of statistics values], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2019, No. 7 (209), pp. 35-45.

10. *Mel'nikov A.K.* Napravlenie modernizatsii chastotnogo metoda rascheta tochnykh raspredeleniy veroyatnostey znacheniy statistik [Direction of modernization of the frequency method of exact probability distributions for statistics values], *Obozrenie prikladnoy i promyshlennoy matematiki* [Review of applied and industrial mathematics], 2017, Vol. 24, Issue 5, pp. 324-325. Available at: https://tvp.ru/conferen/vsppmXVIII/kisso073.pdf (accessed 12 July 2018).
11. *Zuev Yu.A.* Sovremennaya diskretnaya matematika: Ot perechislitel'noy kombinatoriki do kriptografii XXI veka [Modern discrete mathematics: From enumerative combinatorics to cryptography of the XXI century]. Moscow: LENARD, 2019, 720 p. (Fundamentals of Information security No. 17). ISBN 978-5-9710-5660-7.
12. *Mel'nikov A.K.* Metodika rascheta raspredeleniya veroyatnostey znacheniy simmetrichnykh additivno razdelyaemykh statistik, priblizhennykh k ikh tochnomu raspredeleniyu [A method for calculating the probability distribution of values of symmetric additively separated statistics that are close to their exact distribution], *Nauchnyy vestnik NGTU* [Science bulle-tin of the Novosibirsk state technical university], 2018, No. 1 (70), pp. 153-166. ISBN 1814-1196. Doi: 10.17212/1814-1196-2018-1-153-166.
13. *Kramer G.* Matematicheskie metody statistiki [Mathematical methods of statistics]. Moscow: Mir, 1975, 648 p.
14. *Knut D.E.* Iskusstvo programmirovaniya dlya EVM. T. 3. Sortirovka i poisk [The art of computer programming. Vol. 3. Sorting and Searching]. Moscow: Mir, 1978, 844 p.
15. Spisok TOR 500 55-y redaktsiya ot 22 iyunya 2020 goda [TOP 500 list 55 edition of june 22, 2020].
16. *Voevodin V.V., Voevodin Vl.V.* Parallel'nye vychisleniya [Parallel computing]. St. Petersburg: BKhV-Peterburg, 2002, 608 p. ISBN 5-94157-160-7.
17. *Murav'ev S.A.* Istoriya otechestvennoy elektronnoy vychislitel'noy tekhniki [History of Russian electronic computer technology], under the General editorship of the Director of the Department of radio-electronic industry of the Ministry of industry and trade of Russia S.V. Khokhlov. Moscow: OOO "Izdatel'skiy dom "Stolichnaya entsiklopediya", 2017, 680 p. ISBN 978-5-903989-34-8.
18. *Guzik V.F., Kalyaev I.A., Levin I.I.* Rekonfiguriruemye vychislitel'nye sistemy [Reconfigurable computing systems], under the General ed. of I.A. Kalyayev. Taganrog: Izd-vo YuFU, 2016, 472 p. ISBN 978-5-9275-1918-7.
19. *Sachkov V.N.* Vvedenie v kombinatornye metody diskretnoy matematiki [Introduction to combinatorial methods of discrete mathematics]. Moscow: Nauka. Gl. red. fiz.-mat. lit., 1982, 384 p.
20. Svidetel'stvo o gosudarstvennoy registratsii programmy dlya EVM № 2018664980. Raschet veroyatnostey znacheniy statistiki maksimal'noy chastoty. Pravoobladatel' Mel'nikov A.K. Avtor: Mel'nikov A.K. i Zelyukin N.B. Zayavka № 2018662123. Data postupleniya 01 noyabrya 2018 g. Data gosudarstvennoy registratsii v Reestre programm dlya EVM 27 noyabrya 2018 g. [Certificate of state registration of computer software № 2018664980. Calculation of probabilities of maximum frequency statistics. Proprietor – A.K. Melnikov. Authors: A.K. Melnikov and N.B. Zeliukin. Application № 2018662123. Date of filing – November 01, 2018. Date of state registration in the Register of computer software – November 27, 2018].
21. *Mel'nikov A.K.* Otnositel'naya algoritmicheskaya slozhnost' rascheta tochnykh priblizheniy raspredeleniy veroyatnostey znacheniy statistik [Relative algorithmic complexity of exact approximations for probability distributions of statistics values], *Modeli, metody i tekhnologii intellektual'nogo upravleniya (IU-2019): Mater. 12-y Mul'tikonferentsii po problemam upravleniya (MKPU-2019)* [Models, methods and technologies of intelligent management (IU-2019): Materials of the 12th Multi-Conference on Management Problems (ICPU-2019)]: in 4 vol. Vol. 1. Rostov-on-Don; Taganrog: Izd-vo YuFU, 2019, pp. 108-112. ISBN 978-5-9275-3189-9.

Статью рекомендовал к опубликованию д.т.н., профессор И.И. Левин.

Мельников Андрей Кимович – АО «Вычислительные решения»; e-mail: ak@comp-sol.ru; 117587, г. Москва, Варшавское шоссе, 125, тел.: +74952870035; к.т.н.; доцент ВАК; г.н.с.

Melnikov Andrey Kimovich – SC «Computing solutions»; e-mail: ak@comp-sol.ru; 125, Varshavskoye roag, Moscow, 117587, Russia; phone: +784952870035; cand. of eng. sc.; associate professor of SAC; chief research officer.

УДК 519.224.22 DOI 10.18522/2311-3103-2020-7-68-78

А.К. Мельников

РАСЧЕТ КОЛИЧЕСТВА РЕШЕНИЙ УРАВНЕНИЯ ПЕРВОЙ КРАТНОСТИ ТИПОВ В УСЛОВИЯХ ОГРАНИЧЕНИЙ НА ЧАСТОТУ ВСТРЕЧАЕМОСТИ ЗНАКОВ АЛФАВИТА

Рассматривается количество решений уравнения первой кратности типов, составленного из векторов кратности типов, каждый элемент которого представляет собой число вхождений элементов определенного типа (какого-либо знака алфавита) в рассматриваемую выборку. Уравнение первой кратности типов связывает между собой число вхождений элементов всех типов в рассматриваемую выборку и объём этой выборки. Основное внимание в статье уделено выводу и доказательству правильности выражения, определяющего количество неотрицательных целочисленных решений уравнения первой кратности типов в условиях ограничений на частоту встречаемости знаков алфавита. Решение уравнения первой кратности типов является основой расчета точных приближений вероятностей значений статистик методом первой кратности, где в качестве точных приближений выступают Δ–точные распределения, отличающиеся от точных распределений не более чем на заранее заданную, сколь угодно малую величину Δ. Величина, выражающая количество решений уравнения первой кратности типов, является одной из величин определяющих алгоритмическую сложность метода первой кратности, без знания значения которой нельзя определить параметры выборок, для которых при ограничениях на вычислительный ресурс могут быть рассчитаны точные приближения распределений. Также величина выражающая количества решений уравнения первой кратности типов используется в методе первой кратности для ограничения области поиска решений уравнения. Количество решений уравнения первой кратности рассматривается в условиях ограничения на максимальное значение элементов вектора кратности, при этом рассматривается случай, когда один или несколько элементов алфавита могут в выборке отсутствовать. Впервые получено выражение, определяющее количество неотрицательных целочисленных решений уравнения первой кратности типов в условиях ограничений сверху на значения частот встречаемости знаков и возможности отсутствия одного или нескольких знаков алфавита в рассматриваемой выборке. Получены аналитические выражения, позволяющие для любых значений мощности алфавита, объёма выборки и ограничения на значение максимальной частоты встречаемости знаков алфавита вычислять количество целочисленных неотрицательных решений уравнения первой кратности типов. Вид полученного выражения позволяет использовать его при изучении алгоритмической сложности расчетов точных приближений распределений вероятностей значений статистик с заранее указанной точностью Δ.

Вероятность; статистика; точное распределение; точное приближение; вектор кратности типов; линейное уравнение; алгоритмическая сложность.

A.K. Melnikov

CALCULATION OF THE NUMBER OF SOLUTIONS TO THE EQUATION OF THE FIRST MULTIPLICITY OF TYPES UNDER RESTRICTIONS ON THE FREQUENCY OF OCCURRENCE OF ALPHABET CHARACTERS

The article considers the number of solutions to the equation of the first multiplicity of types, composed of vectors of multiplicity of types, each element of which is the number of occurrences of elements of a certain type (any sign of the alphabet) in the sample under consideration. The equation of the first multiplicity of types relates the number of occurrences of elements of all types in the sample under consideration and the volume of this sample. The main attention is paid to the conclusion and proof of the correctness of the expression that determines the number of non-negative integer solutions of the equation of the first multiplicity of types under conditions of restrictions on the frequency of occurrence of alphabet characters. The solution of the equation of

the first multiplicity of types is the basis for calculating exact approximations of the probabilities of statistical values by the first multiplicity method, where the exact approximations are Δ–exact distributions that differ from the exact distributions by no more than a predetermined, arbitrarily small value Δ. The value that expresses the number of solutions to the equation of the first multiplicity of types is one of the values that determine the algorithmic complexity of the method of the first multiplicity, without knowing the value of which it is impossible to determine the parameters of samples for which, under restrictions on the computational resource, exact approximations of distributions can be calculated. Also, the value expressing the number of solutions to the equation of the first multiplicity of types is used in the method of the first multiplicity to limit the search area for solutions to the equation. The number of solutions to the equation of the first multiplicity is considered under conditions of restriction on the maximum value of the elements of the multiplicity vector, and the case is considered when one or more elements of the alphabet may be missing in the sample. First obtained the expression that defines the number of nonnegative integer solutions to equations of the first multiplicity of types in terms of restrictions on the values of the frequencies of occurrence of signs and the possibility of absence of one or more characters of the alphabet in the sample reviewed. Analytical expressions are obtained that allow calculating the number of integer nonnegative solutions of the equation of the first multiplicity of types for any values of the alphabet power, the sample size, and the limit on the maximum frequency of occurrence of alphabet characters. The form of the obtained expression allows you to use it when studying the algorithmic complexity of calculating exact approximations of probability distributions of statistical values with a pre-specified accuracy Δ.

Probability; statistics; exact distribution; an accurate approximation of the vector of multiplicity of types; the linear equation algorithmic complexity.

Введение. Для расчета точных распределений [1] и их точных приближений (Δ-точных распределений [2]) может применяться метод, основанный на решении уравнения первой кратности типов [3], объединяющего информацию о частотах встречаемости знаков алфавита мощности N в последовательности (выборки) длины (объёма) n. Для оценки алгоритмической сложности [4] метода расчета точных приближений, основанного на решении уравнения первой кратности, необходимо уметь рассчитывать количество решений этого уравнения.

Данная работа посвящена расчету количества решений уравнения первой кратности типов.

Постановка задачи. Необходимо рассчитать $K_h(N, n, r)$ количество целочисленных упорядоченных неотрицательных решений уравнения первой кратности типов

$$h_1 + h_2 + ... + h_N = n, \quad (1)$$

где N мощность алфавита $A_N = \{a_1, a_2, ... , a_N\}$, n длина последовательности (объёма выборки) и h_i частота встречаемости a_i знака алфавита A_N. Условия расчета следующие:

1. N, n, и $m \in$ N − множеству натуральных чисел (целых положительных чисел больших нуля),

2. N мощность алфавита $A_N = \{a_1, a_2, ... , a_N\}$,

3. n длина последовательности или объём выборки, $N \le n$,

4. $\{0 \le h_i \le n \mid \forall i = \overline{1.N}\}$ вектор первой кратности типов (вектор частот встречаемости знаков алфавита в последовательности),

5. r ограничение на значения координат (частот) вектора первой кратности типов $0 \le h_i \le r \mid \forall i = \overline{1.N}$, $r \le n$;

Обозначим условия ограничения на координаты вектора кратности типов через

$$\{0 \le h_i \le r \mid \forall i = \overline{1, N}\}. \quad (2)$$

Условие упорядоченности решений, означает, что каждая переменная вектора решений $\{h_1^{(i)}, h_2^{(i)}, ..., h_N^{(i)}\}$ уравнения (1) пронумерована.

Условия не отрицательности и упорядоченности решений уравнения (1) можно продемонстрировать двумя вырожденными решениями, которые считаются различными.

$$\underbrace{0+0+...+0}_{N-1}+n=n \qquad n+\underbrace{0+0+...+0}_{N-1}=n.$$

Известно, что в частном случае, при $r=n$ число $K_h(N, n, n)$ целочисленных неотрицательных упорядоченных решений уравнения (1) в условиях

$$0 \le h_i \le n \mid \forall i = \overline{1, N}$$

равно числу сочетаний с повторениями $\overline{C}_N^n$

$$K_h(N,n,n) = \overline{C}_N^n = C_{N+n-1}^n = \binom{N+n-1}{n} = \frac{(N+n-1)!}{n!\cdot(N-1)!}\cdot$$

Данная проблематика подробно рассматривалась Боровков А.А. [5], Колчиным В.Ф. [6], Сачковым В.Н. [7] и Зуевым Ю.А. [8]. Получение количества решений уравнения в общем случае рассмотрим с помощью двух методов: метода взаимно однозначного соответствия и метода включения-исключения.

Метод взаимно однозначного соответствия. Для рассмотрения общего случая для $0 \le r \le n$ несколько преобразуем уравнение (1) путем прибавления единицы к каждой неизвестной h_i левой части уравнения и N к правой части уравнения (1). Получим

$$(h_1 + 1) + (h_2 + 1) + ... + (h_N + 1) = n + N \cdot \qquad (3)$$

Условие (2) преобразуем путем прибавления 1 (единиц) к неизвестной h_i и значениям нижней и верхней границы. Получим

$$1 \le (h_i + 1) \le r + 1 \mid \forall i = \overline{1, N} \cdot \qquad (4)$$

Теперь установим взаимно однозначного соответствия между числом целочисленных положительных решений уравнения (1) в условии (2) и числом целочисленных положительных решений уравнения (3) в условии (4) описанным у многих авторов методом, в частности у известного американского математика Marshall Hall (М. Холл) [10] на стр. 11 при рассмотрении вопроса о числе сочетаний с повторениями, и на портале [11].

Обязательно отметим очевидный факт, что ввиду проведения тождественных преобразований и установления взаимно однозначного соответствия между {(1), (2)} и {(3), (4)} число решений уравнения (1) в условиях (2) равно числу решений уравнения (3) в условиях (4).

Более общий случай для числа решений уравнения с ограничением на значения всех неизвестных рассматривается Сачковым В.Н. [7] при рассмотрении в параграфе 3 коммутативного несимметричного n-базиса, являющегося частным случаем обобщенной комбинаторной схемы, приведен результат (3.23 стр. 215), определяющий число решений $K_{nm}(s)$ (число композиций) уравнения

$$x_1+x_2+...+x_n=m, \qquad (5)$$

где под n подразумевается мощность алфавита, под m длина последовательности и под s ограничение на значения неизвестных (координаты вектора первой кратности)

$$1 \le x_i \le s \mid \forall i = \overline{1,n}. \tag{6}$$

Этот результат следующий

$$K_{nm}(s) = \sum_{k=0}^{\min(n,\frac{m-n}{s})} (-1)^k \binom{n}{k} \binom{m-sk-1}{n-1}. \tag{7}$$

Здесь и далее под максимальным значением для индекса суммирования k понимается целое неотрицательное число меньшее или равное $\min(n, \frac{m-n}{s})$

$$\max k \le \min(n, \frac{m-n}{s}).$$

Отличие условия (2) от условия (6) в том, что мы рассматриваем модель когда некоторые знаки алфавита $A_N = \{a_1, a_2, \dots, a_N\}$ могут не реализоваться в рассматриваемой нами последовательности и поэтому некоторые $h_i = 0$ могут равняться 0. В условиях (6) рассматривается более узкий случай, когда все знаки алфавита реализованы в рассматриваемой последовательности и поэтому $x_i \ne 0 \mid \forall i = \overline{1,n}$, где здесь n мощность алфавита.

Теперь определим взаимно однозначное соответствие между уравнениями (3) и (5):

$$(h_1 + 1) + (h_2 + 1) + \dots + (h_N + 1) = n + N$$

$$x_1 + x_2 + \dots + x_n = m.$$

Взаимно однозначное соответствие может быть определено при

$$n=N,\ x_i = h_i + 1 \mid \forall i = \overline{1,N} \text{ и } m=n+N. \tag{8}$$

Взаимно однозначное соответствие между условиями (4) и (6):

$$1 \le (h_i + 1) \le r + 1 \mid \forall i = \overline{1,N}$$

$$1 \le x_i \le s \mid \forall i = \overline{1,n}$$

определяется, учитывая что $n=N$ при

$$x_i = h_i + 1 \mid \forall i = \overline{1,N} \text{ и } s=r+1. \tag{9}$$

Теперь при применении взаимно однозначного соответствия (8) и (9) выражение (7) принимает вид

$$K_h(N,n,r) = \sum_{i=0}^{\min(N,\frac{n}{r+1})} (-1)^i \binom{N}{i} \binom{n+N-(r+1)i-1}{N-1} =$$

$$= \sum_{i=0}^{\min(N,\frac{n}{r+1})} (-1)^i \binom{N}{i} \binom{N+n-1-(r+1)i}{N-1}. \tag{10}$$

В случаи $r=n$, исходя из (10) $K_h(N,n,r)$ будет равняться $K_h(N,n,n)$

$$K_h(N,n,n)=\sum_{i=0}^{\min\left(N,\frac{n}{n+1}\right)}(-1)^i\binom{N}{i}\binom{n+N-(n+1)i-1}{N-1}=\sum_{i=0}^{0}(-1)^0\binom{N}{0}\binom{n+N-1}{N-1}=$$

$$=1\cdot1\cdot\binom{N+n-1}{N-1}=\binom{N+n-1}{n}=\overline{C}_N^n=C_{N+n-1}^n, \quad (11)$$

что полностью согласуется с приведенным ранее результатом. Таким образом изложение метода нахождения числа решений линейного уравнения, основанного на построении взаимно однозначных соответствий, завершено.

Метод включения-исключения. Для подтверждения правильности полученного решения о количестве упорядоченных неотрицательных целочисленных решений уравнения первой кратности типов в условиях (2), кратко, приведем пример другого решения данной задачи, предложенного Колодзеем А.В. и основанного на применения общеизвестного метода включения-исключения, рассматриваемого, например, в [12, 13].

Обозначим через Ω – множество всех упорядоченных целочисленных неотрицательных решений уравнения (1) в условиях (2) без учёта ограничений $0\le h_i\ \le n\mid\forall i=\overline{1,N}$. Тогда, как указывалось выше, число таких решений $|\Omega|$ равно числу сочетаний с повторениями *n* по *N* – $\overline{C}_N^n$.

$$|\Omega|=K_h(N,n,n)=\overline{C}_N^n=\binom{N+n-1}{N-1}=\binom{N+n-1}{n}=C_{N+n-1}^n.$$

Обозначим через Ω_j множество всех упорядоченных целочисленных неотрицательных решений уравнения (1) при условии $h_j>r.$ Ясно, что число решений уравнения (1) в условиях $\{0\le h_i\ \le r\mid\forall i=\overline{1,N}\}$ равно

$$K_h(N,n,r)=|\Omega|-\left|\bigcup_{j=1}^{N}\Omega_j\right|. \quad (12)$$

Для вычисления $\left|\bigcup_{j=1}^{N}\Omega_j\right|$ применим метод включения – исключения, см., например [11].

$$\left|\bigcup_{j=1}^{N}\Omega_j\right|=\sum_{i-1}^{N}|\Omega_i|-\sum_{1\le i_1<i_2\le N}\left|\Omega_{i_1}\bigcap\Omega_{i_2}\right|+\ldots+ \quad (13)$$

$$+(-1)^{k-1}\cdot\sum_{1\le i_1<\ldots<i_k\le N}\left|\Omega_{i_1}\bigcap\Omega_{i_2}\bigcap\ldots\bigcap\Omega_{i_k}\right|+\ldots$$

$$+(-1)^{n-1}\cdot\left|\Omega_1\bigcap\Omega_2\bigcap\ldots\bigcap\Omega_N\right|\cdot$$

В каждой *k*–й сумме

$$\sum_{1\le i_1<\ldots<i_k\le N}\left|\Omega_{i_1}\bigcap\Omega_{i_2}\bigcap\ldots\bigcap\Omega_{i_k}\right|$$

из (13) число слагаемых равно биномиальному коэффициенту C_N^k [14].

Рассмотрим отдельно каждое *k*-е слагаемое $k = \overline{2,N}$ из (13). Оно, по построению, есть число упорядоченных целочисленных неотрицательных решений уравнения (1)

$$h_1 + h_2 + ... + h_N = n$$

при условии, что *k* переменных $h_{i_j} > r \mid j = \overline{1,k}$.

Теперь для этого *k*–ого слагаемого из (13) и представляющего его уравнения (1) вычтем из каждой переменной $h_{i_j} > r \mid j = \overline{1,k}$, число (*r*+1), тогда уравнение (1) примет вид

$$h_1 + (h_{j_1} - r - 1) + ... + (h_{j_2} - r - 1) + ... + (h_{j_k} - r - 1) + ... + h_N = \quad (14)$$

$$= n - k(r+1).$$

Теперь преобразуем уравнение (14) следующим образом:

♦ переменным h_i, не подвергшимся уменьшению, присвоим значения переменных y_i,

♦ переменным $(h_{j_\nu} - r - 1)$, подвергшимся уменьшению, также присвоим значения переменных y_{j_ν}.

Таким образом переопределив переменные уравнения (14) получаем эквивалентное уравнение

$$y_1 + y_2 + ... + y_N = n - k(r+1), \quad (15)$$

число упорядоченных целочисленных неотрицательных решений которого равно числу целочисленных решений уравнения (14).

Как известно из [8, 9] число упорядоченных целочисленных неотрицательных решений уравнения $x_1+x_2+...+x_N=m$, равно

$$\binom{N+m-1}{N-1},$$

а тогда число решений уравнения (15) при $m=n-k(r+1)$ равно

$$\binom{N+n-k(r+1)-1}{N-1}.$$

Теперь в соответствии с (12), (13) и (14) и числу слагаемых в каждой *k*–й сумме искомое количество решений $K_h(N,n,r)$ равняется

$$K_h(N,n,r) = |\Omega| - \left| \bigcup_{j=1}^{N} \Omega_j \right| = \binom{N+n-1}{N-1} - \sum_{k=1}^{N} (-1)^{k-1} \binom{N}{k} \binom{N+n-k(r+1)-1}{N-1}$$

$$= \sum_{k=0}^{N} (-1)^k \binom{N}{k} \binom{N+n-k(r+1)-1}{N-1}. \quad (16)$$

Выражение (16), полученное в результате применения метода включения – исключения, соответствует выражению (10), независимо полученному методом взаимно однозначного соответствия. Данный факт может служить дополнительным подтверждением правильности значения (10).

Анализ количества решений уравнения. Для параметров выборок на границе Фишера Р.А. [15] n=5N и соответствующих им ограничениям на значения статистики максимальной частоты $r = m(N,5N,10^{-5})$ для точности Δ=10^{-5}, рассчитанным в соответствии с [16, 17] с помощью [18] для мощностей алфавита N от 2 до 27 были проведены расчеты количества решений уравнения первой кратности без ограничений $K_h(N,n,n)$ на значения частоты и с ограничением на это значение $K_h(N,n,r)$. Результаты частично представлены в табл. 1.

Таблица 1

Количество решений уравнения первой кратности типов с ограничениями и без ограничений для параметров на границе Фишера n=5N

№ п/п	Параметры			$K_h(N,n,n)$	$K_h(N,n,r)$	$K_h(N,n,n)-K_h(N,n,r)$
	N	n	r			
1	*2*	*2*	*4*	*5*	*6*	*7*
	2	10	10	1,10 x 10^{1}	1,10 x 10^{1}	0
1	*2*	*2*	*4*	*5*	*6*	*7*
	3	15	14	1,36 x 10^{2}	1,33 x 10^{2}	3,00 x 10^{0}
	5	25	17	2,38 x 10^{4}	2,21 x 10^{4}	1,65 x 10^{3}
	6	30	17	3,25 x 10^{5}	2,88 x 10^{5}	3,71 x 10^{4}
	9	45	18	8,86 x 10^{8}	7,23 x 10^{8}	1,63 x 10^{8}
	10	50	18	1,26 x 10^{10}	9,84 x 10^{9}	2,72 x 10^{9}
	11	55	19	1,79 x 10^{11}	1,44 x 10^{11}	3,49 x 10^{10}
	16	80	19	1,10 x 10^{17}	7,53 x 10^{16}	3,51 x 10^{16}
	17	85	19	1,60 x 10^{18}	1,06 x 10^{18}	5,44 x 10^{17}
	18	90	19	2,32 x 10^{19}	1,48 x 10^{19}	8,39 x 10^{18}
	19	95	19	3,37 x 10^{20}	2,08 x 10^{20}	1,29 x 10^{20}
	20	100	20	4,91 x 10^{21}	3,27 x 10^{21}	1,64 x 10^{21}
	21	105	20	7,16 x 10^{22}	4,64 x 10^{22}	2,51 x 10^{22}
	22	110	20	1,04 x 10^{24}	6,60 x 10^{23}	3,84 x 10^{23}
	27	135	20	7,00 x 10^{29}	3,86 x 10^{29}	3,13 x 10^{29}

Разница в количестве решений с ограничениями и без представлена на рис. 1. На рис. 2 представлено процентное отношение числа решений с ограничениями к общему числу решений.

Рис. 1. Разность общего число решений уравнения первой кратности и числа его решений с ограничениями

Анализ значений в колонках 5, 6 и 7 табл. 1 и рис. 1 и 2 позволяет сделать следующие основные выводы о количестве решений уравнения первой кратности типов.

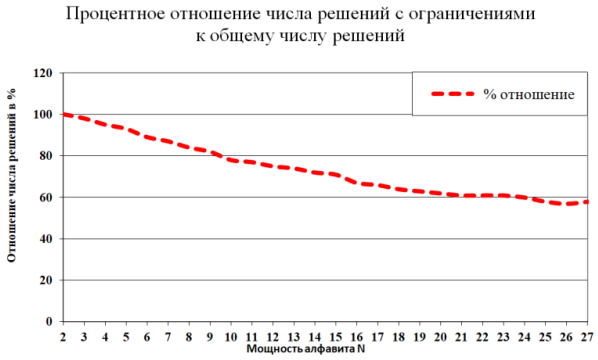


Рис. 2. Процентное отношение числа решений с ограничениями к общему числу решений уравнения первой кратности

Общее число решений уравнения практически всегда больше числа решений с ограничениями.

Разность в количестве решений увеличивается с ростом мощности алфавита N и объёма выборки n.

Для ограничения r на максимальную частоту встречаемости знаков алфавита, вычисляемого из точности $\Delta=10^{-5}$, процентное отношение числа решений с ограничениями к общему числу решений уравнения падает с ростом мощности алфавита N и объёма выборки n.

Как показывалось в [19] для оценки алгоритмической сложности расчета точных приближений распределений вероятностей значений статистик методом первой кратности необходимо, для всех значений параметров выборок N и n, уметь оценивать, как общее число перебираемых векторов возможных решений рассматриваемого нами уравнения первой кратности, так и само количество получаемых решений. Вид полученного выражения (10) аналитически прост, не содержит предельных и остаточных членов, как в предельных разложениях [20], и может быть вычислен для любых параметров выборок как с ограничениями на значения частоты так и без ограничений.

Заключение и выводы. Получены аналитические выражения, позволяющие для любых значений мощности алфавита N, объёма выборки n и ограничения на значение максимальной частоты встречаемости знаков алфавита r вычислять количество целочисленных неотрицательных решений уравнения первой кратности типов. Впервые решения получены для самых минимальных условий на выборку, учитывающих возможность отсутствия в выборке одного либо нескольких знаков из рассматриваемого алфавита. Достоверность полученных выражений подтверждается его логическим доказательством и применением при его выводе известных методов.

Вид полученного выражения позволяет использовать его при изучении алгоритмической сложности расчетов точных приближений распределений вероятностей значений статистик с заранее указанной точностью Δ. В частности полученной выражение используется для получения выражения алгоритмической сложности расчета точных приближений вероятностей распределений значений статистик методом первой кратности.

Анализ сравнения вычисленных значений количества решений уравнения первой кратности с ограничениями на частоту и без ограничений показывает их значительную разницу, увеличивающуюся с ростом мощности алфавита и объема выборки.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Мельников А.К.* Сложность расчета точных распределений вероятности симметричных аддитивно разделяемых статистик и область применения предельных распределений // Доклады ТУСУР. – Томск, 2017. – Т. 20, № 4. – С. 126-130. – ISSN 1818-0442.
2. *Мельников А.К., Ронжин А.Ф.* Обобщенный статистический метод анализа текстов, основанный на расчете распределений вероятности значений статистик // Информатика и её применения. – 2016. – Т. 10. – Вып. 4. – С. 91-97. – ISSN 1992-2264.
3. *Melnikov A.K.* Application of the calculation method of near-exact statistics probability distributions // Обозрение прикладной и промышленной математики. – 2018. – Т. 25. – Вып. 2. – С. 153-154. – ISSN 0869-8325. – https://tvp.ru/conferen/vsppmXIX/repso050.pdf (дата обращения: 24.01.2019).
4. *Юдин Д.Б., Юдин А.Д.* Математики измеряют сложность. – М.: Книжный дом "ЛИБРОКОМ", 2018. – 192 с. – ISBN 978-5-397-06042-4.
5. *Боровков А.А.* Математическая статистика. – Новосибирск: Изд-во ИМ СОРАН, Наука, 1997. – 772 с.
6. *Колчин В.Ф., Севастьянов Б.А., Чистяков В.П.* Случайные размещения. – М.: Наука, Главная редакция физико-математической литературы, 1976. – 224 с.
7. *Сачков В.Н.* Введение в комбинаторные методы дискретной математики. – М.: Наука. Гл. ред. физ.-мат. лит., 1982. – 384 стр.
8. *Зуев Ю.А*. Современная дискретная математика: От перечислительной комбинаторики до криптографии XXI века. – М.: ЛЕНАРД, 2019. – 720 с. (Основы защиты информации № 17). – IBSN 978-5-9710-5660-7.
9. *Зуев Ю.А.* Современная дискретная математика: От перечислительной комбинаторики до криптографии XXI века. Более 700 задач с решениями. – М.: ЛЕНАРД, 2019. – 304 с. (Основы защиты информации № 18). – ISBN 978-5-9710-5662-1.
10. *Холл М.* Комбинаторика. – М.: Мир, 1970. – 424 с.
11. http://www.problems.ru/view_problem_details_new.php?id=30719.
12. *Эндрюс Г.* Теория разбиений: пер. с англ. – М.: Наука. Гл. ред. физ.-мат. лит., 1982. – 256 с.
13. *Дж. Риордан*. Введение в комбинаторный анализ. – М.: Изд-во иностранной лит., 1963. – 288 с.
14. *Дж. Риордан*. Комбинаторные тождества. – М.: Наука. Гл. ред. физ.-мат. лит., 1982. – 255 с.
15. *Фишер Р.А.* Статистические методы для исследователей: пер. с англ. – М.: Госстатиздат, 1958. – 73 с.
16. *Мельников А.К.* Относительная алгоритмическая сложность расчета точных приближений распределений вероятностей значений статистик // Модели, методы и технологии интеллектуального управления (ИУ-2019): Матер. 12-й Мультиконференции по проблемам управления (МКПУ-2019): в 4 т. Т. 1. – Ростов-на-Дону; Таганрог: Изд-во ЮФУ, 2019. – С. 108-112. – ISBN 978-5-9275-3189-9.
17. *Мельников А.К.* Относительная алгоритмическая сложность расчета точных приближений распределений вероятностей значений статистик // Известия ЮФУ. Технические науки. – 2019. – № 7 (209). – С. 35-45.
18. Свидетельство о государственной регистрации программы для ЭВМ № 2018664980. Расчет вероятностей значений статистики максимальной частоты. Правообладатель Мельников А.К. Автор: Мельников А.К. и Зелюкин Н.Б. Заявка № 2018662123. Дата поступления 01 ноября 2018 г. Дата государственной регистрации в Реестре программ для ЭВМ 27 ноября 2018 г.
19. *Мельников А.К.* Методика расчета распределения вероятностей значений симметричных аддитивно разделяемых статистик, приближенных к их точному распределению // Научный вестник НГТУ. – 2018. – № 1 (70). – С. 153-166. – ISBN 1814-1196. – Doi: 10.17212/1814-1196-2018-1-153-166.
20. *Hutchinson T.P*. 1979. The validity of the chi-squared test when expected frequencis are small: A list of recent research references // Commun. Stat. A Theor. – Vol. 8, No. 4. – P. 327-335.

REFERENCES

1. *Mel'nikov A.K.* Slozhnost' rascheta tochnykh raspredeleniy veroyatnosti simmetrichnykh additivno razdelyaemykh statistik i oblast' primeneniya predel'nykh raspredeleniy [The complexity of calculating the exact probability distributions of symmetric additive-separated statistics and the application of limit distributions], *Doklady TUSUR* [Proceedings of Tomsk State University]. Tomsk, 2017, Vol. 20, No. 4, pp. 126-130. ISSN 1818-0442.
2. *Mel'nikov A.K., Ronzhin A.F.* Obobshchennyy statisticheskiy metod analiza tekstov, osnovannyy na raschete raspredeleniy veroyatnosti znacheniy statistik [Generalized statistical method of text analysis based on calculation of probability distribution of statistical values] *Informatika i ee primeneniya* [Informatics and its applications], 2016, Vol. 10, Issue 4, pp. 91-97. ISSN 1992-2264.
3. *Melnikov A.K.* Application of the calculation method of near-exact statistics probability distributions, Obozrenie prikladnoy i promyshlennoy matematiki [Review of Applied and Industrial Mathematics], 2018, Vol. 25, Issue 2, pp. 153-154. ISSN 0869-8325. Available at: https://tvp.ru/conferen/vsppmXIX/repso050.pdf (accessed 24 January 2019).
4. *Yudin D.B., Yudin A.D.* Matematiki izmeryayut slozhnost' [Mathematicians measure complexity]. Moscow: Knizhnyy dom "LIBROKOM", 2018, 192 p. ISBN 978-5-397-06042-4.
5. *Borovkov A.A.* Matematicheskaya statistika [Mathematical statistics]. Novosibirsk: Izd-vo IM SORAN, Nauka, 1997, 772 p.
6. *Kolchin V.F., Sevast'yanov B.A., Chistyakov V.P.* Sluchaynye razmeshcheniya [Random placements]. M.: Nauka, Glavnaya redaktsiya fiziko-matematicheskoy literatury, 1976, 224 p.
7. *Sachkov V.N.* Vvedenie v kombinatornye metody diskretnoy matematiki [Introduction to combinatorial methods of discrete mathematics]. Moscow: Nauka. Gl. red. fiz.-mat. lit., 1982, 384 p.
8. *Zuev Yu.A.* Sovremennaya diskretnaya matematika: Ot perechislitel'noy kombinatoriki do kriptografii XXI veka [Modern discrete mathematics: From enumerative combinatorics to cryptography of the XXI century]. Moscow: LENARD, 2019, 720 p. (Fundamentals of Information security No. 17). ISBN 978-5-9710-5660-7.
9. *Zuev Yu.A.* Sovremennaya diskretnaya matematika: Ot perechislitel'noy kombinatoriki do kriptografii XXI veka. Bolee 700 zadach s resheniyami [Sovremennaya diskretnaya matematika: Ot perechislitel'noy kombinatoriki do kriptografii XXI veka]. Moscow: LENARD, 2019, 304 p. (Osnovy zashchity informatsii No. 18). ISBN 978-5-9710-5662-1.
10. *Kholl M.* Kombinatorika [Combinatorial theory]. Moscow: Mir, 1970, 424 p.
11. http://www.problems.ru/view_problem_details_new.php?id=30719.
12. *Endryus G.* Teoriya razbieniy [The theory of partitions]: transl. from engl. Moscow: Nauka. Gl. red. fiz.-mat. lit., 1982, 256 p.
13. *Dzh. Riordan.* Vvedenie v kombinatornyy analiz [Introduction to combinatorial analysis]. Moscow.: Izd-vo inostrannoy lit., 1963, 288 p.
14. *Dzh. Riordan.* Kombinatornye tozhdestva [Combinatorial identities]. Moscow: Nauka. Gl. red. fiz.-mat. lit., 1982, 255 p.
15. *Fisher R.A.* Statisticheskie metody dlya issledovateley [Statistical methods for research workers]: transl. from engl. Moscow: Gosstatizdat, 1958, 73 p.
16. *Mel'nikov A.K.* Otnositel'naya algoritmicheskaya slozhnost' rascheta tochnykh priblizheniy raspredeleniy veroyatnostey znacheniy statistik [Relative algorithmic complexity of exact approximations for probability distributions of statistics values], *Modeli, metody i tekhnologii intellektual'nogo upravleniya (IU-2019): Mater. 12-y Mul'tikonferentsii po problemam upravleniya (MKPU-2019)* [Models, methods and technologies of intelligent management (IU-2019): Materials of the 12th Multi-Conference on Management Problems (ICPU-2019)]: i n 4 vol. Vol. 1. Rostov-on-Don; Taganrog: Izd-vo YuFU, 2019, pp. 108-112. ISBN 978-5-9275-3189-9.
17. *Mel'nikov A.K.* Otnositel'naya algoritmicheskaya slozhnost' rascheta tochnykh priblizheniy raspredeleniy veroyatnostey znacheniy statistik [Relative algorithmic complexity of exact approximations for probability distributions of statistics values], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2019, No. 7 (209), pp. 35-45.
18. Svidetel'stvo o gosudarstvennoy registratsii programmy dlya EVM № 2018664980. Raschet veroyatnostey znacheniy statistiki maksimal'noy chastoty. Pravoobladatel' Mel'nikov A.K. Avtor: Mel'nikov A.K. i Zelyukin N.B. Zayavka № 2018662123. Data postupleniya 01 noyabrya 2018 g. Data gosudarstvennoy registratsii v Reestre programm dlya EVM 27

noyabrya 2018 g. [Certificate of state registration of computer software № 2018664980. Calculation of probabilities of maximum frequency statistics. Proprietor – A.K. Melnikov. Authors: A.K. Melnikov and N.B. Zeliukin. Application № 2018662123. Date of filing – November 01, 2018. Date of state registration in the Register of computer software – November 27, 2018].
19. *Mel'nikov A.K.* Metodika rascheta raspredeleniya veroyatnostey znacheniy simmetrichnykh additivno razdelyaemykh statistik, priblizhennykh k ikh tochnomu raspredeleniyu [A method for calculating the probability distribution of values of symmetric additively separated statistics that are close to their exact distribution], *Nauchnyy vestnik NGTU* [Science bulle-tin of the Novosibirsk state technical university], 2018, No. 1 (70), pp. 153-166. ISBN 1814-1196. Doi: 10.17212/1814-1196-2018-1-153-166.
20. *Hutchinson T.P.* 1979. The validity of the chi-squared test when expected frequencis are small: A list of recent research references, *Commun. Stat. A Theor*., Vol. 8, No. 4, pp. 327-335.

Статью рекомендовал к опубликованию д.т.н., профессор И.И. Левин.

Мельников Андрей Кимович – АО «Вычислительные решения»; e-mail: ak@comp-sol.ru; 117587, г. Москва, Варшавское шоссе, 125, тел.: +74952870035; к.т.н.; доцент ВАК; г.н.с.

Melnikov Andrey Kimovich – SC «Computing solutions»; e-mail: ak@comp-sol.ru; 125, Varshavskoye roag, Moscow, 117587, Russia; phone: +784952870035; cand. of eng. sc.; associate professor of SAC; chief research officer.

УДК 519.178 DOI 10.18522/2311-3103-2020-7-78-93

Д.В. Михайлов

ПРЕОБРАЗОВАНИЕ НЕКОТОРЫХ ВИДОВ ПОСЛЕДОВАТЕЛЬНЫХ ИНФОРМАЦИОННЫХ ГРАФОВ В ПАРАЛЛЕЛЬНО-КОНВЕЙЕРНУЮ ФОРМУ

Многие задачи цифровой обработки сигналов могут быть представлены в виде информационных графов. Реконфигурируемые вычислительные системы, построенные на основе ПЛИС, могут иметь структуру, непосредственно соответствующую информационному графу решаемой задачи. Построение графа задачи и последующее создание вычислительной структуры может занимать значительное время при выполнении их вручную. В связи с этим возникает необходимость создания алгоритмов преобразования информационных графов, которые могут выполняться автоматически. В статье предложены алгоритмы преобразования однородных графов, содержащих ассоциативные операции, и смешанных графов, содержащих два типа операций, один из которых является дистрибутивным по отношению к другому. Преобразования графов первого типа (состоящих из операций одного типа) сводятся к переходу от последовательной формы графа к пирамидальной для ускорения выполнения всех операций графа. В случае если имеющегося количества оборудования недостаточно для реализации всех операций графа, применяется преобразование, разбивающее исходный граф на изоморфные подграфы. Размер подграфа зависит от имеющегося вычислительного ресурса. В этом случае вычислительная структура будет соответствовать такому подграфу. Преобразования графов второго типа (состоящих из операций двух типов, одни из которых являются дистрибутивными по отношению к другим) сводятся к разделению графа на подграфы, содержащие операции одного типа, соединённые особым образом. После этого эти подграфы могут быть преобразованы в пирамидальную форму для ускорения выполнения всех операций графа. При этом количество вершин с дистрибутивными операциями может значительно возрасти, в связи с чем может потребоваться сокращение их числа. Отсюда следует, что при преобразовании графов второго типа не обходимо выбирать конкретную форму, к которой будет приведён граф, исходя из соотношения его размера и имеющегося вычислительного ресурса. Таким образом, предложенные алгоритмы преобразования информационных графов различных типов могут быть эффективно использованы при разработке вычислительных структур, основанных на ПЛИС.

Графы; реконфигурируемые вычислительные системы; преобразование графов.

D.V. Mikhailov

CONVERTING SOME TYPES OF SEQUENTIAL INFORMATION GRAPHS INTO PARALLEL-PIPELINE FORM

Many digital signal processing tasks can be represented in the form of information graphs. Reconfigurable computing systems based on FPGAs can have a structure that directly corresponds to the information graph of the problem being solved. The construction of the task graph and the subsequent creation of the computational structure can take a significant amount of time when performed manually. In this regard, it becomes necessary to create algorithms for transforming information graphs that can be performed automatically. The article proposes algorithms for transforming homogeneous graphs containing associative operations and mixed graphs containing two types of operations, one of which is distributive with respect to the other. Transformations of graphs of the first type (consisting of operations of the same type) are reduced to the transition from a sequential form of a graph to a pyramidal form to speed up the execution of all graph operations. If the available amount of equipment is not enough to implement all operations of the graph, a transformation is applied that splits the original graph into isomorphic subgraphs. The size of the subgraph depends on the available computing resources. In this case, the computational structure will correspond to such a subgraph. Transformations of graphs of the second type (consisting of operations of two types, some of which are distributive with respect to others) are reduced to dividing the graph into subgraphs containing operations of the same type, connected in a special way. After that, these subgraphs can be converted into a pyramid shape to speed up the execution of all graph operations. In this case, the number of vertices with distributive operations can increase significantly, and therefore it may be necessary to reduce their number. It follows that when transforming graphs of the second type, it is necessary to choose a specific form to which the graph will be reduced, based on the ratio of its size and the available computing resource. Thus, the proposed algorithms for transforming information graphs of various types can be effectively used in the development of computational structures based on FPGAs.

Graphs; reconfigurable computing systems; graph transformation.

Введение. Множество задач из области цифровой обработки сигналов может быть представлено в виде информационного графа. При этом важной задачей является преобразование этих графов таким образом, чтобы повысить быстродействие вычислительных структур, построенных на их основе. Примером таких структур могут служить реконфигурируемые вычислительные системы на основе программируемых логических интегральных схем [1], которые выгодно отличаются от систем, построенных на универсальных процессорах именно тем, что имеют возможность перестраивать структуру в соответствии с графом решаемой задачи. В работе [2] освещён вопрос преобразования линейных однородных информационных графов, содержащих ассоциативные операции, однако преобразования, целью которых является ускорение вычислений в соответствующих данным графам вычислительных структурах, могут быть применены не только к графам такого вида. При этом ассоциативность является важным условием этого [3, 4].

Постановка задачи. Рассмотрим последовательный информационный граф (рис. 1). В общем случае время, которое необходимо будет затратить на полное прохождение графа (последовательное выполнение всех его операций) будет рассчитываться по формуле:

$$T_G = \left(\sum_{n=1}^{N-1} l_{gn} + \sum_{k=0}^{N-2} L_{lk}\right) \cdot \tau,$$

где l_{gn} – латентности вершины g_n, $n \in [1, N]$, L_{lk} – латентность дуги l_k, $k \in [0, N-1]$, τ – длительность одного такта.

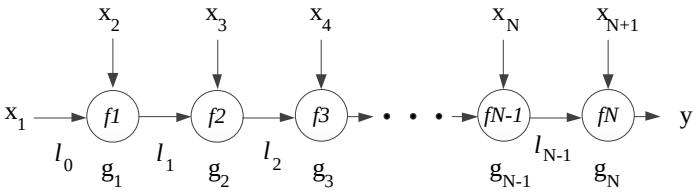


Рис. 1. Пример последовательного информационного графа

В случае если латентность всех вершин графа одинакова, а латентность соединительных дуг пренебрежимо мала, время выполнения всех операций графа будет равно

$$T_G = l_f \cdot N \cdot \tau,$$

где l_f – латентность каждой вершины графа.

Поскольку в реальных задачах значение N может достигать десятков и сотен тысяч, возникает необходимость сокращения параметра T_G. Если все входы такого графа независимы друг от друга, то мы в зависимости от прочих его свойств можем изменить структуру графа таким образом, чтобы уменьшить время выполнения всех его операций.

Предлагаемое решение. Допустим, что у нас имеется однородный последовательный информационный граф, все N вершин которого представляют собой ассоциативные операции одного типа (рис. 2). Обозначим эту операцию как *f*.

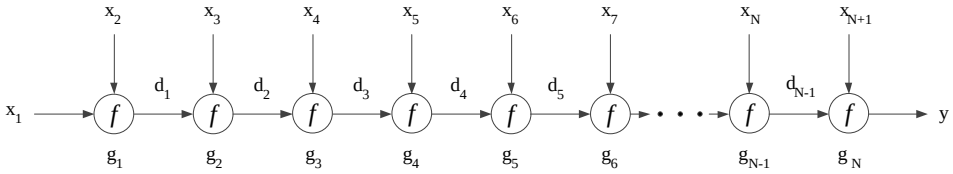


Рис. 2. Однородный последовательный информационный граф, состоящий из ассоциативных операций f

В случае если размер доступного нам оборудования составляет $R > N$, мы можем реализовать весь этот граф целиком. Для того чтобы сократить время выполнения всех операций графа, целесообразно привести его к пирамидальному виду.

Определим первую операцию – преобразование пары смежных вершин, представляющих собой одинаковые ассоциативные операции *f*, в последовательном информационном графе в параллельно-последовательную форму. Назовём её преобразованием пары ассоциативных вершин.

Согласно определению свойства ассоциативности:

$$\Big(\big((x_1 * x_2) * x_3\big) * \ldots\Big) * x_N = (x_1 * x_2) * \ldots * (x_{N-1} * x_N), \quad (1)$$

где * – рассматриваемая нами ассоциативная операция.

Введём параметр L – номер яруса, к которому относится вершина (изначально равен нулю для всех вершин).

Следовательно, мы можем произвести следующее преобразование элемента графа, включающего в себя две вершины (рис. 3):

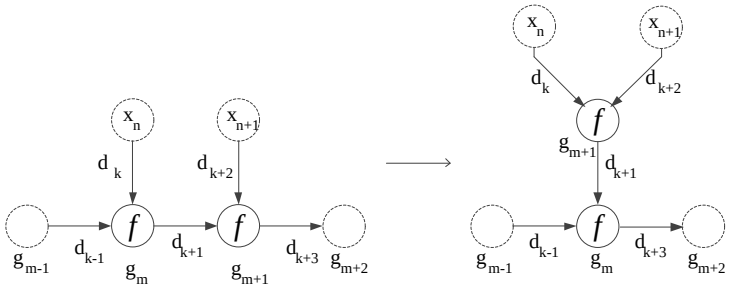


Рис. 3. Графический вид преобразования пары ассоциативных вершин

После преобразования параметр L вершины g_m будет увеличен на 1.

Для преобразования всего графа нам необходимо разбить его на пары смежных вершин и произвести над каждой парой преобразование пары ассоциативных вершин (рис. 4).

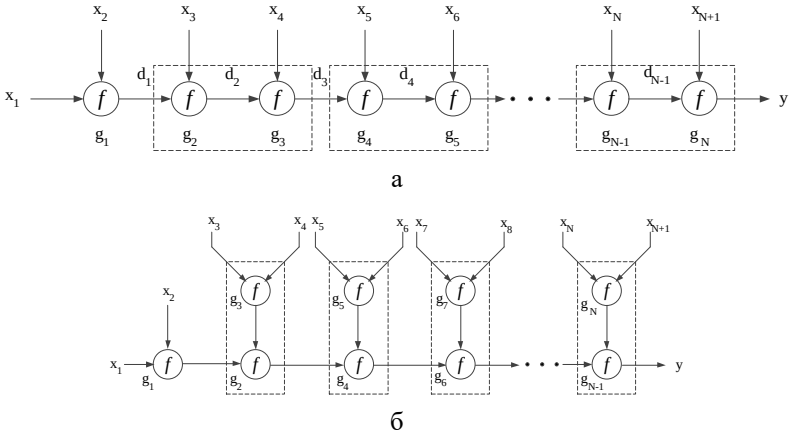


Рис. 4. Граф перед первым выполнением преобразования пары ассоциативных вершин над каждой парой (а) и после него (б)

В случае если количество вершин в графе чётное, то не преобразованными остаются первая и последняя вершины.

Затем мы повторяем преобразование для нижних вершин со значением $L = 1$, выбирая пары вершин по признаку наличия общих инцидентных дуг. Результат показан на рис. 5.

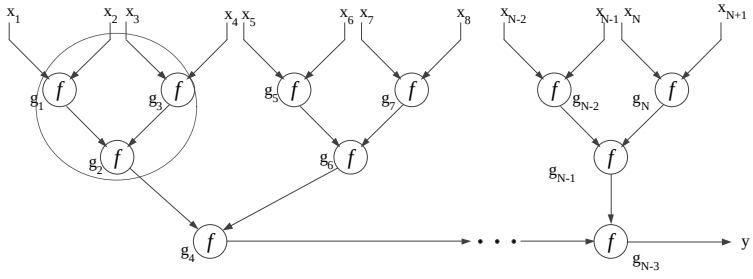


Рис. 5. Граф после второго выполнения преобразования пары ассоциативных вершин

Таким образом, алгоритм преобразования пары ассоциативных вершин для всего графа выглядит следующим образом:

1. Делим всё множество вершин на пары смежных, за исключением первой вершины и последней (в случае, если общее количество вершин в графе *N* чётное).
2. Последовательно производим преобразование пары ассоциативных вершин над парами, начиная с первой. В случае если *N* чётное, увеличиваем значение параметра *L* для последней вершины на 1.
3. Выбираем вершины с параметром *L* = 1, разбиваем их на пары по тому же принципу (не включаем в пару первую вершину и последнюю, в случае, если таких вершин чётное количество).
4. Повторяем п. 2 для этого множества.
5. П. 3 и п. 4 повторяем пока для очередного значения L количество вершин с таким значением не будет меньше трёх.

В результате граф будет приведён к пирамидальному виду (рис. 6).

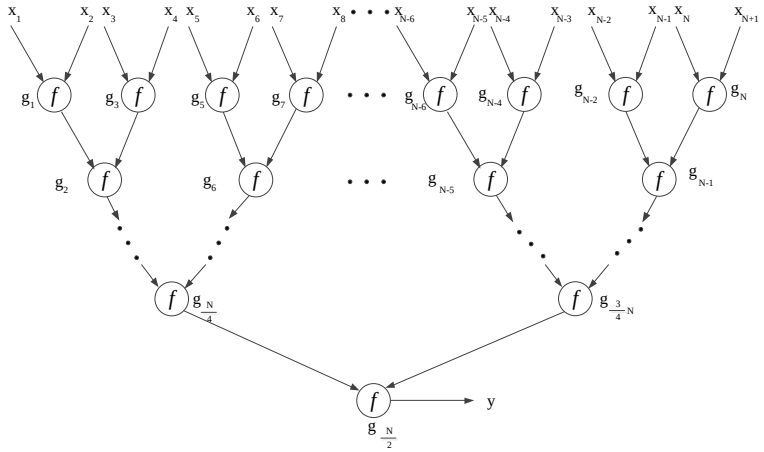


Рис. 6. Вид графа после выполнения всех доступных преобразований пар ассоциативных вершин

Время выполнения всех операций такого графа значительно сократится и составит

$$T_G = l_g \cdot \lceil log_2(N) \rceil \cdot \tau.$$

Таким образом, если количество доступного оборудования составляет $R \geq N$, мы можем ускорить полное выполнение всех операций графа в E_L раз, где

$$E_L = \frac{l_g \cdot N \cdot \tau}{l_g \cdot \lceil log_2(N) \rceil \cdot \tau} = \frac{N}{\lceil log_2(N) \rceil}.$$

В случае если $R < N$ мы не сможем реализовать весь граф целиком. В этом случае применяются методы редукции производительности, для применения которых необходимо представить граф как структуру, состоящую из подграфов (рис. 7). В этом случае множество входных вершин подграфов $x_1, x_2, \ldots, x_{N+1}$ заменяется кортежными вершинами, а дуги инцидентные связям подграфов заменяются обратными связями.

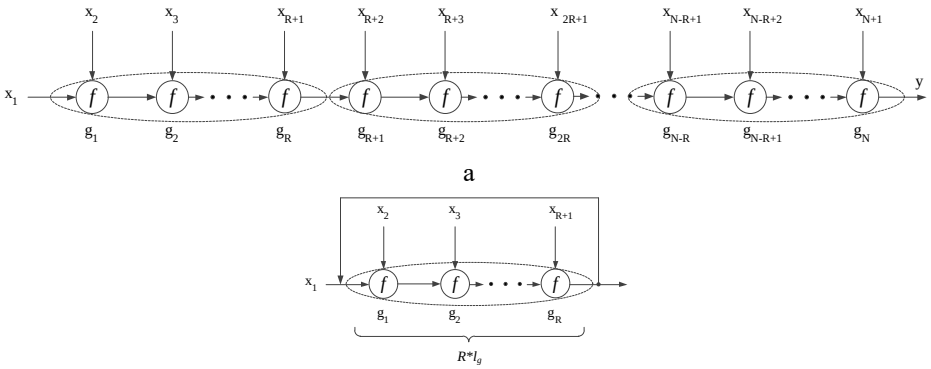


Рис. 7. Представление графа в виде совокупности подграфов (а) и вид вычислительной структуры с обратной связью (б)

В случае, если мы преобразуем таким образом исходный последовательный граф, мы столкнёмся со следующей проблемой: время заполнения вычислительного конвейера составит $R \cdot l_g$, т.к. подграфы являются информационно зависимыми и прежде, чем подавать следующий набор данных, необходимо дождаться полной обработки предыдущего, которая займёт $R \cdot l_g$ тактов. В результате мы получим скважность, равную R, что увеличит время выполнения всех операций графа в R раз.

$$T_G = R \cdot l_g \cdot N \cdot \tau.$$

Согласно теореме Иванова [5] если имеется некоторый информационный граф G, состоящий из множества взаимосвязанных изоморфных подграфов p_1, p_2, …, p_N, и ресурса системы недостаточно для аппаратной реализации всего графа G, то максимальная эффективность (удельная производительность) достигается за счет аппаратной реализации единственного подграфа p_i. В рассматриваемом случае это будет означать реализацию одной вершины (рис. 8).

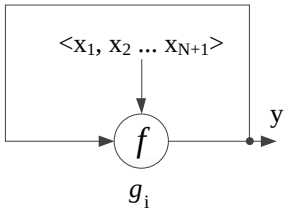


Рис. 8. Оптимальная структура графа согласно теореме Иванова

Время выполнения всех операций в этом случае составит

$$T_G = l_g \cdot (N + 1) \cdot \tau,$$

т.е. будет практически равно времени выполнения всех операций графа в случае, если $R \geq N$.

Эту величину можно уменьшить, если изоморфные подграфы, на которые разбивается исходный граф, привести к пирамидальному виду, как это было показано выше, и, что главное, обеспечить скважность, равную единице. Рассмотрим частный случай, когда параметр l_g равен единице.

Для преобразования всего графа к этой форме нам необходимо разбить его на части определённым образом. Все вершины делятся на группы, в каждую из которых *R-1* вершин. Между этими группами оставляется по одной вершине, которые составят последовательную часть графа после преобразования (рис. 9).

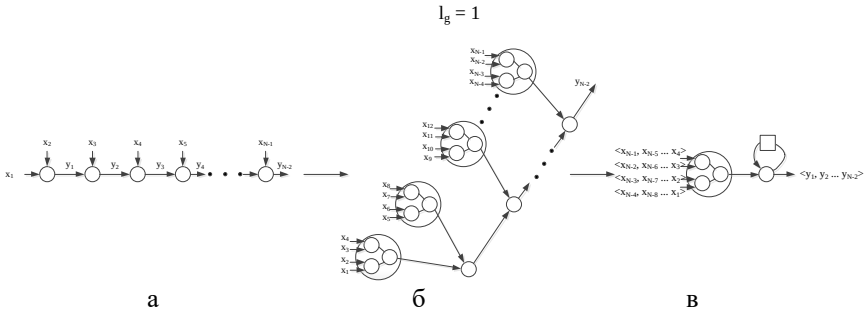


Рис. 9. Преобразование графа в параллельно-последовательную форму для случая $l_f = 1$. Исходный граф (а), преобразованный граф (б) и соответствующая ему вычислительная структура (в)

Мы выбираем первую группу вершин и преобразовываем её согласно пп. 1-5, как если бы эта группа вершин была целым графом. Затем мы переходим к следующей группе вершин и повторяем эти преобразования над ней. Одиночным вершинам присваивается значение параметра *L* равное -1.

После того, как преобразование выполнено над всеми группами, мы выделяем вторую из них и одиночную вершину, расположенную на графе слева от неё (рис. 10).

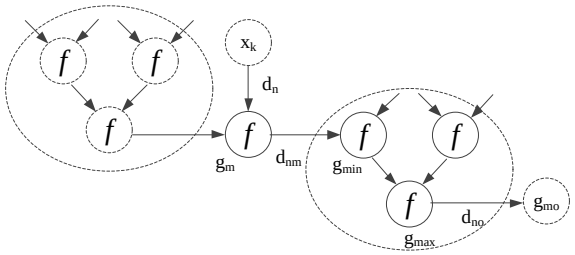


Рис. 10. Участок графа перед выполнением преобразования группы ассоциативных вершин

Результат преобразования показан на рис. 11.

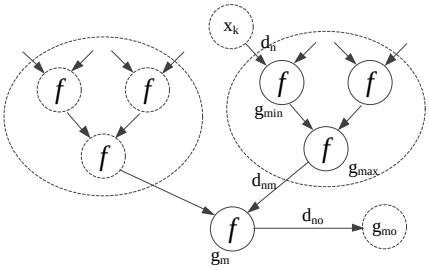


Рис. 11. Участок графа после выполнения преобразования группы ассоциативных вершин

Затем выделяем следующую пару «вершина с параметром L равным -1» и «группа вершин» и повторяем преобразование над этой парой – и так далее до обработки всех групп. Если общее количество вершин графа, который мы преобразовываем, равно

$$N = R \cdot k - 1 + n,$$

где k – количество групп, $k \in \mathbb{N}$, а $n \neq 0$, последняя группа дополняется транзитными вершинами, число которых равно $R - 1 - n$.

Таким образом, если у нас достаточно вычислительного ресурса для реализации R вычислительных узлов, выполняющих операцию f с латентностью 1, а размер графа составляет N, то граф будет разбит на $\left\lceil \frac{N}{R} \right\rceil$ базовых подграфов, каждый из которых будет содержать $R-1$ вершин в пирамидальной структуре и одну вершину, обеспечивающую обратную связь. В последней группе вершин будет содержаться $n = N - \left\lceil \frac{N}{R} \right\rceil \cdot R$ транзитных вершин, не выполняющих вычислений, но используемых для того, чтобы подграфы сохраняли регулярную структуру. Время выполнения всех операций графа будет равно

$$T_G = \left(\lceil log_2(R-1) \rceil + 1 + \left\lceil \frac{N+1}{R+1} \right\rceil \right) \cdot \tau. \quad (4)$$

Теперь рассмотрим последовательный информационный граф, состоящий из чередующихся вершин α и β, которые являются ассоциативными и операция β является дистрибутивной по отношению к операции α. Для простоты примем, что первая вершина графа является вершиной α, что количество вершин α равно количеству вершин β и что вершины разных типов чередуются (рис. 12).

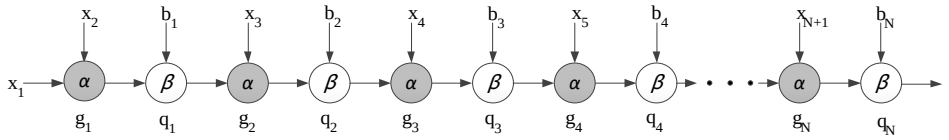


Рис. 12. Граф с чередованием двух ассоциативных операций, одна из которых является дистрибутивной по отношению к другой

Время выполнения всех операций графа будет равно

$$T_G = (l_{g\alpha} + l_{g\beta}) \cdot N \cdot \tau.$$

Для уменьшения T_G мы можем преобразовать граф в пирамидальную форму, как это было показано выше. Однако данный граф неоднороден и операции α и β в общем случае не являются ассоциативными по отношению друг к другу (хотя и являются ассоциативными по отдельности). Следовательно, прежде чем преобразовывать граф в пирамидальную форму, нам будет необходимо изменить его таким образом, чтобы выделить в нём однородные фрагменты, над которыми затем будет производиться преобразование пары ассоциативных вершин.

Согласно правилу дистрибутивности

$$(x_1 * x_2) \cdot b_1 = x_1 \cdot b_1 * x_2 \cdot b_1,$$

где $*$ – ассоциативная операция, $\cdot$ – операция, дистрибутивная по отношению к $*$.

Рассмотрим участок графа, на котором последовательно соединены две вершины: первая с операцией α, а вторая с операцией β. Мы можем преобразовать фрагмент графа, включающий в себя последовательно соединённые вершины α и β, продублировав вершину β и перенеся её с выхода вершины α, на каждый из её входов (рис. 13):

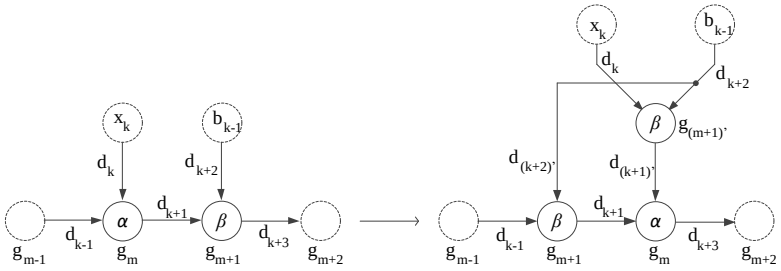


Рис. 13. Графический вид преобразования участка из двух вершин, одна из которых является дистрибутивной по отношению к другой

Последовательное применение преобразования смешанной пары вершин, когда мы перемещаем вершины β налево от вершин α и при этом с каждым переносом добавляем их копии, приведёт к тому, что исходный граф разделится на ветви, одна из которых состоит только из вершин α, а все остальные – только из вершин β (рис. 14).

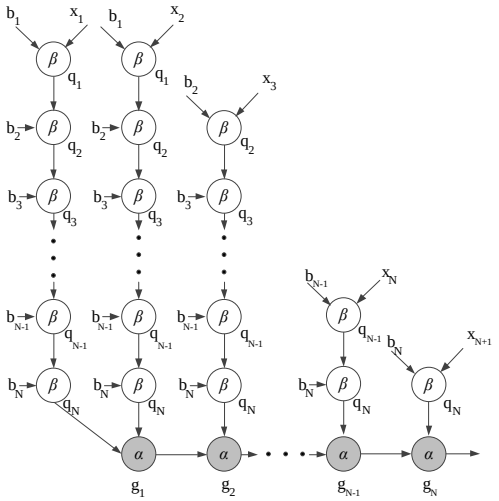


Рис. 14. Результат последовательного выполнения преобразования смешанной пары вершин над графом

Теперь, когда граф разбит на однородные фрагменты, состоящие из ассоциативных операций, мы можем выполнить над каждым из фрагментов преобразование пары ассоциативных вершин. В результате мы получим граф, состоящий из пирамиды из вершин α, на входы которой подаются выходы пирамид, состоящих из вершин β. При этом размер пирамид из вершин β будет меняться от N вершин (для двух крайне левых), до одной вершины (крайне правая пирамида) (рис. 15).

После выполнения этого преобразования и последующего выполнения преобразования над ассоциативными вершинами время выполнения всех операций графа составит

$$T_G = (l_{g\alpha} + l_{g\beta}) \cdot \lceil \log_2 N \rceil \cdot \tau.$$

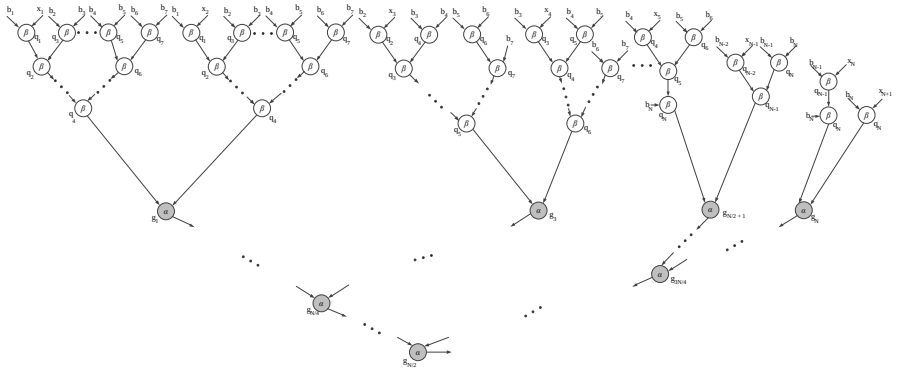
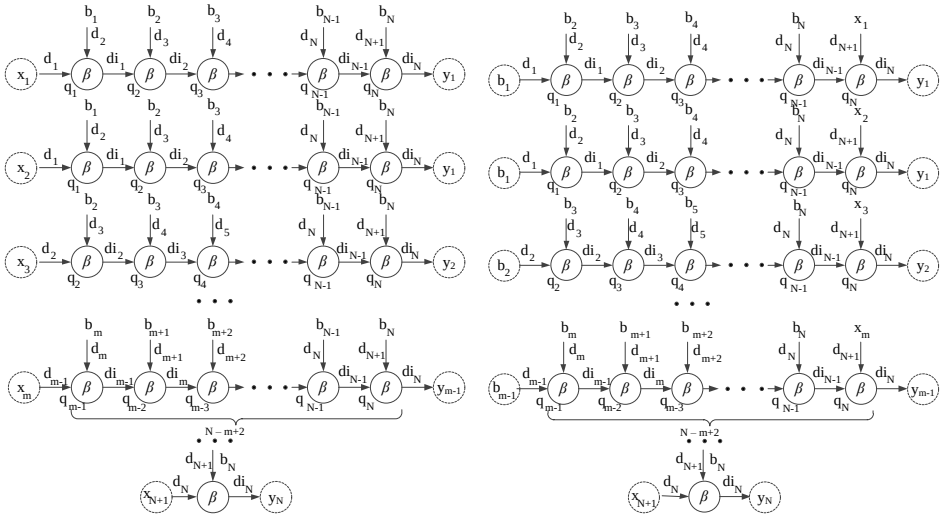


Рис. 15. Пирамидальная форма неоднородного графа

Это значение будет применимо для случая, когда у нас хватает оборудования R_α и R_β для реализации всего графа. Однако несложно заметить, что при использовании преобразования пары с дистрибутивной вершиной количество вершин N_β возрастает и в итоге становится равно $\frac{N^2+N}{2}$. N_α при этом остаётся равно N. Это приведёт к значительному повышению затрат оборудования: так, если в исходном графе было 1000 умножителей, после преобразования их количество составит 500500. В связи с этим желательно сократить количество затрачиваемого оборудования настолько, насколько это возможно.

Для сокращения затрат оборудования необходимо будет отказаться от преобразования ветвей с вершинами β к пирамидальному виду. В этом случае ветви графа с вершинами β будут иметь линейную структуру (рис. 16):



а б

Рис. 16. Структура ветвей вершин β сразу после выполнения третьего преобразования (а) и после выполнения преобразования множества вершин β (б)

Анализируя эту структуру, мы можем заметить следующее: 1. переменная x (в исходном графе входные данные недистрибутивных вершин) для каждой ветви своя; 2. две первые ветви включают в себя все переменные b (в исходном графе

входные данные дистрибутивных вершин), а каждая последующая включает на одну меньше, так, что *m*-я ветвь включает в себя переменные *b* с индексами (*m-1 … N*). Из этого можно сделать следующий вывод: для сокращения количества используемых вычислительных блоков β необходимо вынести блоки, на вход которых подаются переменные *x*, вплотную к части графа, состоящей из элементов α.

После выполнения этого преобразования надо всеми ветвями дистрибутивных вершин граф примет вид, показанный на рис. 17. Повторяющиеся вершины заштрихованы и могут быть удалены из графа.

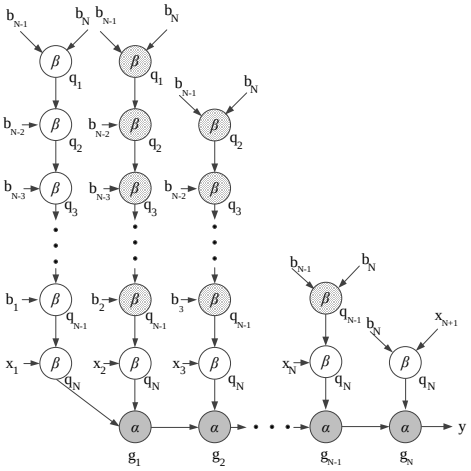


Рис. 17. Структура ветвей дистрибутивных вершин графа после перестановки вершин со входами x

После этого количество уникальных вершин β в преобразованном графе будет примерно равно *2*N-1*, а не $\frac{N^2+N}{2}$. Количество вершин α не изменится и составит *N*.

Подграф, состоящий из вершин α, может быть приведён к пирамидальной форме (рис 18).

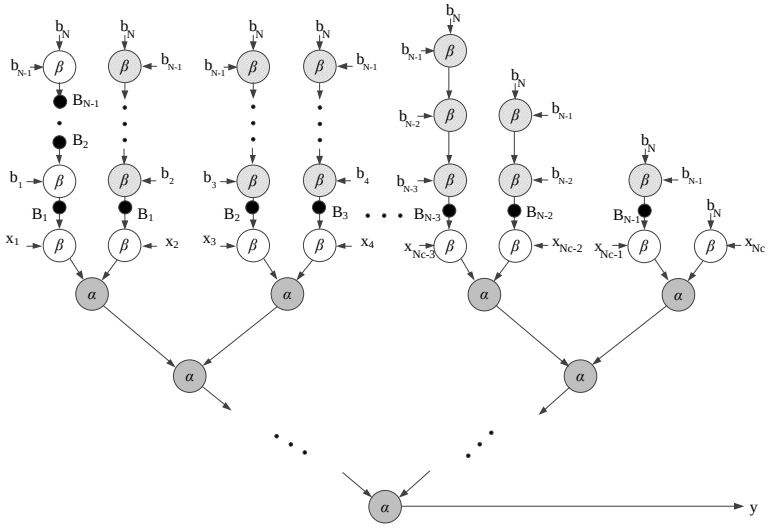


Рис. 18. Вариант линейно-пирамидальной структуры графа

В этом случае время выполнения всех операций вычислительной структуры, соответствующей этому графу, составит:

$$T_G = (l_{g\alpha} \cdot \lceil log_2 N \rceil + l_{g\beta} \cdot N) \cdot \tau.$$

В случае если $R_\alpha < N_\alpha$ и/или $R_\beta < N_\beta$, мы будем вынуждены разбить граф на подграфы и реализовывать по одному из них единовременно.

Рассмотрим множество вершин $\{g_1, g_2, \ldots g_N\}$ на рис. 14. В случае если граф будет охвачен петлёй обратной связи, желательно минимизировать длину цепочки, соединяющей вершину g_1 и выход y. Следовательно необходимо произвести следующее преобразование (рис. 19):

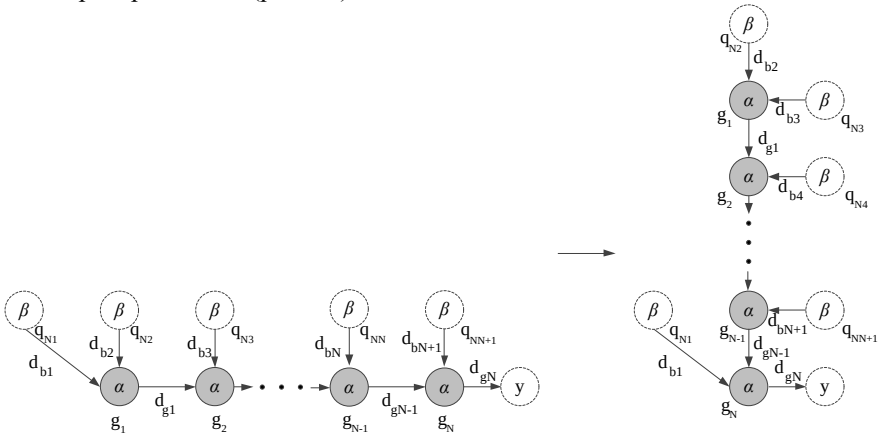


Рис. 19. Графический вид преобразования множества вершин α

Последовательное применение преобразования смешанной пары вершин и последующая перестановка вершин, с целью сократить повторяющиеся вершины, приведёт подграф к следующему виду:

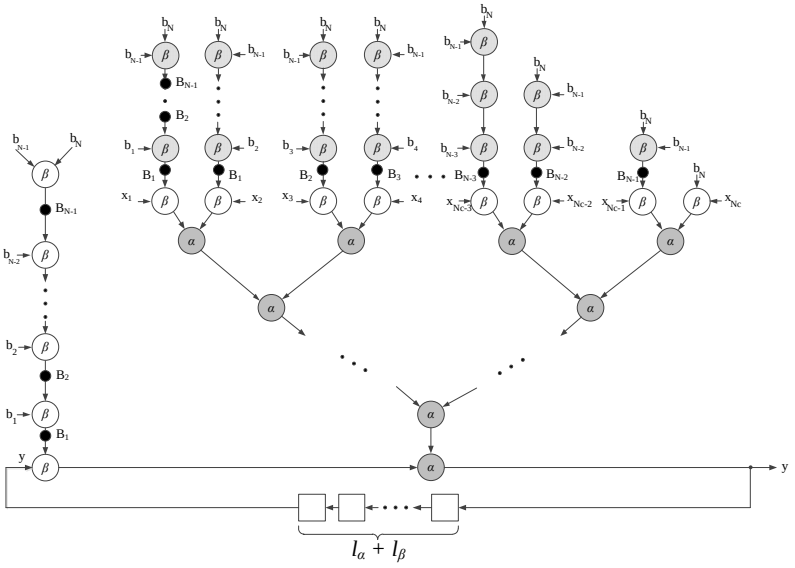


Рис. 20. Вид подграфа после преобразований

Согласно теореме Иванова максимальная производительность в данном случае будет достигаться при реализации фрагмента, состоящего из двух последовательно соединённых вычислительных блоков: α и β (рис. 21).

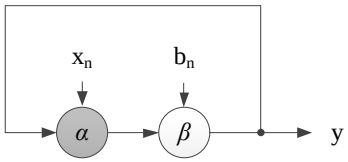


Рис. 21. Наименьший смешанный подграф, соответствующий оптимальной вычислительной структуре согласно теореме Иванова

Время реализации всех операций графа в этом случае составит:

$$T_G = (l_\alpha + l_\beta + \frac{N}{2}) \cdot \tau \cdot S,$$

где $S = l_\alpha + l_\beta$ – скважность, с которой будут подаваться данные, равная латентности подсистемы, реализующей наименьший смешанный подграф: последовательно соединённые блоки α и β;

$N = N_\alpha + N_\beta$ – общее число вершин в исходном графе.

Общая латентность подсистемы, равная $l_\alpha + l_\beta$, определит также время заполнения вычислительного конвейера, которое, в свою очередь, определит минимальную скважность, с которой будут подаваться входные данные. Подача данных плотным потоком здесь будет невозможна, поскольку предыдущая партия должна будет закончить обработку и попасть на вход блока α до того, как на другой его вход будет подана следующая партия.

Очевидно, что максимальным членом этого выражения будет являться $\frac{N}{2}$ и что значение этого выражения никак не зависит от доступного нам вычислительного ресурса. Это объясняется тем, что если мы реализуем большее количество пар блоков α и β, то скважность, с которой придётся подавать входные данные, возрастёт пропорционально. Более того, при реализации такой вычислительной структуры возникнут дополнительные задержки. Таким образом кажется, что мы никак не можем ускорить выполнение всех операций такого графа, в случае, если у нас не хватает оборудования на то, чтобы реализовать его целиком.

Однако лучший результат может быть достигнут в случае, если мы реализуем фрагмент вида, представленного на рис. 30, состоящий из M_α вершин α и M_β вершин β, где $M_\alpha = l_\alpha + l_\beta$, $M_\beta = 2 \cdot (l_\alpha + l_\beta)$. В этом случае раз в S тактов будут подаваться $2 \cdot (l_\alpha + l_\beta)$ данных, что будет означать ускорение подачи данных в $l_\alpha + l_\beta$ раз по сравнению с предыдущим вариантом. В результате время выполнения всех операций станет равно:

$$T_G = (L + \left\lceil \frac{N}{2 \cdot (l_\alpha + l_\beta)} \right\rceil) \cdot \tau \cdot S,$$

где $L = (l_\alpha + l_\beta) \cdot l_\beta + \lceil log_2(l_\alpha + l_\beta) \rceil \cdot l_\alpha$ – латентность критического (т.е. наибольшего) пути внутри вычислительной структуры.

Заключение. Таким образом, в зависимости от соотношения числа доступных нам вычислительных блоков обоих типов, их латентности и общего числа вершин в рассматриваемом графе, мы можем выделить несколько вариантов построения вычислительной структуры, каждый из которых выполняет все операции графа со своей скоростью. Это позволит выбирать оптимальный способ преобразования графа в зависимости от соотношения числа его вершин и доступных нам вычислительных ресурсов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Левин И.И., Дордопуло А.И.* К вопросу об автоматическом создании параллельных прикладных программ для реконфигурируемых вычислительных систем // Вычислительные технологии. – 2020. – Т. 25, № 1. – С. 66-81.
2. *Левин И.И., Дордопуло А.И., Писаренко И.В., Михайлов Д.В.* Представление графов с ассоциативными операциями на языке программирования SET@L // Известия ЮФУ. Технические науки. – 2020. – № 3. – С. 98-111.
3. *Кнут Д.Э.* Искусство программирования. Т.4, А. Комбинаторные алгоритмы. Ч. 1: пер. с англ. – М.: ООО «И. Д. Вильямс», 2013. – 960 с.
4. *Новиков Ф.* Дискретная математика. – 3-е изд. – СПб.: Питер, 2019. – 496 с.
5. *Иванов А.И.* Методы и средства создания эффективного параллельно-конвейерного программного обеспечения вычислительных систем, построенных на основе ПЛИС-технологии: дисс. … канд. техн. наук, по специальности: 05.13.11 “Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей”, научный руководитель: чл.-корр. РАН, д.т.н., проф. Каляев И.А., дис. совет ТРТУ Д 212.259.05, 2005. – 182 с.
6. Задача суммирования элементов массива. – Лаборатории Параллельных информационных технологий НИВЦ МГУ. – URL: https://parallel.ru/fpga/Summ2 (дата обращения: 27.04.2020).
7. *Ефимов С.С.* Обзор методов распараллеливания алгоритмов решения некоторых задач вычислительной дискретной математики // Математические структуры и моделирование. – 2007. – Вып. 17. – С. 72-93.
8. *Tessier R., Pocek K., DeHon A.* Reconfigurable Computing Architectures // Proceedings of the IEEE. – 2015. – Vol. 103, No. 3. – P. 332-354.
9. *Mittal S., Vetter J.* A survey of CPU-GPU heterogeneous computing techniques // ACM Computing Surveys. – 2015. – Vol. 47. – Art. 69.
10. *Waidyasooriya H.M., Hariyama M., Uchiyama K.* Design of FPGA-Based Computing Systems with OpenCL. – Cham: Springer, 2018. – 126 p.
11. *Reinhard Diestel.* Graph Theory. Springer-Verlag, Heidelberg // Graduate Texts in Mathematics. – 2016. – Vol. 173. – 447 p. – ISBN 978-3-662-53621-6.
12. *Edward A. Bender.* Lists, Decisions and Graphs. With an Introduction to Probability. S. Gill Williamson. – 2010. – 251 p.
13. *Trudeau, Richard J.* Introduction to graph theory. – Dover Publications, Inc. New York, 1993. – 224 p.
14. *Старченко А.В., Берцун В.Н.* Методы параллельных вычислений. – Томск: Изд-во Том. ун-та, 2013. – 223 с.
15. *Левин И.И., Дордопуло А.И.* К вопросу об автоматическом создании параллельных прикладных программ для реконфигурируемых вычислительных систем // Вычислительные технологии. – 2020. – Т. 25, № 1. – С. 66-81.
16. *Каляев А.В., Левин И.И.* Модульно-наращиваемые многопроцессорные системы со структурно-процедурной организацией вычислений. – М.: Янус-К, 2003. – 380 с.
17. *Левин И.И., Дордопуло А.И., Гудков В.А. и др.* Средства программирования реконфигурируемых и гибридных вычислительных систем на основе ПЛИС // XIII Междунар. конф. «Параллельные вычислительные технологии» (ПаВТ-2019): Короткие статьи и описания плакатов. – Челябинск: Изд. центр ЮУрГУ, 2019. – С. 299-312.
18. *Дасгупта С., Пападимитриу Х., Вазирани У.* Алгоритмы: пер. с англ. / под ред. А. Шеня. – М.: МЦНМО, 2014. – 320 с.
19. *Харари Ф.* Теория графов. – М.: Мир, 1973. – 300 с.
20. *Кормен Т.М. и др.* Часть VI. Алгоритмы для работы с графами. Алгоритмы: построение и анализ = Introduction to Algorithms. – 2-е изд. – М.: Вильямс, 2006. – 1296 с.

REFERENCES

1. *Levin I.I., Dordopulo A.I.* K voprosu ob avtomaticheskom sozdanii parallel'nykh prikladnykh programm dlya rekonfiguriruemykh vychislitel'nykh sistem [On the problem of automatic development of parallel applications for reconfigurable computer systems], *Vychislitel'nye tekhnologii* [Computational Technologies], 2020, Vol. 25, No. 1, pp. 66-81.

2. *Levin I.I., Dordopulo A.I., Pisarenko I.V., Mikhaylov D.V.* Predstavlenie grafov s assotsiativnymi operatsiyami na yazyke programmirovaniya SET@L [Description of graphs with associative operations in SET@L programming language], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2020, No. 3, pp. 98-111.
3. Knut D.E. Iskusstvo programmirovaniya. T.4, A. Kombinatornye algoritmy [The Art of Computer Programming. Vol. 4, A. Combinatorial Algorithms]. Part 1: transl. from engl. Moscow: OOO «I. D. Vil'yams», 2013, 960 p.
4. *Novikov F.* Diskretnaya matematika [Discrete Mathematics]. 3rd ed. Saint Petersburg: Piter, 2019, 496 p.
5. *Ivanov A.I.* Metody i sredstva sozdaniya effektivnogo parallel'no-konveyernogo programmnogo obespecheniya vychislitel'nykh sistem, postroennykh na osnove PLIS-tekhnologii: diss. … kand. tekhn. nauk, po spetsial'nosti: 05.13.11 "Matematicheskoe i programmnoe obespechenie vychislitel'nykh mashin, kompleksov i komp'yuternykh setey" [Methods and tools for creating effective parallel pipeline software for computing systems based on FPGA technology: diss. … cand. of eng. sc., specialty: 05.13.11 "Mathematical and software support for computers, complexes and computer networks"], scientific director: Corresponding Member of RAS, d.t.s., proff. Kalyaev I.A., dis. council TSREU D 212.259.05, 2005, 182 p.
6. Zadacha summirovaniya elementov massiva [Problem of Array Elements' Summation], The Laboratory of Parallel Information Technologies of the Research Computing Center of the Moscow State University. Available at: https://parallel.ru/fpga/Summ2 (accessed 27 April 2020).
7. *Efimov S.S.* Obzor metodov rasparallelivaniya algoritmov resheniya nekotorykh zadach vychislitel'noy diskretnoy matematiki [Review of Parallelizing Methods for Algorithms Aimed at Solution of Certain Problems of Computational Discrete Mathematics], *Matematicheskie struktury i modelirovanie* [Mathematical Structures and Modeling], 2007, Issue 17, pp. 72-93.
8. *Tessier R., Pocek K., DeHon A.* Reconfigurable Computing Architectures, *Proceedings of the IEEE*, 2015, Vol. 103, No. 3, pp. 332-354.
9. *Mittal S., Vetter J*. A survey of CPU-GPU heterogeneous computing techniques, *ACM Computing Surveys*, 2015, Vol. 47, Art. 69.
10. *Waidyasooriya H.M., Hariyama M., Uchiyama K.* Design of FPGA-Based Computing Systems with OpenCL. Cham: Springer, 2018, 126 p.
11. *Reinhard Diestel.* Graph Theory. Springer-Verlag, Heidelberg, *Graduate Texts in Mathematics*, 2016, Vol. 173, 447 p. ISBN 978-3-662-53621-6.
12. *Edward A. Bender.* Lists, Decisions and Graphs. With an Introduction to Probability. S. Gill Williamson, 2010, 251 p.
13. *Trudeau, Richard J.* Introduction to graph theory. Dover Publications, Inc. New York, 1993, 224 p.
14. *Starchenko A.V., Bertsun V.N.* Metody parallel'nykh vychisleniy [Methods of Parallel Computing]. Tomsk: Izd-vo Tom. un-ta, 2013, 223 p.
15. *Levin I.I., Dordopulo A.I.* K voprosu ob avtomaticheskom sozdanii parallel'nykh prikladnykh programm dlya rekonfiguriruemykh vychislitel'nykh sistem [On the problem of automatic development of parallel applications for reconfigurable computer systems], *Vychislitel'nye tekhnologii* [Computational Technologies], 2020, Vol. 25, No. 1, pp. 66-81.
16. *Kalyaev A.V., Levin I.I.* Modul'no-narashchivaemye mnogoprotsessornye sistemy so strukturno-protsedurnoy organizatsiey vychisleniy [Modular-Expandable Multiprocessor Systems with Structural and Procedural Organization of Calculations]. Moscow: Yanus-K, 2003, 380 p.
17. *Levin I.I., Dordopulo A.I., Gudkov V.A. i dr.* Sredstva programmirovaniya rekonfiguriruemykh i gibridnykh vychislitel'nykh sistem na osnove PLIS [Tools for Programming of Reconfigurable and Hybrid Computer Systems Based on FPGAs], *XIII Mezhdunar. konf. «Parallel'nye vychislitel'nye tekhnologii» (PaVT-2019): Korotkie stat'i i opisaniya plakatov* [XIII International Conference «Parallel computational technologies» (PCT'2019), short papers and poster descriptions]. Chelyabinsk: Izd. tsentr YuUrGU, 2019, pp. 299-312.
18. *Dasgupta S., Papadimitriu Kh., Vazirani U.* Algoritmy [Algorithms]: transl. from engl., ed. by A. Shenya. Moscow: MTSNMO, 2014, 320 p.
19. *Kharari F.* Teoriya grafov [Graph Theory]. Moscow: Mir, 1973, 300 p.
20. *Kormen T.M. i dr. Chast' VI.* Algoritmy dlya raboty s grafami. Algoritmy: postroenie i analiz = Introduction to Algorithms [Part IV. Algorithms for working with graphs, Algorithms: construction and analysis = Introduction to Algorithms]. 2nd ed. Moscow: Vil'yams, 2006, 1296 p.

Статью рекомендовал к опубликованию д.т.н. Э.В. Мельник.

Михайлов Денис Васильевич – ООО "Научно-исследовательский центр супер-ЭВМ и нейрокомпьютеров";e-mail: mixailow.den@gmail.com; 347900, г. Таганрог, ул. Фрунзе, 61, кв. 11; тел.: 89287502869; аспирант.

Mikhailov Denis Vasilevich – LLC "Scientific-research center of supercomputers and Neurocomputers"; e-mail: mixailow.den@gmail.com; 61, Frunze street, ap. 11, Taganrog, 347900, Russia; phone: +79287502869; postgraduate student.

Раздел III. Реконфигурируемые вычислительные системы

УДК 004.442.2 DOI 10.18522/2311-3103-2020-7-94-106

А.И. Дордопуло, И.И. Левин, В.А. Гудков, А.А. Гуленок, А.В. Бовкун, С.А. Дудко

КОМПЛЕКС СРЕДСТВ ТРАНСЛЯЦИИ ПРОГРАММ НА ЯЗЫКЕ С В ПРОГРАММЫ НА ЯЗЫКЕ ПОТОКА ДАННЫХ COLAMO

Рассматриваются программные средства трансляции последовательных программ на языке С в масштабируемые параллельно-конвейерные программы на языке программирования реконфигурируемых вычислительных систем COLAMO. В отличие от существующих средств высокоуровневого синтеза, результатом трансляции является не IP-ядро фрагмента задачи, а комплексное решение задачи для многокристальных реконфигурируемых вычислительных систем с автоматической синхронизацией информационных и управляющих сигналов. Рассмотрены основные этапы трансляции последовательной программы на языке С: преобразование в информационный граф, анализ информационных зависимостей и выделение функциональных подграфов, преобразование в масштабируемую ресурсонезависимую параллельно-конвейерную форму и масштабирование программы на языке COLAMO для заданной многокристальной реконфигурируемой вычислительной системы. Масштабирование программы осуществляется с помощью методов редукции производительности абсолютно-параллельной формы задачи – информационного графа, который адаптируется под архитектуру реконфигурируемой вычислительной системы. Разработан ряд правил, позволяющих существенно сократить число шагов преобразований при масштабировании задачи и обеспечить плотный поток обработки данных в функциональных подграфах задачи. Созданный комплекс средств трансляции программ на языке С в конфигурационные файлы ПЛИС позволяет существенно сократить время синтеза вычислительной структуры задачи для многокристальных РВС и обеспечить сокращение общего времени решения задачи.

Информационные графы; компилятор; трансляция программ; язык С; редукция производительности; реконфигурируемые вычислительные системы; программирование многопроцессорных вычислительных систем.

A.I. Dordopulo, I.I. Levin, V.A. Gudkov, A.A. Gulenok, A.V. Bovkun, S.A. Dudko

HIGH-LEVEL TOOLS FOR TRANSLATION OF C-APPLICATIONS INTO APPLICATIONS IN DATAFLOW LANGUAGE COLAMO

In the paper we review software tools for translation of sequential C-programs into scalable parallel-pipeline programs written in the COLAMO language, used for programming of reconfigurable computer systems. In contrast to existing tools of high-level synthesis, the translation result is not an IP-core of a task fragment, but a complex task solution for multichip reconfigurable computer systems with automatic synchronization of data and control signals. We analysed the main translation steps of a sequential C-program such as transformation into an information graph, analysis of data dependencies and selection of functional subgraphs, transformation into a scalable resource-independent parallel-pipeline form, and scaling a COLAMO-program for a specified multichip reconfigurable computer system. A program is scaled with the help of performance reduction methods, applied to a completely parallel form of a task (an information graph), adapted to the architecture of a reconfigurable computer system. We developed several rules,

significantly reducing the number of transformation steps of task scaling, and providing a continuous flow of data processing in the functional subgraphs of the task. The developed software tools for translation of C-programs into FPGA configuration files significantly decrease the synthesis time of a task computing structure for multichip RCSs and the total task solution time.

Information graph; compiler; translation of programs; C language; performance reduction; reconfigurable computer system; programming of multiprocessor computer systems.

Введение. Сокращение времени решения задачи является основной прагматичной целью высокопроизводительных вычислений. Ускорение вычислений достигается повышением быстродействия вычислительных узлов многопроцессорной вычислительной системы (МВС), максимальным распараллеливанием вычислительных операций или сочетанием обоих подходов [1]. На практике максимальное распараллеливание, как правило, недостижимо, потому что возможности распараллеливания вычислительных операций задачи ограничены соотношением между размерностью задачи и доступным аппаратным ресурсом вычислительной системы. Поэтому при реализации прикладной задачи в МВС разработчик выполняет поиск наиболее рационального варианта ее реализации, обеспечивающего минимальное время решения с учетом характеристик доступного аппаратного ресурса.

Для МВС, построенных на основе процессорной архитектуры, задача представляется в парадигме *передачи управления* от одного процесса (или вычислительного устройства) к другому, на которых они реализуются последовательно с помощью команд процессора. Характерные для большинства задач информационные зависимости по данным при передаче управления могут быть нарушены, что не позволяет достичь ускорения вычислений, пропорционального числу используемых узлов. Поэтому на реальных вычислительных задачах производительность таких МВС существенно сокращается до 10–15% от пиковой.

Учет информационных зависимостей в структуре прикладной задачи обеспечивается в концепции структурно-процедурной организации вычислений (СПОВ), что позволяет обеспечить высокую реальную производительность вычислений на реконфигурируемых вычислительных системах (РВС) [2] с программируемыми логическими интегральными схемами (ПЛИС) [2], которые находят все больше применений при решении вычислительно трудоемких задач в различных областях науки и техники [3]. РВС обладают существенными преимуществами в реальной производительности и энергоэффективности по сравнению с МВС кластерной архитектуры, но их широкое применение во многом сдерживается высокой сложностью программирования. Сложность эффективного программирования РВС на основе ПЛИС, признанная многими исследователями [1–3], несмотря даже на наличие языков программирования высокого уровня (HandelC[4], SystemC[4], COLAMO[2]), побуждает искать решения в области разработки и создания средств автоматической трансляции последовательных программ (например, на языке С) в конфигурационные файлы ПЛИС.

Принципы и методы создания средств высокоуровневого синтеза Трансляторы программ на языке С в конфигурационные файлы ПЛИС [4–9] получили названия средств высокоуровневого синтеза (High Level Synthesis), которые по типу входного языка программирования могут быть разделены на две основные категории [4]: трансляторы проблемно-ориентированных языков (Domain Specific Languages – адаптированной к определенной проблемной области версии языка программирования С) и трансляторы языков общего назначения (General Purpose Languages – диалекты языка программирования С с некоторыми особенностями и ограничениями). В настоящее время активно развиваются как академические (DWARV [5], BAMBU [6] и LEGUP [7]), так и коммерческие (CatapultC, Vivado HLS [8], Vivado Vitis[9]) комплексы проектирования. Подробный и детальный обзор существующих средств высокоуровневого синтеза и сравнительный анализ их

возможностей при трансляции задач некоторых предметных областей приведен в [4]. Подавляющее большинство приведенных в [4] трансляторов программ на языке С преобразуют вычислительно трудоемкий фрагмент задачи в IP-ядро и синтезируют в ПЛИС специализированный вычислитель на основе автоматной модели или процессорной парадигмы. Несмотря на существенный выигрыш в скорости вычислений по сравнению с процессорной реализацией (по данным обзора [4]), используемые для решения задачи IP-ядра реализуют только лишь фрагмент задачи и для создания завершенного решения необходимо участие высококвалифицированного программиста. Масштабирование решения, даже в пределах одного кристалла ПЛИС, также полностью возлагается на программиста.

В отличие от рассмотренных в [4] трансляторов программ на языке С, разрабатываемый в НИЦ супер-ЭВМ и нейрокомпьютеров комплекс средств трансляции программ на языке С в программы на языке потока данных COLAMO исходно предназначен для синтеза комплексного решения прикладной задачи для РВС, содержащих вычислительное поле ПЛИС, связанных пространственной коммутационной системой. Комплекс отображает последовательную программу на языке С в абсолютно-параллельную форму, которую с помощью редукционных преобразований адаптирует под доступный аппаратный ресурс многокристальных РВС с учетом информационных связей и зависимостей данных задачи. При схожей функциональности от наиболее близких аналогов (Xilinx Vivado HLS [8] и Xilinx Vitis [9]) комплекс отличается не только поддержкой многокристальных решений, но и автоматической синхронизацией информационных и управляющих сигналов.

Прикладная задача на последовательной языке программирования С компонентами комплекса преобразуется в абсолютно параллельную форму – информационный граф [1, 2], отражающий естественный параллелизм и конвейеризацию операций, который может быть адаптирован к архитектуре и текущей конфигурации РВС с помощью методов масштабирования (индукции и редукции) производительности кадровой структуры [2]. Масштабирование задачи (рис. 1) представляется движением в трехмерном пространстве: слоев (информационно-независимые реализации базового подграфа), итераций (информационно-зависимые реализации базового подграфа) и команд (вычислительные устройства заданной разрядности, составляющие базовый подграф).

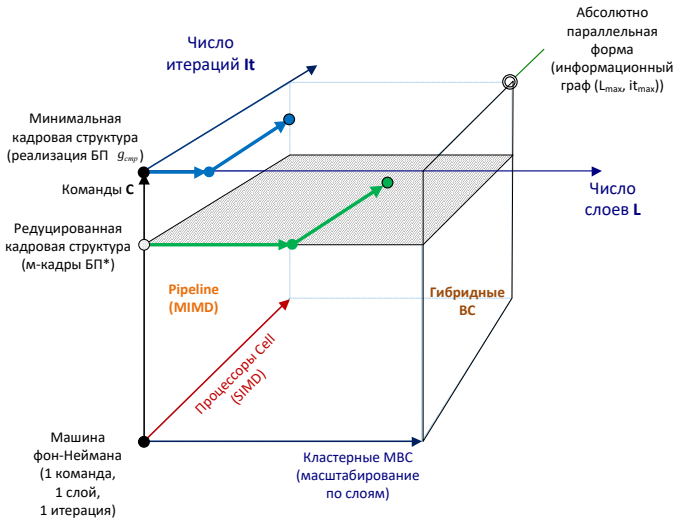


Рис. 1. Пространство реализаций вычислений для различных архитектур

В представленной на рис. 1 модели пространства возможных решений задачи реализации вычислений на процессоре соответствует точка «Машина фон-Неймана (1 команда, 1 слой, 1 итерация)», а характерное для кластерных МВС увеличение числа процессоров при реализации информационно-независимых фрагментов вычислений соответствует движению по оси «Число слоев L». Конвейеризация вычислительных операций при реализации информационно-зависимых фрагментов вычислений, характерная для вычислителей с архитектурой SIMD, примером которой являются процессоры Cell, соответствует движению от этой точки по оси «Число итераций It». Заданные осями «Команды C» и «Число итераций It» параллельные плоскости «Pipeline(MIMD)» и «Гибридные ВС» описывают реализацию вычислений для MIMD-вычислителей и гибридных вычислительных систем с графическими ускорителями.

В парадигме СПОВ [2] масштабирование задачи осуществляется в плоскости слоев и итераций движением от минимальной кадровой структуры, реализующей базовый подграф (БП) задачи $g_{стр}$, к абсолютно-параллельной форме задачи – кадровой структуре $(L_{\max}, It_{\max})$, соответствующей информационному графу задачи. Масштабирование выполняется индуктивным тиражированием минимальной кадровой структуры, реализующей БП $g_{стр}$. Минимальная кадровая структура, как правило, является структурной конвейерной реализацией БП $g_{стр}$, синтезированной инженером-схемотехником вручную для заданного кристалла ПЛИС, оптимальной как по числу каналов, так и по интервалу обработки данных. Поэтому автоматическое масштабирование решения задачи для структурно-процедурной организации вычислений затруднено необходимостью разработки методов и средств не только автоматического выделения БП $g_{стр}$, но и его оптимальной конвейерной реализации.

В рассматриваемом комплексе средств трансляции программ на языке С в программы на языке потока данных COLAMO используется другой подход. Масштабирование начинается не от минимальной кадровой структуры, а от абсолютно-параллельной кадровой формы задачи $(L_{\max}, It_{\max})$, соответствующей информационному графу задачи. Движение осуществляется с помощью методов редукции производительности и аппаратных затрат [10], что не требует ручного выделения БП $g_{стр}$ и позволяет находить решение для случаев, когда доступного аппаратного ресурса недостаточно для реализации даже минимальной кадровой структуры $g_{стр}$. В этом случае она редуцируется до м-кадров (микро-кадров, заштрихованная плоскость на рис. 1), последовательно выполняющих часть команд БП $g_{стр}$ на меньшем аппаратном ресурсе. М-кадры, как и минимальную кадровую структуру, можно индуктивно масштабировать в плоскости слоев и итераций для получения рационального решения.

Принципы масштабирования программы на языке С под доступный аппаратный ресурс при трансляции в COLAMO. Методику автоматического масштабирования задачи в пространстве возможных кадровых структур для многокристальных РВС при трансляции программы на языке С в программы на языке потока данных COLAMO можно представить следующими последовательными преобразованиями.

1. Построение информационного графа задачи – преобразование входной программы на языке С в абсолютно-параллельную форму, описанную синтаксическими конструкциями COLAMO.

2. Анализ структуры информационного графа задачи: выделение подзадач, анализ информационных зависимостей в структуре каждой подзадачи и между ними, определение числа слоев и итераций для всех фрагментов всех подзадач, расщепление скалярных переменных и растягивание массивов по итерациям для устранения нарушений правил однократного присваивания и единственной подстановки.

3. Преобразование задачи в масштабируемую параллельно-конвейерную форму: добавление потоковой составляющей ко всем векторным измерениям массивов и циклам обработки в программе на COLAMO, расчет возможных значений параметров масштабирования по аппаратному ресурсу и числу каналов памяти.

4. Масштабирование задачи: движение от абсолютно-параллельной формы по слоям, итерациям и командам в соответствии с рассчитанными параметрами с помощью методов редукции производительности, расчет и подбор значений параметров масштабирования для рационального использования доступного ресурса РВС.

5. Оптимизация полученного решения задачи по критерию минимума времени: сокращение интервала обработки данных с помощью преобразования к конвейеру конвейеров или макроконвейеру [2].

Выполнение первых трех пунктов приведенной методики позволяет получить масштабируемую ресурсонезависимую параллельную прикладную программу на языке COLAMO, четвертый пункт методики адаптирует полученную программу под доступный вычислительный ресурс, а пятый пункт синтезирует рациональное (по времени выполнения) решение задачи. Далее синтезированная программа на языке программирования COLAMO транслируется в схемотехническую конфигурацию с помощью разработанных транслятора COLAMO и синтезатора Fire!Constructor [10] с автоматическими распределением по кристаллам ПЛИС и синхронизацией вычислений между ПЛИС.

Каждый из пунктов методики является функционально-завершенным преобразованием, которое выполняется отдельным программным компонентом комплекса средств трансляции программ на языке С в программы на языке потока данных COLAMO.

Структура комплекса средств трансляции. Разработанная с учетом описанной методики структурная схема комплекса средств трансляции представлена на рис. 2. Трансляцию из С в COLAMO выполняют следующие программные компоненты:

- транслятор «Ангел», преобразующий программы на языке программирования «С» в стандарте ISO/IEC 9899:1999 в абсолютно-параллельную программу на языке Colamo;
- процессор «Русалка», преобразующий абсолютно-параллельную программу на языке Colamo в ресурсонезависимую программу на языке Colamo;
- процессор «Прокруст», масштабирующий ресурсонезависимую программу на языке Colamo (адаптирующий параметры программы под архитектуру РВС);
- процессор «Щелкунчик», осуществляющий редукцию производительности программы при нехватке аппаратного ресурса.

Ранее разработанные транслятор языка программирования COLAMO и синтезатор многокристальных решений Fire!Constructor транслируют полученную программу на COLAMO и создают многокристальное схемотехническое решение – VHDL-файлы для всех используемых при реализации задачи кристаллов ПЛИС РВС.

Рис. 2. Структурная схема комплекса средств трансляции последовательных программ на языке С в конфигурационные файлы ПЛИС

Синтез загрузочных конфигурационных файлов (*.bit) осуществляется синтезатором системы автоматизированного проектирования Xilinx Vivado для каждого кристалла в отдельности.

Основные этапы трансляции последовательной программы на языке С в параллельную программу на COLAMO. Рассмотрим основные этапы преобразования программы компонентами комплекса.

1. Построение информационного графа задачи. Исходная программа на языке С транслятором «Ангел» преобразуется в абсолютно-параллельную форму – информационный граф – совокупность множеств входных, выходных и операционных вершин задачи, соединенных между собой информационными связями (дугами). Он не содержит циклов, все операционные вершины задачи представлены заданное размерностью задачи число раз.

Информационный граф задачи (ИГЗ) описывается конструкциями языка программирования COLAMO – все массивы исходной последовательной программы становятся мемориальными массивами с параллельным (векторным) типом доступа [2] по всем измерениям. Циклы исходной программы задают слои и итерации для подграфов, являющихся телом цикла. В слоях расположены изоморфные функционально-завершенные информационно независимые подграфы, а итерации характеризуют информационную зависимость данных во времени. Подграфы, расположенные в одном слое, информационно независимы (между ними отсутствуют дуги), а подграфы, расположенные на разных итерациях, зависимы по обрабатываемым данным во времени. На основе структуры исходной последовательной программы (циклов, функций и процедур) ИГЗ представляется в виде объединения подзадач, в каждой из которых выделяется базовый подграф. В базовом подграфе каждой подзадачи выделяются один или несколько функциональных подграфов (ФП) – циклов исходной программы – фрагментов вычислений с заданными функциями масштабирования по слоям и итерациям. Так, для последовательной программы решения систем линейных алгебраических уравнений (СЛАУ) методом Гаусса для обусловленной матрицы можно выделить две подзадачи со своими базовыми подграфами: прямой и обратный ход. Прямой ход, как правило, представляется тройным циклом, а обратный ход – одним циклом по числу строк матрицы. Для БП прямого хода функциональными подграфами будут операторы расчета нормировочного коэффициента строки и вычитание произведения нормировочного множителя и элемента ведущей строки из элемента столбца матрицы. Число итераций циклов по каждой переменной соответствует числу слоев и итераций.

2. Анализ структуры информационного графа задачи (информационных зависимостей исходной программы). На этом этапе транслятор «Ангел» преобразует последовательную программу с произвольным обращением к памяти к параллельной программе, оперирующей потоками данных. Разделение по слоям и итерациям выполняется на основе анализа циклов [12–17] с учетом информационных зависимостей переменных и массивов данных. Для этого в выделенных функциональных подграфах анализируются информационные зависимости и проверяется соблюдение правил единственной подстановки [18] и однократного присваивания, нарушения которых устраняются с помощью расщепление скалярных переменных и растягивание массивов по итерациям [19]. После выполнения всех преобразований создается синтаксически корректная программа на языке COLAMO в абсолютно-параллельной форме с выделенными разметкой **#FuncGraph** функциональными подграфами вычислений.

3. Преобразование задачи в масштабируемую параллельно-конвейерную форму. Масштабируемая параллельно-конвейерная форма, преобразование к которой выполняется процессором «Русалка», позволяет с помощью одной константы (степени параллелизма – числа одновременно реализованных подграфов) автоматически пересчитывать длину обрабатываемых потоков данных для каждого из реализованных подграфов. Для этого каждое измерение всех массивов программы должно быть представлено двумя взаимосвязанными составляющими – векторной и потоковой с параллельным и последовательным типами доступа соответственно. Поэтому все созданные транслятором «Ангел» векторные измерения массивов дополняются связанной с ними потоковой составляющей таким образом, чтобы произведение векторной и потоковой размерностей было в точности равно длине исходного массива в последовательной программе. Также в тексте программы на COLAMO добавляются циклы для обработки потоковой составляющей, границы которых связываются с векторными циклами аналогичным образом. В результате

этого преобразования абсолютно-параллельная программа на языке COLAMO становится параллельно-конвейерной, в которой можно управлять параллелизмом одним параметром с автоматическим пересчетом длины потоковой составляющей и соблюдением синтаксической корректности.

4. Масштабирование задачи. Расчет параметров масштабирования по аппаратному ресурсу и числу каналов памяти, редукция производительности функциональных подграфов задачи и согласование интенсивности потоков данных выполняются процессором «Прокруст» (и, при необходимости, процессором «Щелкунчик», которому отводится вспомогательная роль) и являются, пожалуй, самой сложной частью рассматриваемой методики.

Процессор «Прокруст» рассчитывает и подбирает параметры рациональной реализации всех функциональных подграфов задачи, после чего преобразует масштабируемую параллельно-конвейерную форму с помощью методов редукции производительности. Теоретические положения редукции производительности подробно рассмотрены в [11], а алгоритм синтеза рациональной вычислительной структуры ИГЗ для заданной РВС – в [10].

Для сбалансированного масштабирования и редукции производительности задачи необходимо обеспечить сокращение производительности всех подзадач в одинаковое число раз, заданное коэффициентом редукции производительности. Наиболее трудоемким этапом масштабирования является согласование темпов обработки данных во всех функциональных подграфах ИГЗ и выбор для каждого функционального подграфа наиболее рациональной формы организации вычислений с учетом вычислительной структуры других подграфов и задачи в целом. Для решения этой задачи разработан ряд правил, позволяющих сократить число анализируемых вариантов и общее время решения задачи.

Масштабирование и редукция производительности информационного графа задачи начинается с «флагмана» – ФП, который занимает наибольший аппаратный ресурс (произведение занимаемого телом цикла ресурса и числа итераций цикла). Остальные ФП будем считать «катерами» – сравнительно небольшими по аппаратным затратам ФП подзадачи/задачи, занимающими существенно меньший по сравнению с «флагманом» ресурс. Как правило, «флагманом» является цикл с наибольшим числом итераций и/или с самым «тяжеловесным» телом: аппаратные затраты на реализацию «флагмана» существенно (не менее, чем на один десятичный порядок) превышают аппаратные затраты на реализацию «катера». Рациональная аппаратная реализация «флагмана» крайне важна, т.к. именно «флагман» вносит наибольший вклад в общее время решения задачи, поэтому чем ближе его реализация к оптимальной, тем меньше время решения задачи в целом. Все остальные ФП («катера») при масштабировании согласовываются с ним по числу каналов памяти, аппаратным затратам и интенсивности потоков данных.

Прикладные задачи (рис. 3) содержат либо «флагман», либо «катер», а чаще всего – обе подзадачи, связанные различными типами информационной зависимости. При анализе аппаратных ресурсов ФП и базового подграфа целесообразно сравнивать их с минимальным ресурсом, который заведомо реализуется в одном кристалле ПЛИС, чтобы не масштабировать ФП методами редукции по числу устройств и разрядам. Если ФП помещается в заданный минимальный ресурс, то его можно реализовать структурно без масштабирования. В самом ФП, если он содержит несколько выражений (операторов), используется аналогичная методика – в первую очередь масштабируется (редуцируется) самое большое по занимаемому аппаратному ресурсу («тяжеловесное») выражение.

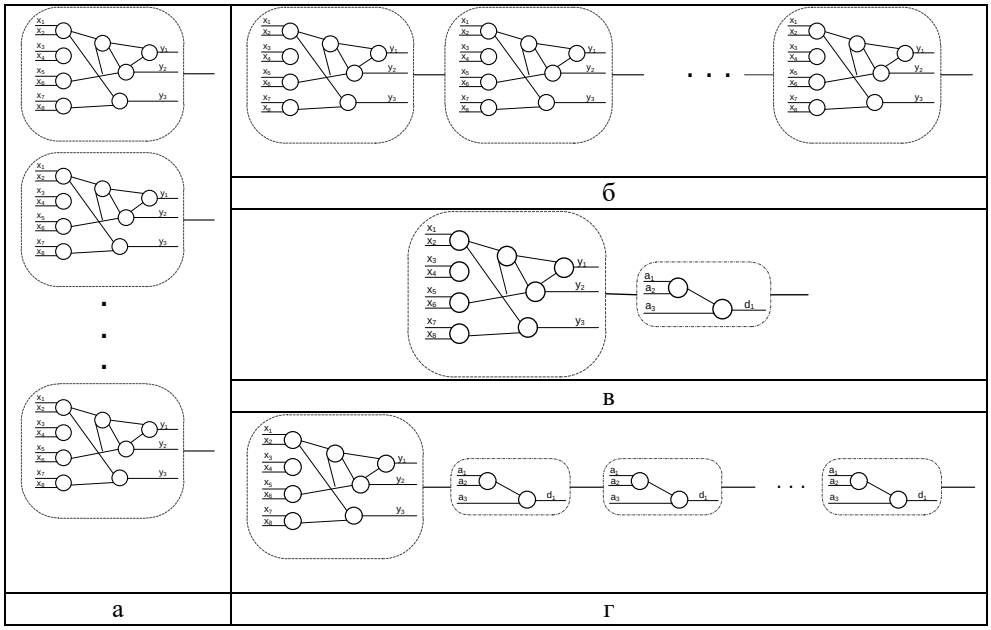


Рис. 3. Структуры информационных графов прикладных задач: а – информационно-независимые «флагманы»; б – информационно-зависимые «флагманы»; в – «флагман»-«катер»; г – «флагман»-множество «катеров»

Один из основных приемов при редукции «флагмана» – сохранить неизменным (нередуцируемым) итерационный цикл, поскольку это сокращает время решения задачи, не увеличивает число каналов памяти и задействует аппаратный ресурс, которого достаточно много. Если это невозможно, то реализуются несколько итерационных ступеней, связанных обратной связью, что приводит к существенному увеличению интервала обработки данных, который можно сократить с помощью оптимизационных преобразований (п.5). Основным нежелательным свойством роста интервала обработки данных является его влияние не только на один вычислительный фрагмент, а распространение на всю задачу, чтобы обеспечить единую интенсивность обработки данных.

«Катер» с маленьким весом (не превышающим 0,05) целесообразно реализовывать структурно, без применения методов редукции, сокращающих аппаратные затраты (по числу устройств и разрядности), поскольку такие ФП не вносят существенного вклада в превышение аппаратного ресурса задачи. Для достижения заданного коэффициента редукции производительности ФП с маленьким весом предпочтительнее использовать интервал обработки данных. При наличии информационной зависимости между слоями (в случае функционально-нерегулярного графа) необходимо редуцировать ФП до процедурной реализации.

Для того чтобы задача масштабировалась (то есть пропорционально увеличивалась или уменьшалась по всем подзадачам), необходимо обеспечить не только одинаковую величину редукции для всех подзадач, но и пропорциональное изменение интенсивности потоков данных между подзадачами. Применительно к редукции это означает, что (если это возможно) для сохранения интенсивности потоков данных целесообразно использовать одинаковые виды редукции с одинаковыми коэффициентами. При использовании разных видов редукции в разных подзадачах в общем случае требуется согласовать интенсивность потоков данных,

что, как правило, приводит к дополнительным аппаратным затратам (структурная реализация блоков согласования – элементы задержки с мультиплексорами/демультиплексорами, буфера, внутренняя двухпортовая память (BRAM)) и увеличивает время решения задачи.

Для согласования интенсивности потоков данных используются методы оптимизации, сокращающие увеличившийся в результате редукции интервал обработки данных (скважность) до сбалансированного по всем подграфам наименьшего значения (в наилучшем случае – до минимального значения, равного 1).

5. Оптимизация полученного решения. Сокращение интервала обработки данных задачи по критерию минимума времени решения выполняется с помощью преобразования к конвейеру конвейеров или макроконвейеру. Необходимость в сокращении интервала обработки данных возникает в ряде случаев, например, когда число реализованных итерационных ступеней меньше, чем требуется для реализации итерации целиком – в этом случае неизбежно возникает обратная связь и возрастает интервал обработки данных.

Интервал обработки данных при обратной связи равен отношению латентности реализованных итерационных ступеней к числу регистров в обратной связи. Поэтому, если невозможно увеличить число реализованных итерационных ступеней, для сокращения интервала обработки и создания плотного потока данных необходимо увеличить число регистров в обратной связи до значения латентности – в этом и состоит преобразование к конвейеру конвейеров (вложенному конвейеру, конвейеру в конвейере). В этом случае можно сократить интервал обработки данных до минимального значения, равного 1, что особенно важно при масштабировании «флагмана», вносящего наибольший вклад в общее время решения задачи. Преобразование к макроконвейеру аналогично по своей цели – сокращение интервала обработки данных, но применяется для процедурно реализованных фрагментов и состоит в увеличении их числа до значения, численно равного числу тактов работы одного процедурного устройства – в этом случае каждый такт будет освобождаться как минимум один блок для обработки очередной порции данных, которые подаются плотным потоком каждый такт, с единичным интервалом обработки данных.

Представленная система ограничений и оптимизаций позволяет синтезировать рациональную организацию вычислений (конвейер в конвейере, макроконвейер) для наиболее вычислительно-трудоемкого функционального подграфа «флагмана», что обеспечивает сокращение общего времени решения задачи, поскольку «катера» будут согласовываться именно с ним.

Результаты экспериментальных исследований разработанного комплекса. С помощью созданной экспериментальной версии комплекса средств трансляции успешно транслированы и реализованы на РВС «Терциус» [20, 21] описанные на языке С следующие прикладные программы линейной алгебры: решение систем линейных алгебраических уравнений (СЛАУ) методом Гаусса для матриц размером 8000×8000, решение СЛАУ методом Якоби для матриц размером 8000×8000 (табл. 1), решения СЛАУ с помощью верхней треугольной и нижней треугольной матриц (LU-разложение) для матриц размером 8000×8000.

Таблица 1

Число итерационных ступеней решения СЛАУ методом Якоби	Время решения		
	ПК	РВС	Ускорение
20 ступеней	0.00288	0.00268	1
200 ступеней	0.0292	0.00345	8,4
950 ступеней	0.2895	0.0056	51

Также с высокой эффективностью синтезированы конфигурационные файлы ПЛИС для ряда прикладных задач символьной обработки, использовавшихся в качестве тестов CHStone [4]: удельная производительность полученных решений составляет не менее 85 % от созданных прикладными программистами на языке COLAMO и не менее 70 % от прикладных решений, разработанных специалистами-схемотехниками. Время трансляции последовательных программ на языке С в программы на языке программирования высокого уровня COLAMO и синтеза конфигурационных файлов ПЛИС (без синтеза загрузочных конфигурационных bit-файлов) не превышает 30 минут.

Заключение. Разработанный комплекс средств трансляции реализует оригинальную методику преобразования последовательных программ на языке С в конфигурационные файлы ПЛИС на основе масштабирования информационного графа, описанного в синтаксисе языка COLAMO, для заданной многокристальной РВС. Масштабирование программы выполняется процессором «Прокруст» с помощью методов редукции производительности и аппаратных затрат, что позволяет синтезировать рациональную организацию вычислений и обеспечить сокращение общего времени решения задачи. Полученные при трансляции ряда прикладных задач экспериментальные результаты позволяют сделать вывод о том, что комплекс средств трансляции программ на языке С в программы на языке потока данных COLAMO позволяет существенно сократить время синтеза решения задачи для многокристальных РВС.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Воеводин В.В., Воеводин Вл.В.* Параллельные вычисления. – СПб.: БХВ-Петербург, 2002. – 608 с. – ISBN 5-94157-160-7.
2. *Гузик В.Ф., Каляев И.А., Левин И.И.* Реконфигурируемые вычислительные системы. – Таганрог: Изд-во ЮФУ, 2016. – 472 с.
3. *Trimberger S.M.* Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology // in Proceedings of the IEEE. – March 2015. – Vol. 103, No. 3. – P. 318-331. – Doi: 10.1109/JPROC.2015.2392104.
4. *Nane R. et al.* A Survey and Evaluation of FPGA High-Level Synthesis Tools // in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. – Oct. 2016. – Vol. 35, No. 10. – P. 1591-1604. – Doi: 10.1109/TCAD.2015.2513673.
5. *Nane R., Sima V.-M., Olivier B., Meeuws R., Yankova Y., Bertels K.* DWARV 2.0: A CoSy-based C-to-VHDL Hardware Compiler // In FPL. – 2012. – P. 619-622.
6. *Pilato C. and Ferrandi F.* Bambu: A Modular Framework for the High Level Synthesis of Memory-intensive Applications // In FPL. 2013. – P. 1-4.
7. *Canis A., Choi J., Aldham M., Zhang V., Kammoona A., Anderson J.H., Brown S., Czajkowski T.* LegUp: High-Level Synthesis for FPGA-based Processor/Accelerator Systems // In ACM FPGA. – 2011. – P. 33-36.
8. Make Slow Software Run Fast with Vivado HLS. – https://www.xilinx.com/publications/xcellonline/run-fast-with-Vivado-HLS.pdf.
9. Vitis Unified Software Platform Documentation. Application Acceleration Development. – https://www.xilinx.com/support/documentation/sw_manuals/xilinx2019_2/ug1393-vitis-application-acceleration.pdf (дата обращения: 10.11.2020).
10. *Levin Ilya, Dordopulo Alexey, Gudkov Vyacheslav, Gulenok Andrey, Bovkun Alexander, Yevstafiyev Georgyi, Alekseev Kirill.* Software Development Tools for FPGA-Based Reconfigurable Systems Programming // In: Communications in Computer and Information Science, Vol. 1129, Chapter Parallel Computing Technologies. – 2019. – P. 1-16.
11. *Левин И.И., Дордопуло А.И.* К вопросу об автоматическом создании параллельных прикладных программ для реконфигурируемых вычислительных систем // Вычислительные технологии. – 2020. – Т. 25, № 1. – С. 66-81.
12. *Morvan A., Derrien S. and Quinton P.* Efficient nested loop pipelining in high level synthesis using polyhedral bubble insertion // 2011 International Conference on Field-Programmable Technology, New Delhi, 2011. – P. 1-10. – Doi: 10.1109/FPT.2011.6132715.

13. *Jensen, Nicklas, Karlsson, Sven*. Improving Loop Dependence Analysis // ACM Transactions on Architecture and Code Optimization. – 2017. – Vol. 14 (3). – P. 1-24. – Doi: 10.1145/3095754.
14. *Solihin Yan*. Fundamentals of parallel computer architecture*:* multichip and multicore systems. – Chapman and Hall/CRC, 2016. – ISBN 978-1-4822-1118-4.
15. *Cooper, Keith D., Torczon, Linda*. Engineering a Compiler. – Morgan Kaufmann, 2005. – ISBN 1-55860-698-X.
16. *Kennedy Ken, Allen Randy*. Optimizing Compilers for Modern Architectures. A Dependence-based Approach. – Morgan Kaufmann, 2001. – ISBN 1-55860-286-0.
17. *Muchnick Steven S.* Advanced Compiler Design and Implementation. – Morgan Kaufmann, 1997. – ISBN 1-55860-320-4.
18. Векторизация программ // Векторизация программ: теория, методы, реализация: Сб. переводов статей. – М.: Мир, 1991. – С. 246-267.
19. Системы параллельной обработки: пер. с англ. / под ред. Д. Ивенса. – М.: Мир, 1985. – 416 с.
20. *Levin Ilya I. et al.* Reconfigurable computer systems: from the first FPGAs towards liquid cooling systems // Supercomputing Frontiers and Innovations. – 2016. – Vol. 3 (1). – P. 22-40. – Doi: 10.14529/jsfi160102.
21. *Kalyaev I.A., Levin I.I., Dordopulo A.I., Slasten L.M.* FPGA – based Reconfigurable Computer Systems // Proc. of 2013 Science and Information Conference SAI-2013- Oct 9, 2013, London, UK. – P. 148-155.

REFERENCES

1. *Voevodin V.V., Voevodin Vl.V.* Parallel'nye vychisleniya [Parallel computing]. Saint Petersburg: BKhV-Peterburg, 2002, 608 p. ISBN 5-94157-160-7.
2. *Guzik V.F., Kalyaev I.A., Levin I.I.* Rekonfiguriruemye vychislitel'nye sistemy [Reconfigurable computing systems]. Taganrog: Izd-vo YuFU, 2016, 472 p.
3. *Trimberger S.M*. Three Ages of FPGAs: A Retrospective on the First Thirty Years of FPGA Technology, *in Proceedings of the IEEE*, March 2015, Vol. 103, No. 3, pp. 318-331. Doi: 10.1109/JPROC.2015.2392104.
4. *Nane R. et al.* A Survey and Evaluation of FPGA High-Level Synthesis Tools, *in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Oct. 2016, Vol. 35, No. 10, pp. 1591-1604. Doi: 10.1109/TCAD.2015.2513673.
5. *Nane R., Sima V.-M., Olivier B., Meeuws R., Yankova Y., Bertels K.* DWARV 2.0: A CoSy-based C-to-VHDL Hardware Compiler, *In FPL*, 2012, pp. 619-622.
6. *Pilato C. and Ferrandi F*. Bambu: A Modular Framework for the High Level Synthesis of Memory-intensive Applications, *In FPL*, 2013, pp. 1-4.
7. *Canis A., Choi J., Aldham M., Zhang V., Kammoona A., Anderson J.H., Brown S., Czajkowski T.* LegUp: High-Level Synthesis for FPGA-based Processor/Accelerator Systems, *In ACM FPGA*, 2011, pp. 33-36.
8. Make Slow Software Run Fast with Vivado HLS. Available at: https://www.xilinx.com/ publications/xcellonline/run-fast-with-Vivado-HLS.pdf.
9. Vitis Unified Software Platform Documentation. Application Acceleration Development. Available at: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2019_2/ ug1393 -vitis-application-acceleration.pdf (accessed 10 November 2020).
10. *Levin Ilya, Dordopulo Alexey, Gudkov Vyacheslav, Gulenok Andrey, Bovkun Alexander, Yevstafiyev Georgyi, Alekseev Kirill.* Software Development Tools for FPGA-Based Reconfigurable Systems Programming, *In: Communications in Computer and Information Science, Vol. 1129, Chapter Parallel Computing Technologies*, 2019, pp. 1-16.
11. *Levin I.I., Dordopulo A.I.* K voprosu ob avtomaticheskom sozdanii parallel'nykh prikladnykh programm dlya rekonfiguriruemykh vychislitel'nykh sistem [On the issue of automatic creation of parallel application programs for reconfigurable computing systems], *Vychislitel'nye tekhnologii* [Computing technologies], 2020, Vol. 25, No. 1, pp. 66-81.
12. *Morvan A., Derrien S. and Quinton P*. Efficient nested loop pipelining in high level synthesis using polyhedral bubble insertion, *2011 International Conference on Field-Programmable Technology, New Delhi, 2011,* pp. 1-10. Doi: 10.1109/FPT.2011.6132715.
13. *Jensen, Nicklas, Karlsson, Sven*. Improving Loop Dependence Analysis, *ACM Transactions on Architecture and Code Optimization,* 2017, Vol. 14 (3), pp. 1-24. Doi: 10.1145/3095754.

14. *Solihin Yan.* Fundamentals of parallel computer architecture*:* multichip and multicore systems. Chapman and Hall/CRC, 2016. ISBN 978-1-4822-1118-4.
15. *Cooper, Keith D., Torczon, Linda.* Engineering a Compiler. Morgan Kaufmann, 2005. ISBN 1-55860-698-X.
16. *Kennedy Ken, Allen Randy.* Optimizing Compilers for Modern Architectures. A Dependence-based Approach. Morgan Kaufmann, 2001. ISBN 1-55860-286-0.
17. *Muchnick Steven S.* Advanced Compiler Design and Implementation. Morgan Kaufmann, 1997. ISBN 1-55860-320-4.
18. Vektorizatsiya programm [Vectorization of programs], *Vektorizatsiya programm: teoriya, metody, realizatsiya: Sb. perevodov statey* [Vectorization of programs: theory, methods, implementation: Collection of translations of articles]. Moscow: Mir, 1991, pp. 246-267.
19. Sistemy parallel'noy obrabotki [Parallel processing systems]: trans. from engl, ed. by D. Ivensa. Moscow: Mir, 1985, 416 p.
20. *Levin Ilya I. et al.* Reconfigurable computer systems: from the first FPGAs towards liquid cooling systems, *Supercomputing Frontiers and Innovations,* 2016, Vol. 3 (1), pp. 22-40. Doi: 10.14529/jsfi160102.
21. *Kalyaev I.A., Levin I.I., Dordopulo A.I., Slasten L.M.* FPGA – based Reconfigurable Computer Systems, *Proc. of 2013 Science and Information Conference SAI-2013- Oct 9, 2013, London, UK,* pp. 148-155.

Статью рекомендовал к опубликованию д.т.н. Э.В. Мельник.

Дордопуло Алексей Игоревич – Общество с ограниченной ответственностью «НИЦ супер-ЭВМ и нейрокомпьютеров»; e-mail: dordopulo@superevm.ru; 347935, г. Таганрог, 9-й переулок, д. 44; тел.: 88634477407; начальник отдела математико-алгоритмического обеспечения; к.т.н.

Гуленок Андрей Александрович – e-mail: andrei_gulenok@mail.ru; 347910, г. Таганрог, ул. 1-я Котельная, 71, кв. 301; тел.: +79085083496; начальник сектора; к.т.н.

Бовкун Александр Викторович – e-mail: simans2002@mail.ru; 347917, г. Таганрог, ул. Северная, 67; тел.: 88634477407; н.с., к.т.н.

Левин Илья Израилевич – Южный федеральный университет; e-mail: iilevin@sfedu.ru; 347928, г. Таганрог, ул. Петровская, 15, кв. 143; тел.: 88634612111; и.о. зав. кафедрой интеллектуальных и многопроцессорных систем; д.т.н.; профессор.

Гудков Вячеслав Александрович – e-mail: Vgudkov@sfedu.ru; 347905, г. Таганрог, ул. Дзержинского, 110; тел.: 88634477407; к.т.н.

Дудко Сергей Анатольевич – e-mail: dudko@sfedu.ru; 347900, г. Таганрог, пер. Тургеневский, 44; тел.: +79034318173; кафедра интеллектуальных и многопроцессорных систем; аспирант.

Dordopulo Alexey Igorevich – “Supercomputers and Neurocomputers Research Center” Co. Ltd.; e-mail: dordopulo@superevm.ru; 44, 9th lane, Taganrog, 347935, Russia; phone: +78634477407; head of the division of mathematic and algorithmic support; cand. of eng. sc.

Gulenok Andrey Aleksandrovich – e-mail: andrei_gulenok@mail.ru; 1st Kotelnaya 71, app. 301, Taganrog, 347910, Russia; phone +79085083496; head of sector; cand. of eng. sc.

Bovkun Aleksandr Victorovich – e-mail: simans2002@mail.ru; Severnaya 67, Taganrog, 347917, Russia; phone +78634477407; researcher; cand. of eng. sc.

Levin Ilya Izrailevich – Southern Federal University; e-mail: iilevin@sfedu.ru; 15, Petrovskaya str., ap. 143, Taganrog, 347928, Russia; phone: +78634612111; head of the department of intellectual and multiprocessor systems;dr. of eng. sc.; professor.

Gudkov Vyacheslav Aleksandrovich – e-mail: Vgudkov@sfedu.ru; Dzerzhinskogo str., ap. 110, Taganrog, 347905, Russia; phone: +78634477407; cand. of eng. sc.

Dudko Sergei Anatolievich – e-mail: dudko@sfedu.ru; 44, Turgenevskii lane, Taganrog, 347900, Russia; phone: +79034318173; the department of intellectual and multiprocessor systems; graduate student.

УДК 004.472.43 DOI 10.18522/2311-3103-2020-7-107-121

С.А. Дудко

ЭКВИВАЛЕНТНЫЕ ПРЕОБРАЗОВАНИЯ НЕКОТОРЫХ ВИДОВ РЕКУРСИВНЫХ НЕЛИНЕЙНЫХ ВЫЧИСЛИТЕЛЬНЫХ СТРУКТУР ДЛЯ ЭФФЕКТИВНОЙ РЕАЛИЗАЦИИ НА РЕКОНФИГУРИРУЕМЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

Рассматриваются методы информационно-эквивалентных преобразований некоторых видов нелинейных вычислительных структур с обратными связями: квадратичных, дробных и условных. Наличие обратных связей в конвейерной вычислительной структуре, решаемых на реконфигурируемых вычислительных системах прикладных задач, приводит к замедлению скорости формирования данных, так как для вычисления очередного значения требуется дождаться результата по обратной связи. При этом замедление происходит не только на участке с обратной связью, но и во всей вычислительной структуре, что приводит к увеличению времени, за которое данная задача может быть решена. Предыдущие фрагменты вынуждены задерживать свои данные перед подачей в обратную связь, а последующие вынуждены простаивать, ожидая данные на выходе обратной связи. На сегодняшний день не существует средств автоматического проектирования прикладных задач для реконфигурируемых вычислительных систем, которые оптимизировали бы такие вычислительные структуры в автоматическом режиме. Поэтому пользователь вынужден самостоятельно изучать текст исходной программы и искать в нем выражения, содержащие обратные связи, а затем оптимизировать их. Это приводит к увеличению времени, требующегося для создания эффективных прикладных программ. Предложенные методы преобразований позволяют сократить интервал обработки данных (в лучшем случае до единицы) при решении прикладных задач на реконфигурируемых вычислительных системах. Для реализации информационно-эквивалентных преобразований необходимо, чтобы в вычислительной системе имелся дополнительный аппаратный ресурс. Реализация данных преобразований в оптимизирующем синтезаторе схемотехнических решений позволяет проводить оптимизацию вычислительной структуры с обратными связями автоматически. Это позволяет сократить время разработки эффективных прикладных программ, содержащих обратные связи, с нескольких дней до нескольких минут.

Информационно-эквивалентные преобразования; оптимизирующий синтезатор; реконфигурируемые вычислительные системы; нелинейные вычислительные структуры.

S.A. Dudko

EQUIVALENT TRANSFORMATIONS FOR SOME KINDS OF RECURSIVE NON-LINEAR COMPUTING STRUCTURES FOR EFFICIENT IMPLEMENTATION ON RECONFIGURABLE COMPUTER SYSTEMS

In the paper, we consider data-equivalent transformations of some kinds of non-linear computing structures, such as quadratic, fractional and conditional. All computing structures contain feedbacks. If a pipeline computing structure of a task, implemented on a reconfigurable computer system, contains feedbacks, the data processing rate slows down, because it is necessary to wait for feedback results to calculate the next value. The processing rate slows down not only in the chain with feedback, but in the whole computing structure. As a result, the task solution time increases. Previous fragments have to delay their data to supply it into a chain with feedback, and subsequent ones have to remain idle waiting for the feedback result data. At present, there are no software development tools for reconfigurable computer systems with automatic optimization of such computing structures. So, the user has to analyze the source code to find expressions with feedbacks, and to optimize them. As a result, the development time of efficient applications considerably increases. We suggest methods decreasing the data processing time interval (down to unity in the best case) for applied tasks solved on reconfigurable com-

puter systems. Besides, the task solution time also decreases. Owing to the suggested methods, implemented in the optimizing synthesizer of circuit solutions, transformations are performed automatically. As a result, the development time for efficient applied tasks with feedbacks decreases from several days to several minutes.

Data-equivalent transformation; optimizing synthesizer; reconfigurable computer system; non-linear computing structure.

Введение. В настоящее время одной из проблем, снижающих производительность реконфигурируемых вычислительных систем (РВС) [1, 2], построенных на основе программируемых логических интегральных схем (ПЛИС) [3, 4], является большой интервал обработки данных в вычислительных структурах [5]. Чаще всего такая проблема возникает, когда в структуре решаемой задачи образуются обратные связи, обусловленные рекурсивными структурами самой задачи или методами организации вычислений.

Наличие рекурсивных вычислительных структур негативно сказывается на времени решения прикладной задачи, поскольку для начала очередной итерации вычислений необходимо дождаться формирования выходного сигнала в обратной связи. Это приводит к тому, что часть вычислительных ресурсов вынуждена простаивать в ожидании необходимых данных.

Пример фрагмента рекурсивной вычислительной структуры показан на рис. 1. Блок α является рекурсивным и требует для вычислений данные из блока G, а также предыдущее данное, сформированное блоком α. Блок F принимает выходные данные, полученные блоком α, и продолжает необходимые вычисления. Если блок α выполняет вычисления за S тактов, то это приводит к тому, что блок G вынужден подавать свои выходные данные на вход блока α раз в S тактов, соответственно и блок F будет получать данные для обработки один раз в S тактов. Если S>1, то это приведет к увеличению времени, требуемого на обработку всего потока данных и, соответственно, к увеличению времени решения задачи.

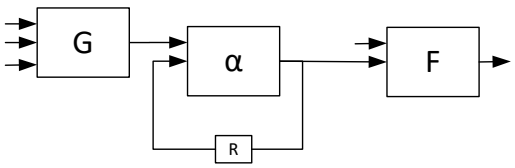


Рис. 1. Пример вычислительной структуры с обратной связью

В самом общем случае время, необходимое для решения задачи на конвейерной вычислительной структуре, при заданном аппаратном ресурсе может быть рассчитано по следующей формуле:

$$T_R \approx NS\tau,$$

где T_R – время решения задачи при доступном аппаратном ресурсе R, N – длина потока данных, S – интервал обработки данных, τ – длительность такта.

Соответственно, чтобы сократить время решения задачи, необходимо уменьшить значение одного из сомножителей данной формулы. Уменьшить количество обрабатываемых данных или длительность такта не представляется возможным, поэтому, единственным способом сокращения времени решения задачи является сокращение интервала обработки данных.

Существующие средства разработки программ для РВС [6–8], в лучшем случае могут подсказать пользователю, что в вычислительной структуре решаемой задачи обнаружены обратные связи, но провести оптимизацию подобных структур

они не способны. Пользователь вынужден самостоятельно искать образовавшиеся обратные связи в тексте исходной программы и оптимизировать их. При этом задача поиска обратных связей в исходном тексте является достаточно трудоемкой, так как структуры, образующие обратную связь, могут располагаться как в различных частях одного файла с текстом исходной программы, так и в различных связанных файлах.

За счет этого существенно возрастает время создания эффективных прикладных программ на РВС, так как даже опытному пользователю будет сложно отыскать и эффективно оптимизировать все обратные связи (особенно при их большом количестве).

Для того чтобы сократить время разработки эффективных прикладных программ, необходимо автоматизировать процедуру оптимизации обратных связей в вычислительной структуре задачи. Под оптимизацией обратных связей будем понимать процесс уменьшения интервала обработки данных (в наилучшем случае до единицы, S=1) по сравнению с интервалом исходной задачи.

Поиск и оптимизацию рекурсивных структур будем приводить не в тексте исходной программы, а в его промежуточном представлении в виде плоской вычислительной структуры, что позволит быстрее находить и обрабатывать обратные связи, так как все вершины, образующие обратную связь, располагаются последовательно друг за другом.

В работе [9] были рассмотрены методы преобразования линейных вычислительных структур с обратными связями. Настоящая статья содержит методы преобразования некоторых видов нелинейных вычислительных структур, таких как квадратичные, дробные и условные.

Преобразования квадратичных вычислительных структур. Одним из примеров нелинейных вычислительных структур [10, 11] являются квадратичные структуры, образующиеся при вычислении выражений, аналогичных квадратным уравнениям. Пример квадратного уравнения может быть представлен в следующем виде:

$$y_i = y_{i-1}^2 * \mathrm{a}_i + b_i .$$

Обозначим операцию «умножения» как β, а операцию «сложения» как φ. Тогда соответствующая данному уравнению вычислительная структура будет представлена на рис. 2.

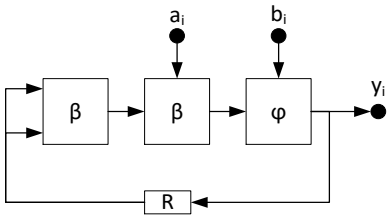


Рис. 2. Квадратичная вычислительная структура

Следует отметить, что операционные вершины β и φ могут представлять собой любые алгебраические операции, выполняемые над любыми множествами, образующими кольцо относительно данных операций [12, 13].

При наличии обратных связей интервал обработки данных может быть рассчитан по следующей формуле:

$$S = \frac{\sum L}{R}, \quad (1)$$

где L – латентность отдельных операционных вершин в пути обратной связи, R – количество регистров в обратной связи. Путь – последовательность вершин, образующая обратную связь.

Будем считать, что латентность операционных вершин β и φ равна единице. Тогда интервал обработки данных, в соответствии с формулой (1), будет равен 3. Отметим, что данная обратная связь состоит из двух аналогичных путей (так как левая вершина β имеет два входа, соответствующих выходу обратной связи), что усложняет применение информационно-эквивалентных преобразований.

Для того чтобы уменьшить интервал обработки данных в подобных структурах, воспользуемся методом автоподстановки [9]. Для этого развернем исходную обратную связь на 1 шаг (рис. 3,а), что приведет к росту необходимого для реализации аппаратного ресурса, но при этом позволит установить в обратную связь дополнительный регистр. Опираясь на эквивалентные преобразования дистрибутивных и ассоциативных операционных вершин [9], преобразуем вычислительную структуру, показанную на рис. 3,а.

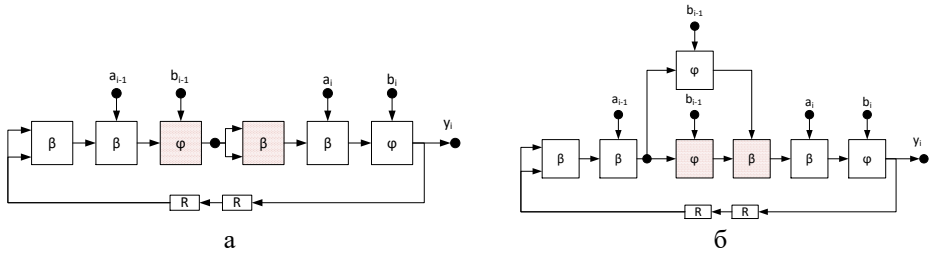


Рис. 3. Эквивалентные преобразования: а – преобразование развертки обратной связи на 1 шаг; б – дублирование операционной вершины для удаления ветвления

Как можно заметить, в обратной связи имеется операционная вершина, выходной сигнал которой ветвится на два (сигнал между заштрихованными вершинами на рис. 3,а). Данное ветвление не позволяет произвести преобразование дистрибутивных операционных вершин, поэтому необходимо избавиться от данного ветвления с помощью метода дублирования вычислений (рис. 3,б). Применение данного метода приводит к тому, что для реализации вычислительной структуры потребуется дополнительный аппаратный ресурс, необходимый на реализацию продублированной вершины. Далее, для заштрихованных на рис. 3,б вершин воспользуемся преобразованием дистрибутивных операционных вершин, после чего получим вычислительную структуру, показанную на рис. 4,а. Как можно заметить, в результате преобразования дистрибутивных операционных вершин появилась новая операционная вершина, требующая дополнительного аппаратного ресурса. После данных преобразований интервал обработки данных не уменьшился (7/2 в соответствии с формулой 1), а количество занимаемого аппаратного ресурса возросло, поэтому необходимо продолжить дальнейшие преобразования.

Повторно избавимся от образовавшихся ветвлений между заштрихованными вершинами, показанными на рис. 4,а. Полученная после дублирования операционных вершин вычислительная структура показана на рис. 4,б.

Далее продолжим выполнять преобразования дистрибутивных и ассоциативных операционных вершин до тех пор, пока это возможно.

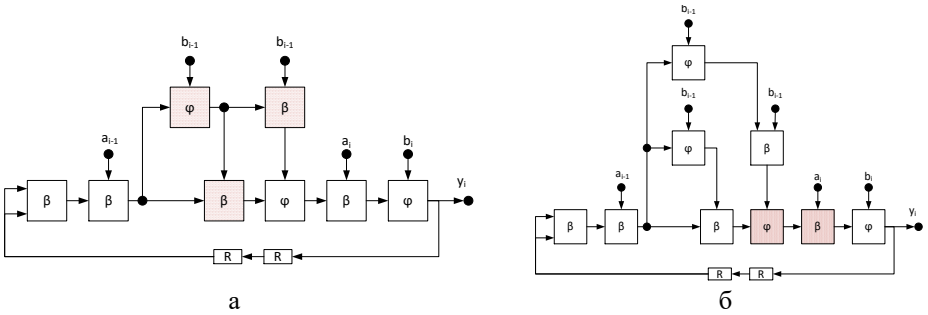


Рис. 4. Применение эквивалентных преобразований: а – преобразование дистрибутивных операционных вершин; б – дублирование операционной вершины для удаления ветвления

Полученная в итоге вычислительная структура (рис. 5), не может быть преобразована дальше с помощью дистрибутивных или ассоциативных преобразований. При этом интервал обработки данных (6/2) равен интервалу исходной вычислительной структуры, показанной на рис. 2. Поэтому для дальнейших оптимизаций можно использовать преобразование, которое будем называть обратным дистрибутивным преобразованием. Его схема показана на рис. 6.

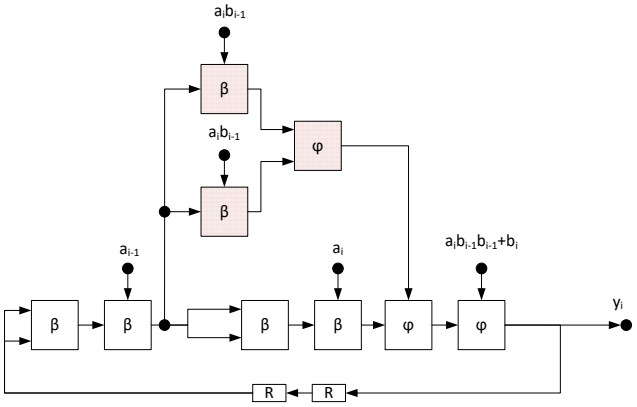


Рис. 5. Вычислительная структура после цепочки преобразований дистрибутивных и ассоциативных операционных вершин

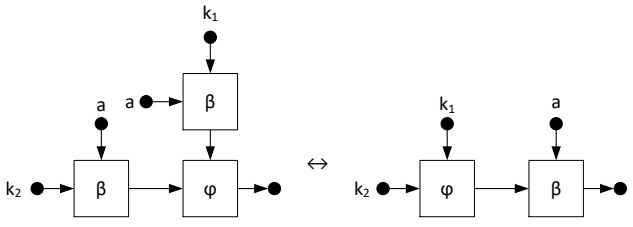


Рис. 6. Обратное преобразование дистрибутивных вершин

Если входные данные k_1 и k_2 равны между собой, то можно воспользоваться преобразованием ассоциативных вершин с общим входным операндом (рис. 7) и продолжить оптимизацию исходной вычислительной структуры.

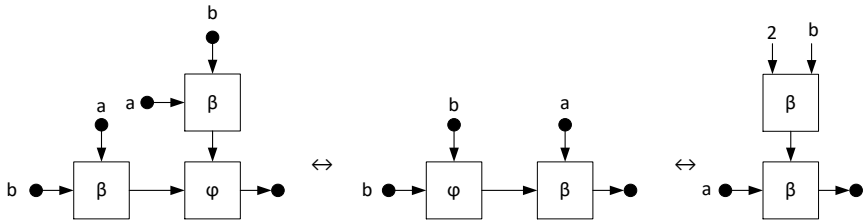
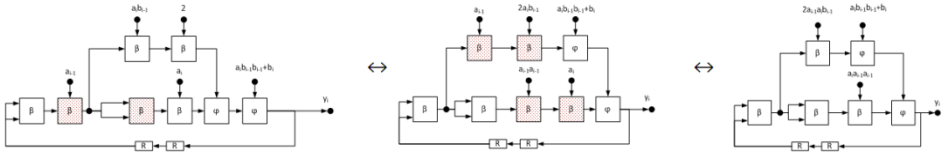


Рис. 7. Обратное преобразования дистрибутивных вершин и преобразование ассоциативных вершин с общим входным операндом

Воспользовавшись описанными выше преобразованиями, можем изменить вычислительную структуру, показанную на рис. 5, а затем применить преобразования дистрибутивных и ассоциативных операционных вершин, для того чтобы вынести часть вычислений за пределы обратной связи и, тем самым, уменьшив длину пути по обратной связи (рис. 8).



а б в

Рис. 8. Вычислительная структура после применения преобразования ассоциативных вершин

На первом шаге (рис. 8,а) необходимо избавиться от ветвления, продублировав левую заштрихованную операционную вершину β, после чего применить ассоциативное преобразование для заштрихованных вершин. На втором шаге (рис. 8,б) применяются ассоциативные преобразования для двух групп вершин, после чего получаем итоговую вычислительную структуру (рис. 8,в).

После применения всех эквивалентных преобразований получим вычислительную структуру, показанную на рис. 8,в. Интервал обработки данных в соответствии с формулой (1) для данной вычислительной структуры будет равен S = 2 (4 вершины / 2 регистра), что меньше исходного интервала обработки данных в обратной связи (S = 3) в 1,5 раза. Для того чтобы добиться более плотного потока данных, и, тем самым, повысить скорость решения задачи, можно воспользоваться методом автоподстановки повторно.

Преобразования условных вычислительных структур. При решении прикладных задач часто порядок выполнения операций зависит от выполнения каких-либо условий [14]. Для управления порядком выполнения операций применяются условные операторы. В языках программирования высокого уровня таким оператором является IF…THEN…ELSE или его аналоги [15]. При этом условные операторы также могут образовывать рекурсивные выражения. В условных рекурсивных выражениях выходное значение сигнала зависит от выполнения некоторого условия. В схемотехнике для реализации условных операторов используются мультиплексоры [16, 17]. Соответствующий данной программе фрагмент вычислительной структуры будет выглядеть следующим образом (рис. 9), где сигнал CE – какое-либо условие, отвечающее за выбор одной из ветвей вычислений:

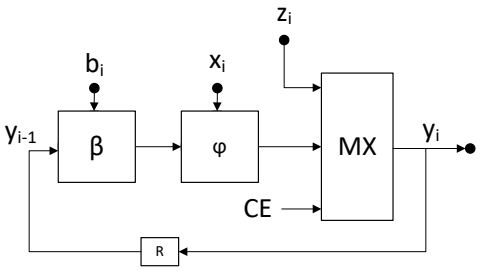


Рис. 9. Пример условной обратной связи

Мультиплексор отвечает за выбор той или иной ветви вычислений, результат которой образует обратную связь. Преобразования подобной конструкции не могут быть проведены с использованием метода автоподстановки, так как мультиплексор не обладает необходимыми свойствами дистрибутивности и ассоциативности. Поэтому для того чтобы применить к подобным структурам метод автоподстановки, необходимо заменить мультиплексор в обратной связи на набор других операционных вершин.

В общем случае для замены мультиплексора могут быть использованы логические операции «OR» и «AND» над битовым представлением данных [19, 18]. Но так как данные операции не удовлетворяют свойствам ассоциативности и дистрибутивности совместно с другими операциями в обратной связи, целесообразно заменить мультиплексор тем же набором операций, что уже входят в обратную связь (рис. 10). Данное преобразование основывается на свойствах теории групп [20].

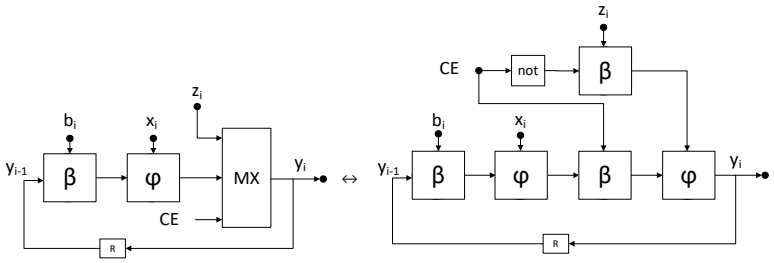


Рис. 10. Замена мультиплексора набором дистрибутивных и ассоциативных операционных вершин

Для корректности данного преобразования необходимо, чтобы множество K, над которым определены операции β и φ образовывало алгебраическое кольцо [12, 13].

По определению кольца в нем должны быть определены нейтральной «ноль» (n^0) для операции φ и нейтральная «единица» (n^1) для операции β, такие, что

$$a\ \varphi\ n^0 = n^0\ \varphi\ a = a; \tag{2}$$

$$x\ \beta\ n^1 = n^1\ \beta\ x = x; \tag{3}$$

$$a\ \beta\ n^0 = n^0\ \beta\ a = n^0. \tag{4}$$

Свойство (4) называется мультипликативным свойством нуля [20].

Преобразование замены мультиплексора набором дистрибутивных вершин (рис. 10) возможно, так как при выполнении операции β с «единицей» (n^1) полученное данное будет пропущено далее без изменений в соответствии со свойством (3). А при выполнении операции β с «нулем» (n^0) данные будут сброшены к n^0 в

соответствии со свойством (4). В итоге при выполнении операции φ с каким-либо числом и n^0 данное число будет без изменений преобразовано в выходное значение в соответствии с (2).

После замены мультиплексора набором дистрибутивных операционных вершин необходимо воспользоваться эквивалентными преобразованиями дистрибутивных и ассоциативных операционных вершин для того, чтобы упростить полученную вычислительную структуру. Данные преобразования показаны на рис. 11, где вершина $\overline{CE}$ является результатом выполнения операции «NOT» над вершиной CE. На первом шаге к заштрихованным вершинам применяется преобразование дистрибутивности, а затем на втором шаге - преобразование ассоциативности.

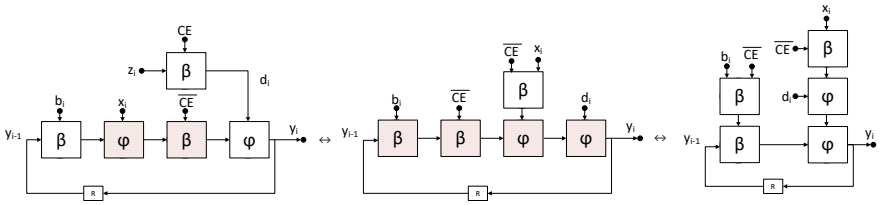


Рис. 11. Цепочка преобразований дистрибутивных и ассоциативных вершин

После этого к упрощенной вычислительной структуре для уменьшения интервала обработки данных может быть применено преобразование автоподстановки, которое было рассмотрено ранее.

Таким образом, замена мультиплексора набором дистрибутивных операционных вершин позволяет преобразовать исходную вычислительную структуру, к которой затем могут быть применены методы эквивалентных преобразований, направленные на снижение интервала обработки данных.

Эквивалентные преобразования над прямыми и обратными операционными вершинами. Перед тем как перейти к рассмотрению следующего типа нелинейных вычислительных структур, необходимо рассмотреть набор базовых эквивалентных преобразований над обратными операционными вершинами (вершинами, соответствующими обратным к β и φ операциям).

Пусть определено множество G, в котором выполняются операции β и φ. Для проведения рассматриваемых далее преобразований необходимо, чтобы данное множество образовывало алгебраическое поле [13, 20], т.е. должны выполняться следующие условия:

1) Для операций β и φ существуют нейтральные элементы n^1 и n^0 соответственно.

2) Для каждого элемента x из множества G существует обратный элемент $\bar{x}$.

3) Для операций β и φ определены обратные операции β^{-1} и φ^{-1} соответственно, такие что

$$\bar{x} = n^1\,\beta^{-1}\,x;$$
$$\bar{x} = n^0\,\varphi^{-1}\,x.$$

Если все условия выполняются, то к вычислительной структуре могут быть применены следующие эквивалентные преобразования.

Эквивалентное преобразование замены обратной операционной вершины β^{-1} прямой операционной вершиной β (рис. 12,а) осуществляется путем инвертирования входного оператора на одном из входов (в зависимости от типа дистрибутивности обратной операционной вершины). Блок «1/x» обозначает операцию получения обратного операнда.

При выполнении прямой операции с прямым и обратным операндами в результате образуется нейтральный элемент, который позволяет исключить операционную вершину из вычислительной структуры задачи (рис. 12,б).

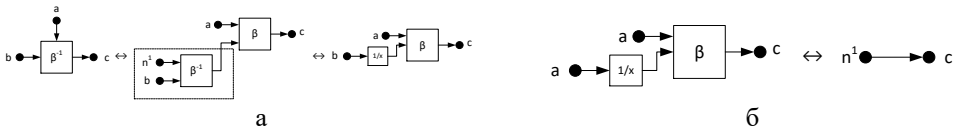


Рис. 12. Эквивалентные преобразования с обратными вершинами: а – замена обратной операционной вершины; б – сокращение прямого и обратного операндов

Если операция получения обратного значения выполняется два раза подряд, то в результате входной операнд не будет изменен. Это позволяет исключить операционные вершины получения обратного значения из вычислительной структуры (рис. 13,а).

Если на выходе операционной вершины расположена операция получения обратного значения, то она может быть перенесена на все входы данной операционной вершины. Верно и обратное: если на всех входах вершины расположены операции получения обратного значения, то они могут быть заменены одной операцией получения обратного значения на выходе данной операционной вершины (рис. 13,б).

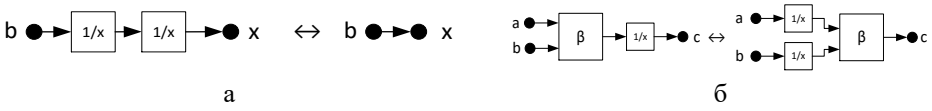


Рис. 13. Эквивалентные преобразования: а – преобразование нескольких операций взятия обратного значения; б – перенос операции получения обратного значения через вершину

Для прямой и обратной операционных вершин также определены преобразование ассоциативных вершин (рис. 14,а) и преобразования дистрибутивных вершин (рис. 14,б). В некоторых случаях дистрибутивность может быть односторонней. Тогда порядок следования операндов будет играть важное значение, как, например, в случае с операцией «деление», которая является дистрибутивной справа.

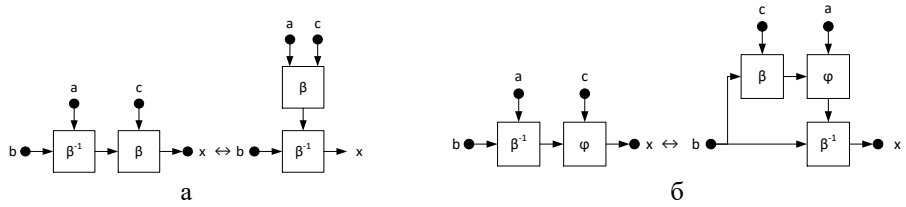


Рис. 14. Эквивалентные преобразования: а – преобразование ассоциативных обратных операционных вершин; б – преобразование дистрибутивных обратных операционных вершин

Далее рассмотрим эквивалентное преобразование «пирамиды» обратных операционных вершин с общим входным операндом, показанное на рис. 15.

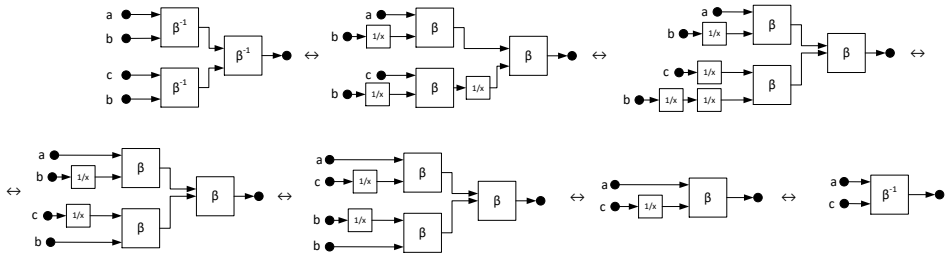


Рис. 15. Преобразование пирамиды обратных операционных вершин с общим входным операндом

Данное преобразование позволяет заменить пирамиду обратных операционных вершин одной обратной операционной вершиной путем применения преобразований, рассмотренных ранее. Данную цепочку преобразований будем называть преобразованием «пирамиды» обратных операционных вершин с одним общим входным операндом.

Использование данных эквивалентных преобразований позволяет в рассматриваемом далее типе нелинейных вычислительных структур избавиться от обратных операционных вершин и упростить структуру задачи.

Преобразования дробных вычислительных структур. Еще одним примером нелинейных вычислительных структур являются «дробные» вычислительные структуры. В ряде случаев подобные структуры могут быть оптимизированы. Для наглядности изложения рассмотрим данное преобразование для операций «+», «*» и «/», которые в общем случае соответствуют операциям «φ», «β» и «β^{-1}». Рассмотрим преобразования подобных структур на примере следующего нелинейного уравнения:

$$y_i = \frac{y_{i-1} * a_i + b_i}{y_{i-1} * c_i + d_i}.$$

Соответствующая данному уравнению вычислительная структура представлена на рис. 16.

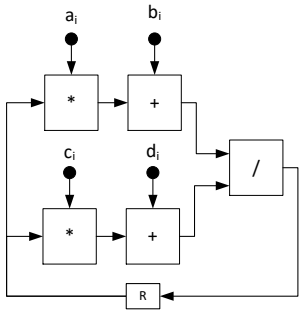
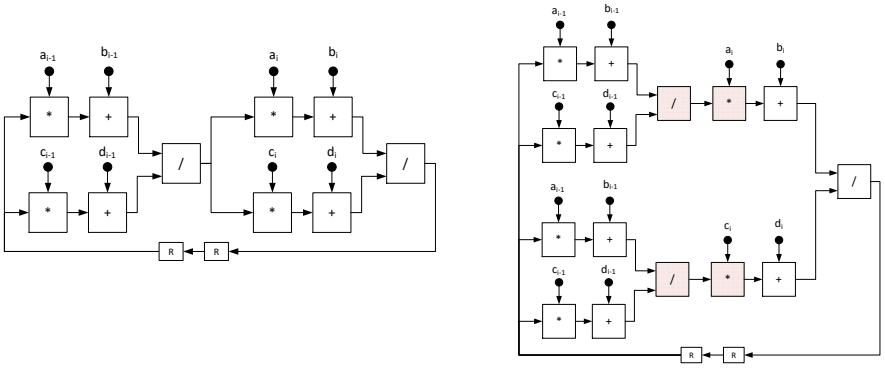


Рис. 16. Исходная нелинейная дробная вычислительная структура

Как можно заметить, в данной вычислительной структуре определена операция «деления», которая является обратной к операции умножения.

Опираясь на преобразования обратных операционных вершин, преобразуем исходную вычислительную структуру. Считая, что латентность каждой отдельной операционной вершины равна 1, получим, что интервал обработки данных в обратной связи равен 3. Попробуем оптимизировать данную вычислительную структуру, применив к ней метод автоподстановки. Развернем исходную вычислитель-

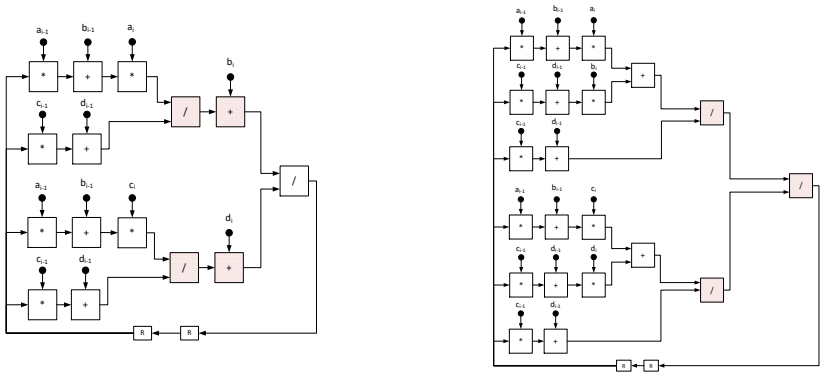
ную структуру на 1 шаг (рис. 17,а), при этом затратив дополнительный аппаратный ресурс и добавив в обратную связь дополнительный регистр. Затем избавимся от ветвлений, применив метод дублирования вычислений, получив вычислительную структуру, показанную на рис. 17,б. Интервал обработки данных для получения на рис. 17,б структуры остался без изменений (6/2 = 3).



а б

Рис. 17. Применение эквивалентных преобразований: а – преобразование автоподстановки на 1 шаг; б – преобразование дублирования вычислений

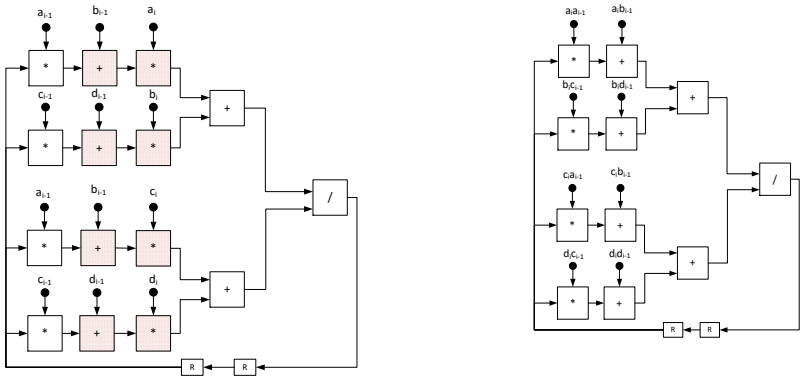
Далее, опираясь на преобразования с прямыми и обратными операционными вершинами, применим для заштрихованных вершин (рис. 17,б) преобразование ассоциативных обратных операционных вершин (рис. 18,а), а затем преобразование дистрибутивных обратных операционных вершин (рис. 18,б). Интервал обработки данных для получения на рис. 18,б структуры остался без изменений (6/2 = 3), при этом наблюдается значительный рост аппаратных затрат.



а б

Рис. 18. Вычислительная структура после применения эквивалентных преобразований: а – после преобразования ассоциативных обратных вершин; б – после преобразования дистрибутивных обратных вершин

Далее воспользуемся эквивалентным преобразованием «пирамиды» обратных операционных вершин (рис. 19,а) и преобразованием дистрибутивных операционных вершин (рис. 19,б):



а б

Рис. 19. Вычислительная структура после применения эквивалентных преобразований: а – после преобразования пирамиды обратных операционных вершин с одним общим входным операндом; б – после преобразования дистрибутивных операционных вершин

Полученная в итоге вычислительная структура обладает меньшим, чем исходная, интервалом обработки данных S = 2 (4/2), что позволяет получить результат вычислений за меньшее количество тактов. Для того чтобы добиться более плотного потока данных, метод автоподстановки может быть применен повторно, до тех пор, пока не будет получен необходимый интервал обработки данных.

Дополнительные аппаратные затраты, необходимые для преобразования нелинейных вычислительных структур, включают в себя дополнительные вершины, получаемые в ходе применения преобразований дистрибутивных операционных вершин, а также ресурс, получаемый при дублировании вычислений. В наиболее общем случае количество необходимого дополнительного аппаратного ресурса может быть рассчитано по следующей формуле:

$$\frac{N^2+N}{2} + \sum_{i=1}^{S}(k-1), \quad (5)$$

где N – исходное количество дистрибутивных операционных вершин, S –интервал обработки данных, k – количество операционных вершин в исходной обратной связи. Сумма соответствует количеству элементов, образованных в результате ветвлений.

При этом к вновь образованным ветвям также могут быть применены дистрибутивные преобразования. Поэтому применение подробных методов целесообразно в случаях, когда другие методы преобразований не подошли, и имеется достаточно высокий запас аппаратных ресурсов.

Заключение. Предложенные методы преобразования нелинейных рекурсивных выражений позволяют без участия пользователя оптимизировать фрагменты вычислительной структуры с высоким интервалом обработки данных и уменьшить время решения прикладной задачи.

Данные преобразования требуют наличия дополнительного аппаратного ресурса, что ограничивает сферу их применения. Метод автоподстановки может быть применен тогда, когда другие методы оптимизация достигли предела критического ресурса (например, при распараллеливании по итерациям закончились каналы ПЛИС).

Разработанные преобразования могут быть применены к различным типам вычислительных структур, таким, как квадратичные, дробные, а также условные. Отличительной особенностью разработанных методов является их применение для информационно-вычислительной структуры задачи, а не для текста исходной программы.

Реализация данных преобразований в оптимизирующем синтезаторе схемотехнических решений позволит проводить все преобразования в автоматическом режиме без участия пользователя и сократить срок разработки эффективных прикладных программ, содержащих обратные связи, с нескольких дней до нескольких минут.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Гузик В.Ф., Каляев И.А., Левин И.И.* Реконфигурируемые вычислительные системы: учеб. пособие / под общ. ред. И.А. Каляева. – Ростов-на-Дону: Изд-во ЮФУ, 2016. – 472 с. – ISBN 978-5-9275-1980-7.
2. *Compton K.* Reconfigurable Computing: A Survey of Systems and Software // ACM Computing Surveys. – 2002. – Vol. 34, No. 2. – P. 171-210.
3. *Попов А.Ю.* Проектирование цифровых устройств с использованием ПЛИС: учеб. пособие. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2009. – 80 с.
4. *Krishna G, Sahadev R.* Fundamentals of FPGA Architecture // Advanced Engineering Technical and Scientific Publisher. – 2017. – Part 2. – P. 12-30.
5. *Каляев А.В.* Модульно-наращиваемые многопроцессорные системы со структурно-процедурной организацией вычислений. – М.: Янус-К, 2003. – 380 с.
6. Intel® Quartus® Prime Standard Edition User Guide 18.1. Getting Started. UG-20173 2018.09.24. – P. 44-47. – URL: https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/ug/archives/ug-qps-getting-started-18-1.pdf (дата обращения: 01.10.2020).
7. Xilinx Vivado Design Suite. User Guide. Synthesis. UG901 (v2017.1) April 19, 2017. – P. 7-38. – URL: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2017_1/ug901-vivado-synthesis.pdf (дата обращения: 25.09.2020).
8. Synopsys Identify Microsemi Edition Instrumentor User Guide, January 2018. – P. 50-51. – URL: https://www.microsemi.com/document-portal/doc_download/136672-synopsys-identify-rtl-l2016-09m-2-debugger-instrumentor-for-libero-soc-v11-8 (дата обращения: 28.09.2020).
9. *Дудко С.А.* Метод преобразования рекуррентных выражений в информационном графе // XVI Ежегодная молодежная научная конференция «Юг России: вызовы времени, открытия, перспективы»: материалы конференции (г. Ростов-на-Дону, 13–28 апреля 2020 г.). – Ростов-на-Дону: Изд-во ЮНЦ РАН, 2020. – 168 с.
10. *Агапова Е.Г.* Вычислительная математика: учеб. пособие / под ред. Т.М. Попова. 2017. – Хабаровск: Изд-во Тихоокеан. гос. ун-та, 2017. – 92 с.
11. *Lindenhovius B., Mislove M., Zamdzhiev V.* Mixed linear and non-linear recursive types // Proceedings of the ACM on Programming Languages. – 2019. – Vol. 3, Article 111.
12. *Васильев А.В., Мазуров В.Д.* Высшая алгебра: В 2 ч. // Конспект лекций. – Новосибирск: Изд-во Новосиб. гос. ун-т., 2010. – Ч. 1. – 143 с.
13. *Aditya R., Zulfikar M.T., Manik N.I.* Testing Division Rings and Fields Using a Computer Program // Procedia Computer Science. – 2015. – Vol. 59. – P. 540-549.
14. *Тюгашев А.А.* Основы программирования. Ч. I. – СПб.: Университет ИТМО, 2016. – 160 с.
15. *Nielsen F.* A Concise and Practical Introduction to Programming Algorithms in Java, Undergraduate Topics in Computer Science. – Springer-Verlag London Limited, 2009.
16. *Акчурин А.Д., Юсупов К.М.* Программирование на языке Verilog: учеб. пособие. – Казань, 2016. – 90 с.
17. *Nabulsi M., Al-Husainy M.* Using Combinational Circuits for Control Purposes // Journal of Computer Science. – 2009. – No. 5 (7). – P. 507-510.
18. *Wang X.* Estimation of Number of Bits in Binary Representation of an Integer // International Journal of Research Studies in Computer Science and Engineering. – 2015. – Vol. 2. – P. 28-31.
19. *Харрис Д.М., Харрис С.Л.* Цифровая схемотехника и архитектура компьютера. – 2-е изд., ДМК-Пресс, 2018. – 792 с.
20. *Воеводин В.В.* Линейная алгебра. – 2-е изд. – М.: Главная редакция физико-математической литературы, 1980.

REFERENCES

1. *Guzik V.F., Kalyaev I.A., Levin I.I.* Rekonfiguriruemye vychislitel'nye sistemy: ucheb. posobie [Reconfigurable computer systems: a tutorial], under the general ed. I.A. Kalyaeva, Rostov-on-Don: Izd-vo YuFU, 2016, 472 p. ISBN 978-5-9275-1980-7.
2. *Compton K.* Reconfigurable Computing: A Survey of Systems and Software, *ACM Computing Surveys,* 2002, Vol. 34, No. 2, pp. 171-210.
3. *Popov A.Yu.* Proektirovanie tsifrovykh ustroystv s ispol'zovaniem PLIS: ucheb. posobie [Designing digital devices using FPGAs: a tutorial]. Moscow: Izd-vo MGTU im. N.E. Baumana, 2009, 80 p.
4. *Krishna G, Sahadev R.* Fundamentals of FPGA Architecture, *Advanced Engineering Technical and Scientific Publisher,* 2017, Part 2, pp. 12-30.
5. *Kalyaev A.V.* Modul'no-narashchivaemye mnogoprotsessornye sistemy so strukturno-protsedurnoy organizatsiey vychisleniy [Modular-scalable multiprocessor systems with structural and procedural organization of calculations]. Moscow: Yanus-K, 2003, 380 p.
6. Intel® Quartus® Prime Standard Edition User Guide 18.1. Getting Started. UG-20173 2018.09.24, pp. 44-47. Available at: https://www.intel.com/content/dam/www/programmable/us/ en/pdfs/literature/ug/archives/ug-qps-getting-started-18-1.pdf (accessed 01 October 2020).
7. Xilinx Vivado Design Suite. User Guide. Synthesis. UG901 (v2017.1) April 19, 2017. – P. 7-38. Available at: https://www.xilinx.com/support/documentation/sw_manuals/xilinx2017_1/ug901-vivado-synthesis.pdf (accessed 25 September 2020).
8. Synopsys Identify Microsemi Edition Instrumentor User Guide, January 2018. – P. 50-51. Available at: https://www.microsemi.com/document-portal/doc_download/136672-synopsys-identify-rtl-l2016-09m-2-debugger-instrumentor-for-libero-soc-v11-8 (accessed 28 September 2020).
9. *Dudko S.A.* Metod preobrazovaniya rekurrentnykh vyrazheniy v informatsionnom grafe [Transforming method of recursive expressions in an information graph], *XVI Ezhegodnaya molodezhnaya nauchnaya konferentsiya «Yug Rossii: vyzovy vremeni, otkrytiya, perspektivy»: materialy konferentsii (g. Rostov-na-Donu, 13–28 aprelya 2020 g.)* [XVI Annual Youth Scientific Conference "South of Russia: Challenges of Time, Discoveries, Prospects": conference proceedings (Rostov-on-Don, April 13-28, 2020)]. Rostov-on-Don: Izd-vo YuNTS RAN, 2020, 168 p.
10. *Agapova E.G.* Vychislitel'naya matematika: ucheb. posobie [Computational mathematics: a tutorial], ed. by T.M. Popova. 2017. Khabarovsk: Izd-vo Tikhookean. gos. un-ta, 2017, 92 p.
11. *Lindenhovius B., Mislove M., Zamdzhiev V.* Mixed linear and non-linear recursive types, *Proceedings of the ACM on Programming Languages,* 2019, Vol. 3, Article 111.
12. *Vasil'ev A.V., Mazurov V.D.* Vysshaya algebra: V 2 ch. [Abstract Algebra: In 2 parts], *Konspekt lektsiy* [Lecture notes]. Novosibirsk: Izd-vo Novosib. gos. un-t., 2010, Part 1, 143 p.
13. *Aditya R., Zulfikar M.T., Manik N.I.* Testing Division Rings and Fields Using a Computer Program, *Procedia Computer Science,* 2015, Vol. 59, pp. 540-549.
14. *Tyugashev A.A.* Osnovy programmirovaniya [Basics of programming]. Part I. Saint Petersburg: Universitet ITMO, 2016, 160 p.
15. *Nielsen F.* A Concise and Practical Introduction to Programming Algorithms in Java, Undergraduate Topics in Computer Science. Springer-Verlag London Limited, 2009.
16. *Akchurin A.D., Yusupov K.M.* Programmirovanie na yazyke Verilog: ucheb. posobie [Verilog Programming: a tutorial]. Kazan', 2016, 90 p.
17. *Nabulsi M., Al-Husainy M.* Using Combinational Circuits for Control Purposes, *Journal of Computer Science,* 2009, No. 5 (7), pp. 507-510.
18. *Wang X.* Estimation of Number of Bits in Binary Representation of an Integer, *International Journal of Research Studies in Computer Science and Engineering,* 2015, Vol. 2, pp. 28-31.
19. *Kharris D.M., Kharris S.L.* Tsifrovaya skhemotekhnika i arkhitektura komp'yutera [Digital Design and Computer Architecture]. 2nd ed., DMK-Press, 2018, 792 p.
20. *Voevodin V.V.* Lineynaya algebra [Linear Algebra]. 2nd ed. Moscow: Glavnaya redaktsiya fiziko-matematicheskoy literatury, 1980.

Статью рекомендовал к опубликованию д.т.н. Э.В. Мельник.

Дудко Сергей Анатольевич – Южный федеральный университет; e-mail: dudko@sfedu.ru; 347900, г. Таганрог, пер. Тургеневский, 44; тел.: +79034318173; Кафедра интеллектуальных и многопроцессорных систем; аспирант.

Dudko Sergei Anatolievich – Southern Federal University, e-mail: dudko@sfedu.ru; 44, Turgenevskii lane, Taganrog, 347900, Russia; phone: +79034318173; the department of intellectual and multiprocessor systems; graduate student.

УДК 004.382.2 DOI 10.18522/2311-3103-2020-7-121-129

А.В. Касаркин

МЕТОД РЕШЕНИЯ ГРАФОВЫХ NP-ПОЛНЫХ ЗАДАЧ НА РЕКОНФИГУРИРУЕМЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ НА ОСНОВЕ ПРИНЦИПА РАСПАРАЛЛЕЛИВАНИЯ ПО ИТЕРАЦИЯМ

При решении графовых NP-полных задач на многопроцессорных системах рост оборудования не приводит к пропорциональному росту производительности системы, поэтому не всегда удается решить задачу за приемлемое время. Целью работы, описанной в статье, является минимизация времени решения задачи поиска максимальных клик графа с использованием реконфигурируемых вычислительных систем (РВС). При решении задачи на РВС методом распараллеливания по слоям рост производительности также замедляется, несмотря на лучшую степень масштабируемости по сравнению с многопроцессорными реализациями. В статье предложен метод создания параллельно-конвейерных программ для реконфигурируемых вычислительных систем на основе распараллеливания по итерациям для решения графовых NP-полных задач. Рассмотрено, что использовать битовый способ представления множеств (как в методе распараллеливания по слоям) для метода распараллеливания по итерациям не является эффективным. Новый метод отличается организацией вычислений, а именно – обработкой неупорядоченных множеств, доступ к элементам которых осуществляется не по адресам (как в массивах), а по значениям (именам вершин и именам дуг графа). Показано, что новый метод на основе распараллеливания по итерациям, несмотря на более низкую удельную производительность, связанную с тем, что вычислительным подструктурам из-за символьного представления множеств необходимо обработать большее число промежуточных данных, обеспечивает практически линейный рост реальной производительности РВС при значительно большем количестве вычислительного ресурса по сравнению с методом распараллеливания по слоям.

Теория множеств; графовые NP-полные задачи; задача о клике; максимальные клики графа; реконфигурируемые вычислительные системы; программируемые логические интегральные схемы (ПЛИС); информационный граф; структурная реализация; конвейер; суперкомпьютеры.

A.V. Kasarkin

A METHOD FOR SOLVING GRAPH NP-COMPLETE TASKS ON RECONFIGURABLE COMPUTER SYSTEMS BASED ON THE ITERATION PARALLELIZING PRINCIPLE

When we solve graph NP-complete tasks on multiprocessor systems, the growth of hardware resource does not lead to the proportional increase of the system performance, and hence, the task solution time is not always reasonable. The aim of our research, given in the paper, is minimization of the solution time of the task of maximal clique enumeration on reconfigurable computer systems (RCS). When we solve tasks on RCSs with the help of the method of parallelizing by layers, the growth of performance also slows down in spite of better scalability in comparison with multiprocessor implementations. In the paper, we suggest a method of parallel-pipeline application development for reconfigurable computer systems. The method is based on parallelizing by

layers for graph NP-complete tasks. We show that the bit representation of sets, which is used for the method of parallelizing by layers, is not efficient for the method of parallelizing by iterations. The new method has another organization of calculations; it processes unordered sets, whose elements are accessed not by addresses (as in arrays), but by values (names of vertices and names of edges of the graph). We show that the new method, based on parallelizing by iterations, provides ramping of the RCS real performance at much larger computational resource in comparison with the method of parallelizing by layers. Its specific performance is lower, because computing substructures are to process more intermediate data due to symbolic representation of sets.

Theory of sets; graph NP-complete tasks; clique task; maximal cliques of a graph; reconfigurable computer system; FPGA; information graph; structural implementation; pipeline; supercomputer.

Введение. Теория графов позволяет решать актуальные на сегодняшний день задачи из таких областей, как анализ социальных и вычислительных сетей, биоинформатика, логистика и многих других [1, 2]. Особое место занимают графовые задачи класса NP-полные, которые зачастую приходится решать быстрыми, но приближенными алгоритмами из-за длительного времени точного решения. Однако приближенное решение, в отличие от точных методов, не всегда приводит к оптимальному результату. В настоящее время одним из основных точных методов решения NP-полных задач является метод ветвей и границ [3]. Отличие данного метода от полного перебора заключается в исключении из обработки подмножеств возможных решений, которые заведомо не приведут к оптимальному результату. Тем не менее вычислительная сложность алгоритмов, основанных на методе ветвей и границ для решения NP-полных задач, остается экспоненциальной.

Поэтому графовые NP-полные задачи (далее – GNPC-задачи) являются наиболее трудновычислимыми. Кроме того, эти задачи характеризуются превалированием количества информационных обменов над количеством вычислений и требуют таких режимов доступа к памяти, которые приводят к нелинейной адресации и большому временному интервалу между повторными чтениями одних и тех же ячеек памяти (плохая пространственно-временная локализация приложений).

При применении стандартных методов решения GNPC-задач на многопроцессорных системах возникает ряд проблем, связанных с их масштабируемостью [4], которые приводят к тому, что при увеличении числа процессоров рост производительности замедляется, а при достижении некоторого их числа существенно замедляется [5-10]. На рис. 1 в качестве примера приведены графики, иллюстрирующие ускорение решения при распараллеливании задачи поиска максимальных клик [11] в зависимости от числа используемых ядер процессора [9, 10].

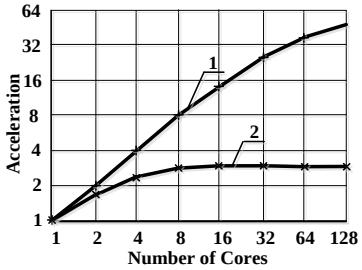


Рис. 1. Графики ускорения времени решения в зависимости от числа используемых ядер процессора: 1 – для графа с 300-ми вершинами из набора тестов «dimacs» (phat300-2) [12]; 2 – для графа Муна–Мозера [13] с 51-й вершиной

Замедление роста производительности связано с тем, что при превышении некоторого числа процессорных ядер затраты на организацию межпроцессорного обмена начинают превышать затраты на собственно вычисления, а транзакционная организация вычислений не позволяет начать выполнять новую операцию, пока процессор не завершит предыдущую операцию.

Для эффективного обмена данными между узлами обработки необходимо использовать вычислительную архитектуру, которая аппаратно поддерживает пространственную коммутацию. Такую возможность обеспечивают реконфигурируемые вычислительные системы (РВС) [14,15], при этом для решения на РВС задачу необходимо представить в виде информационного графа, определяющего вычислительную структуру параллельной программы и правила подачи данных на вход вычислительной структуры [16].

Важно отметить, что информационный граф GNPC-задач при решении методом ветвей и границ определяется не только алгоритмом обработки, но и промежуточными результатами вычислений: структура последующих слоев информационного графа определяется результатами вычислений на предыдущих слоях, то есть информационный граф является функционально нерегулярным. Поэтому для вычислительной структуры GNPC-задачи на РВС необходимо информационный граф привести к функционально-регулярному виду. Одним из способов приведения информационного графа к функционально регулярному виду является его дополнение до регулярной формы [16]. Однако граф, полученный таким способом, является функционально избыточным, что приводит к низкой удельной производительности. Более эффективным способом решения данной проблемы является процедура векторизации [16]. Применение данной процедуры в сочетании с модернизациями, направленными на перераспределение данных в процессе вычислений, как было показано в работах [17, 18], позволило разработать метод распараллеливания по слоям, который сократил время решения задачи поиска максимальных клик на РВС по сравнению с классической многопроцессорной системой. Однако с помощью программы имитационного моделирования вычислительных структур для решения на РВС графовых задач было установлено, что, несмотря на то что метод распараллеливания по слоям имеет в несколько раз лучшую степень масштабируемости по сравнению с методами для многопроцессорных систем, ему присущи те же проблемы – замедление роста производительности при превышении эффективного числа устройств (от 8 до 150 ПЛИС в зависимости от решаемого графа). В то же время существуют РВС [19], которые содержат в себе намного больший вычислительный ресурс, чем тот, при котором метод распараллеливания по слоям обеспечивает окололинейный рост производительности. Поэтому необходимо дальнейшее развитие методов синтеза вычислительной структуры для решения GNPC-задач на РВС.

Метод синтеза вычислительной структуры с распараллеливанием по итерациям для решения GNPC-задач на реконфигурируемых вычислительных системах. Еще одним эффективным методом синтеза вычислительной структуры является метод, в котором используется распараллеливание по итерациям [20]. При этом данные потоком проходят через вычислительную структуру.

Для приведения информационного графа GNPC-задачи к функционально-регулярному виду, позволяющему обеспечить синтез вычислительной структуры с распараллеливанием по итерациям, необходимо выделить базовый подграф [16], а затем информационный граф разделить на слои. При этом каждый слой содержит некоторое количество целых базовых подграфов. Затем в каждом слое все базовые подграфы заменяются одной вершиной. На рис. 2 слои разделены пунктирными эллипсами, а вершины $P_1 \ldots P_{31}$ являются базовыми подграфами.

В результате описанных действий информационный граф преобразуется в функционально-регулярную структуру, обеспечивающую распараллеливание по итерациям, в которой вершины последовательно соединены друг с другом информационными каналами.

На следующем шаге необходимо определиться с форматом обрабатываемых данных. При позиционном способе кодирования, в частности, при битовом способе представления множеств, операции, выполняемые над множествами, являются транзакционными, что приводит к простоям конвейера.

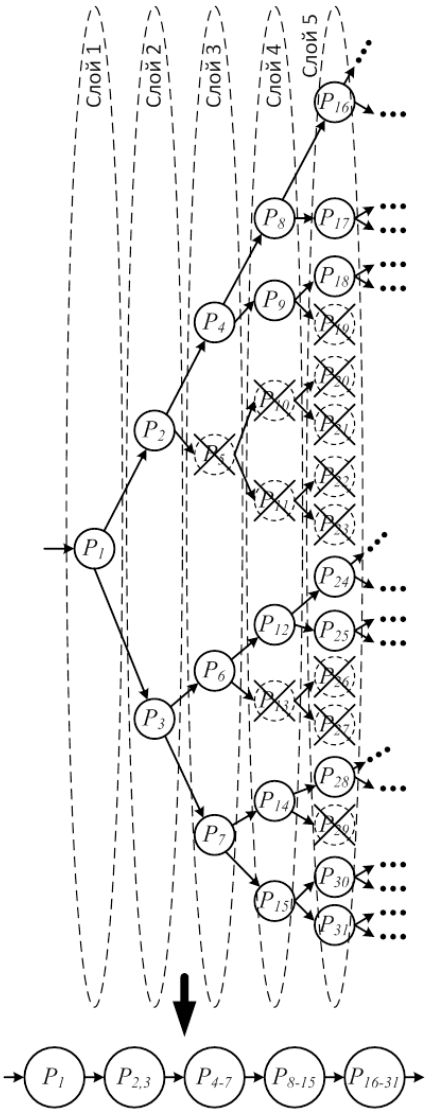


Рис. 2. Информационный граф GNPC-задач с группировкой вершин по порядку выполнения итераций

Поясним это на примере выполнения операции копирования, которая присутствует в алгоритме задачи поиска максимальных клик графа: скопировать любой элемент из множества K в переменную v. В процессе выполнения алгоритма

из-за удаления на каждой итерации элементов из множества K возникает ситуация, когда из множества K в переменную v необходимо скопировать последний оставшийся элемент множества, то есть K={k5}.

В битовом представлении позиция бита соответствует «номеру» вершины решаемого графа, а его значение является признаком присутствия либо отсутствия вершины во множестве. При битовом представлении множество можно сравнить со структурой данных «битовый массив», в которой каждая ячейка имеет свой определенный адрес, и этот адрес и является номером элемента множества. Для выполнения операции копирования элемента множества *K* в переменную *v* необходимо последовательно проверять элементы, начиная с первого элемента, и до нахождения элемента, который присутствует во множестве, таким образом, при битовой реализации результат можно получить только после проверки последнего элемента: k5. Это значит, что при битовом представлении множеств описанная операция является транзакционной: невозможно начать выполнять следующие операции, пока не будет проверен целиком весь массив – все множество K.

При символьном представлении множеств сам элемент множества является значением, элементы множества не упорядочены, и доступ к таким элементам осуществляется не по адресу, а по значению. Также при символьном представлении множество содержит имена только присутствующих в нем элементов, значит, для выполнения операции копирования элемента множества *K* в переменную *v* достаточно дождаться первого элемента множества *K* из потока и скопировать его в переменную *v*. Таким образом, для метода распараллеливания по итерациям битовое представление множеств вершин неэффективно и проигрывает символьному представлению, в котором каждому элементу множества соответствует некоторый символ или натуральное число.

При представлении множеств в символьном виде не представляется возможным провести векторизацию информационного графа для объединения базовых подграфов по слоям, так как генератор адресов не сможет адресоваться к конкретному элементу множества. Это связано с тем, что элементы во множествах располагаются в произвольном порядке, поэтому генератор адресов, который определяет порядок обработки промежуточных данных после проведения процедуры векторизации, не сможет вызвать нужный элемент, не перебрав множество полностью. Другой проблемой является то, что множества имеют переменную длину: длина меняется после прохождения каждой итерации, то есть генератору адресов нужно было бы решать проблему отделения одного множества в потоке от другого, не допускать перемешивания разных множеств при отправке их в конвейер.

Для решения указанной проблемы данные необходимо сгруппировать по множествам, чтобы в дальнейшем генератор адресов мог адресоваться целиком к вызываемому множеству. Таким образом, формируются потоки, состоящие из неупорядоченных множеств различной длины.

Полученная таким образом вычислительная структура представлена на рис. 3. Здесь $P_1 \ldots P_S$ – ступени вычислительного конвейера, каждая из которых соответствует одной итерации алгоритма; $y_1 \ldots y_S$ и $x_1 \ldots x_S$ – каналы, по которым последовательно передаются элементы множеств, а сами множества – параллельно (например, набор множеств K, M и P для задачи поиска максимальных клик графа). Блоки BUF представляют собой память, которая предназначена для хранения промежуточных множеств – результатов обработки итерации. При возникновении запроса BUF передает накопленное множество целиком. Вызов нужного элемента из прочитанного из BUF множества осуществляется в вычислительных конвейерах P по значениям – именам элементов множества.

Обработку одного элемента множества можно сравнить с работой ассоциативной памяти: поиск вершины графа осуществляется не по адресу, а по имени. Однако, в отличие от ассоциативной памяти, такой поиск реализуется не во времени, а в пространстве за счет непрерывного потока имён вершин через вычислительные блоки. В результате этого по структуре движется поток имен – элементов множеств, при этом множества имеют переменную длину.

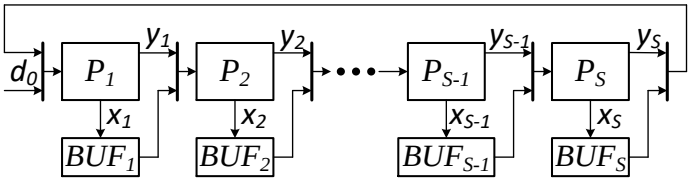


Рис. 3. Вычислительная структура, обеспечивающая распараллеливание по итерациям GNPC-задач

Поясним работу вычислительной структуры, представленной на рис. 5. В начале обработки на вход d_0 вычислительной структуры последовательно поступает начальный набор множеств. В процессе обработки через некоторое время, соответствующее латентности, конвейер P_1 начинает генерировать два новых набора множеств: первый набор - по шине y_1 последовательно, элемент за элементом, поступает на вход конвейера P_2, а второй - по шине x_1 последовательно запоминается в буфере BUF_1. После того как набор из P_1 полностью поступил в блок итерации P_2, на вход данной итерации подается набор множеств, который ранее запомнился в буфере BUF_1. Таким образом, на каждой итерации один из генерируемых наборов множеств идет на следующую итерацию, а второй попадает в буфер, где ожидает своей очереди для отправки на вход следующей итерации.

Оценка эффективности метода синтеза вычислительной структуры с распараллеливанием по итерациям для решения GNPC-задач на реконфигурируемых вычислительных системах. Для проверки работоспособности и эффективности предложенного метода была создана имитационная модель задачи поиска максимальных клик графа, моделирующая выполнение алгоритма таким образом, как он будет исполняться на РВС. Графики, отображающие результаты моделирования при решении графа на 500 вершин с вероятностью наличия ребра 0,6 на ПЛИС XCKU095, представлены на рис. 4 [21].

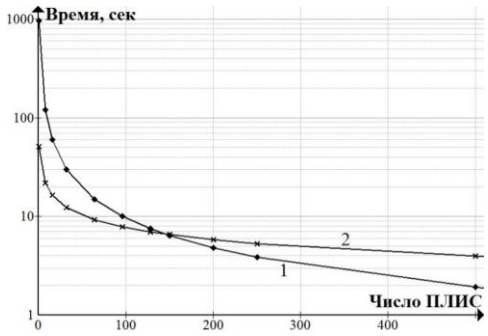


Рис. 4. График времени решения при распараллеливании задачи поиска максимальных клик: 1 – при использовании метода распараллеливания по слоям; 2 – при использовании метода распараллеливания по итерациям

На графиках заметно, что при превышении 145, ПЛИС программа, созданная с помощью метода распараллеливания по итерациям, решит задачу поиска максимальных клик в случайном графе на 500 вершин с вероятностью наличия ребра 0,6 быстрее, чем программа, созданная с помощью метода распараллеливания по слоям. Удельная производительность метода распараллеливания по итерациям при небольшом числе ПЛИС ниже из-за того, что при представлении множеств в символьном виде требуется обработать большее число промежуточных данных. В то же время ускорение метода распараллеливания по итерациям при увеличении вычислительного ресурса близко к линейному за счет значительного уменьшения коммутационных связей, что и обуславливает выигрыш метода при использовании более чем 145 ПЛИС.

Заключение. Новый метод распараллеливания по итерациям для синтеза параллельно-конвейерных программ для РВС позволяет эффективно задействовать намного больший объем вычислительного ресурса по сравнению с методом распараллеливания по слоям. В частности, при обработке графа на 4000 вершин с вероятностью ребра 0,5 время решения задачи поиска максимальных клик по сравнению с методом распараллеливания по слоям сократилось в 10 раз. В то же время, как показано в работах [17, 18], метод распараллеливания по слоям быстрее реализаций для многопроцессорных систем. Таким образом, новый метод распараллеливания по итерациям позволяет обрабатывать графовые задачи большей размерности за приемлемое время и становится эффективней метода распараллеливания по слоям при увеличении количества доступного вычислительного ресурса и размерности решаемых графов.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Кирьянов А.Г., Ляхов А.И., Некрасов П.О., Платов Д.А. и др.* Протокол многоадресной маршрутизации Proximity-based Groupcast in MANET (GiM) // Информационные процессы. – 2012. – № 3. – Т. 12. – С. 213-228.
2. *Курейчик В.М., Глушань В.М., Щербаков Л.И.* Комбинаторные аппаратные модели и алгоритмы в САПР. – М.: Радио и связь, 1990. – 216 с.
3. *Jens Clausen.* Branch and bound algorithms principles and examples. Department of comp sc., university of Copenhagen, Universitetsparken 1, DK-2100 Corpenhagen, Denmark.(March 12, 1999).
4. *Кирюшин Н.К., Михалев И.В.* Использование многоядерных ускорителей для решения задачи пропозициональной выполнимости // Проблемы науки. – 2017. – № 22 (104).
5. *Dasari N.S., Ranjan D., Mohammad Z.* Maximal Clique Enumeration for Large Graphs on Hadoop Framework // Proc. of the First Workshop on Parallel Programming for Analytics Applications. – 2014. – P. 21-30. – Doi: 10.1145/2567634.2567640.
6. *Rossi R.A., Gleich D.F., Gebremedhin A.H.* Parallel Maximum clique Algorithms with Applications to Network Analysis // SIAM J. Sci. Comput. – 2015. – Vol. 37, No. 5. – P. 589-616. – Doi: 10.1137/14100018X.
7. *Kovács L., Szabó G.* Conceptualization with Incremental Bron-Kerbosch Algorithm in Big Data Architecture // Acta Polytechnica Hungarica. – 2016. – Vol. 13, No. 2. – P. 139-158. – Doi: 10.12700/aph.13.2.2016.2.8.
8. *Pattabiraman B. et al.* Fast Algorithms for the Maximum Clique Problem on Massive Graphs with Applications to Overlapping Community Detection // Internet Mathematics. – 2015. – Vol. 11, No. 4-5. – P. 421-448. – Doi: 10.1080/15427951.2014.986778.
9. *Dasari N.S., Zubair M., Ranjan D.* A Novel Parallel Algorithm for Maximal Clique Enumeration on Multicore and Distributed Memory Architectures. – 10 p. – URL: https://pdfs.semanticscholar.org/9827/9e2cedb14085886fcb4473f1ba483a3df195.pdf (дата обращения: 23.09.2020).
10. *Dasari N.S., Desh Z.M.* pbitMCE: A bit-based approach for maximal clique enumeration on multicore processors // in: Proc. 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS 2014). – 2014. – P. 478-485. – DOI: 10.1109/PADSW.2014.7097844.

11. *Bron C., Kerbosch J.* Algorithm 457: Finding All Cliques of an Undirected Graph // Comm of ACM. – 1973. – Vol. 16. – P. 575-577.
12. *Johnson D.J. and Trick M.A.* Eds., Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, Workshop, October 11-13, 1993. Boston, MA, USA: American Mathematical Society, 1996.
13. *Moon J. and Moser L.* On cliques in graphs // Israel Journal of Mathematics. – 1965. – Vol. 3, No. 1. – P. 23-28. Available: Doi: 10.1007/BF02760024.
14. *Каляев И.А. и др.* Реконфигурируемые мультиконвейерные вычислительные структуры / под общ. ред. И.А. Каляева. – 2-е изд., перераб. и доп. – Ростов-на-Дону: Изд-во ЮНЦ РАН, 2009. – 344 с.
15. Научно-исследовательский центр супер-ЭВМ и нейрокомпьютеров. – http://superevm.ru (дата обращения: 23.09.2020).
16. *Каляев А.В., Левин И.И.* Модульно-наращиваемые многопроцессорные системы со структурно-процедурной организацией вычислений. – М.: Янус-К, 2003. – 380 с.
17. *Касаркин А.В., Левин И.И.* Структурная реализация задачи нахождения всех максимальных клик графа на реконфигурируемых вычислительных системах // Вестник компьютерных и информационных технологий. – 2018. – № 10. – С. 3-10. – Doi: 10.14489/vkit.2018.10.pp.003-010.
18. *Касаркин А.В., Левин И.И.* Реализация алгоритма Брона-Кербоша на реконфигурируемых вычислительных системах // Известия ЮФУ. Технические науки. – 2019. – № 7 (209). – С. 142-152. – Doi: 10.23683/2311-3103-2019-7-142-152.
19. *Levin I., Dordopulo A., Fedorov A., Doronchenko Y.* Design Technology for Reconfigurable Computer Systems with Immersion Cooling. In: Voevodin V., Sobolev S. (eds.) Supercomputing. RuSCDays 2018. Communications in Computer and Information Science, Vol 965. Springer, Cham. – Doi: 10.1007/978-3-030-05807-4_47.
20. *Левин И.И., Пелипец А.В.* Эффективная реализация распараллеливания на реконфигурируемых системах // Вестник компьютерных и информационных технологий. – 2018. – № 8. – С. 11-16.
21. *Kasarkin A.V., Levin I.I., Sorokin D.A.* New iteration parallel-based method for solving graph NP-complete problems with reconfigurable computer systems // IOP Conf. Series: Materials Science and Engineering. – 2020. – Vol. 919 (1). – P. 052007(1-7). – Doi: 10.1088/1757-899X/919/5/052007.

REFERENCES

1. *Kir'yanov A.G., Lyahov A.I., Nekrasov P.O., Platov D.A. i dr.* Protokol mnogoadresnoj marshrutizacii Proximity-based Groupcast in MANET (GiM) [A multiaddress routing protocol Proximity-based Groupcast in MANET (GiM)], *Informacionnye process* [Informatsionnyie protsessy], 2012, No. 3, Vol. 12, pp. 213-228.
2. *Kurejchik V.M., Glushan' V.M., Shcherbakov L.I.* Kombinatornye apparatnye modeli i algoritmy v SAPR [Combinatory hardware models and algorithms for CAD]. Moscow: Radio i svyaz', 1990, 216 p.
3. *Jens Clausen.* Branch and bound algorithms principles and examples. Department of comp sc., university of Copenhagen, Universitetsparken 1, DK-2100 Corpenhagen, Denmark.(March 12, 1999).
4. *Kiryushin N.K., Mikhalev I.V.* Ispol'zovanie mnogoyadernykh uskoriteley dlya resheniya zadachi propozitsional'noy vypolnimosti [Use of multicore accelerators for tasks of propositional satisfiability], *Problemy nauki* [Problemy Nauki], 2017, No. 22 (104).
5. *Dasari N.S., Ranjan D., Mohammad Z.* Maximal Clique Enumeration for Large Graphs on Hadoop Framework, *Proc. of the First Workshop on Parallel Programming for Analytics Applications*, 2014, pp. 21-30. Doi: 10.1145/2567634.2567640.
6. *Rossi R.A., Gleich D.F., Gebremedhin A.H.* Parallel Maximum clique Algorithms with Applications to Network Analysis, *SIAM J. Sci. Comput.*, 2015, Vol. 37, No. 5, pp. 589-616. Doi: 10.1137/14100018X.
7. *Kovács L., Szabó G.* Conceptualization with Incremental Bron-Kerbosch Algorithm in Big Data Architecture, *Acta Polytechnica Hungarica*, 2016, Vol. 13, No. 2, pp. 139-158. Doi: 10.12700/aph.13.2.2016.2.8.

8. *Pattabiraman B. et al.* Fast Algorithms for the Maximum Clique Problem on Massive Graphs with Applications to Overlapping Community Detection, *Internet Mathematics*, 2015, Vol. 11, No. 4-5, pp. 421-448. Doi: 10.1080/15427951.2014.986778.
9. *Dasari N.S., Zubair M., Ranjan D.* A Novel Parallel Algorithm for Maximal Clique Enumeration on Multicore and Distributed Memory Architectures. 10 p. Available at: https://pdfs.semanticscholar.org/9827/9e2cedb14085886fcb4473f1ba483a3df195.pdf (accessed 23 September 2020).
10. *Dasari N.S., Desh Z.M.* pbitMCE: A bit-based approach for maximal clique enumeration on multicore processors, *in: Proc. 20th IEEE International Conference on Parallel and Distributed Systems (ICPADS 2014)*, 2014, pp. 478-485. Doi: 10.1109/PADSW.2014.7097844.
11. *Bron C., Kerbosch J.* Algorithm 457: Finding All Cliques of an Undirected Graph, *Comm of ACM*, 1973, Vol. 16, pp. 575-577.
12. *Johnson D.J. and Trick M.A.* Eds., Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, Workshop, October 11-13, 1993. Boston, MA, USA: American Mathematical Society, 1996.
13. *Moon J. and Moser L*. On cliques in graphs, *Israel Journal of Mathematics*, 1965, Vol. 3, No. 1, pp. 23-28. Available: Doi: 10.1007/BF02760024.
14. *Kalyaev I.A. i dr.* Rekonfiguriruemye mul'tikonveyernye vychislitel'nye struktury [Reconfigurable multipipeline computing structures], under general ed. of I.A. Kalyaev. 2nd ed., revised and enlarged. Rostov-on-Don: Izd-vo YUNTS RAN, 2009, 344 p.
15. Nauchno-issledovatel'skiy tsentr super-EVM i neyrokomp'yuterov [Scientific research center of supercomputers and neurocomputers]. Available at: http://superevm.ru (accessed 23 September 2020).
16. *Kalyaev A.V., Levin I.I.* Modul'no-narashchivaemye mnogoprotsessornye sistemy so strukturno-protsedurnoy organizatsiey vychisleniy [Modular-scalable multiprocessor systems with structural and procedural organization of calculations]. Moscow: Yanus-K, 2003, 380 p.
17. *Kasarkin A.V., Levin I.I.* Strukturnaya realizatsiya zadachi nakhozhdeniya vsekh maksimal'nykh klik grafa na rekonfiguriruemykh vychislitel'nykh sistemakh [Structural implementation of the task of maximal clique enumeration on reconfigurable computer systems], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Herald of computer and information technologies], 2018, No. 10, pp. 3-10. Doi: 10.14489/vkit.2018.10.pp.003-010.
18. *Kasarkin A.V., Levin I.I.* Realizatsiya algoritma Brona-Kerbosha na rekonfiguriruemykh vychislitel'nykh sistemakh [Implementation of the Bron-Kerbosch algorithm on reconfigurable computer systems], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering sciences], 2019, No. 7 (209), pp. 142-152. Doi: 10.23683/2311-3103-2019-7-142-152.
19. *Levin I., Dordopulo A., Fedorov A., Doronchenko Y.* Design Technology for Reconfigurable Computer Systems with Immersion Cooling. In: Voevodin V., Sobolev S. (eds.) Supercomputing. RuSCDays 2018. Communications in Computer and Information Science, Vol 965. Springer, Cham. Doi: 10.1007/978-3-030-05807-4_47.
20. *Levin I.I., Pelipets A.V.* Effektivnaya realizatsiya rasparallelivaniya na rekonfiguriruemykh sistemakh [Efficient Parallel Execution on Reconfigurable Systems], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Herald of computer and information technologies], 2018, No. 8, pp. 11-16.
21. *Kasarkin A.V., Levin I.I., Sorokin D.A.* New iteration parallel-based method for solving graph NP-complete problems with reconfigurable computer systems, *IOP Conf. Series: Materials Science and Engineering*, 2020, Vol. 919 (1), pp. 052007(1-7). Doi: 10.1088/1757-899X/919/5/052007.

Статью рекомендовал к опубликованию д.т.н. Э.В. Мельник.

Касаркин Алексей Викторович – Южный федеральный университет; e-mail: kav589@mail.ru; 347931, г. Таганрог, пер. Итальянский, 106; тел.: +79045065636; кафедра ИМС, аспирант.

Kasarkin Alexey Viktorovich – Southern Federal University; e-mail: kav589@mail.ru; 106, Italyansky lane, Taganrog, 347931, Russia; phone: +79045065636; graduate student.

УДК 004.424.2 DOI 10.18522/2311-3103-2020-7-130-142

М.Д. Чекина

РЕАЛИЗАЦИЯ ФРАКТАЛЬНОГО СЖАТИЯ И ДЕКОМПРЕССИИ ИЗОБРАЖЕНИЙ ПАРАЛЛЕЛЬНО-КОВЕЙЕРНЫМ СПОСОБОМ НА РЕКОНФИГУРИРУЕМЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ

Фрактальные алгоритмы находят все большее количество областей применения – от компьютерной графики до моделирования сложных физических процессов, но для их программной реализации требуются значительные вычислительные мощности. Фрактальное сжатие изображений отличается высокой степенью компрессии данных при хорошем качестве восстановленного изображения. Целью данной работы является повышение производительности реконфигурируемых вычислительных систем (РВС) при реализации фрактального сжатия и декомпрессии изображений. В работе описаны разработанные методы фрактального сжатия и последующей декомпрессии изображений, реализованные параллельно-конвейерным способом для РВС. Основная идея параллельной реализации фрактального сжатия изображений сводится к параллельному выполнению попарного сравнения доменных и ранговых блоков. Для достижения наилучшей производительности необходимо одновременно сравнивать максимальное количество пар. При практической реализации фрактального сжатия изображений на РВС учитываются такие критические ресурсы, как количество входных каналов и количество логических ячеек ПЛИС. Для задачи фрактального сжатия изображения критическим ресурсом являются каналы данных, поэтому параллельная организация вычислений заменяется параллельно-конвейерной после выполнения редукцию производительности параллельной вычислительной структуры. Подача каждого операнда в вычислительную структуру осуществляется последовательно (побитово), что экономит вычислительный ресурс и уменьшает простой оборудования. Для хранения коэффициентов системы итерируемых функций, кодирующих изображение, введена структура данных, задающая отношения между номерами ранговых и доменных блоков и соответствующими параметрами. Для удобства последующей декомпрессии элементы массива, кодирующего сжатое изображение, упорядочены по номерам ранговых блоков, что позволяет избежать двойной косвенной адресации в вычислительной структуре. Представленный подход позволяет масштабировать параллельно-конвейерную программу на любое количество программируемых логических интегральных схем (ПЛИС). Практическая реализация фрактального сжатия изображений, выполненная на реконфигурируемом компьютере Терциус-2, содержащем восемь ПЛИС, обеспечивает ускорение в 15000 раз по сравнению с универсальным многоядерным процессором и в 18–25 раз по сравнению с существующими решениями для ПЛИС. Реализация декомпрессии изображения на реконфигурируемом компьютере показывает ускорение в 380 раз по сравнению с аналогичной реализацией для многоядерного универсального процессора.

Фракталы; фрактальное сжатие изображений; ПЛИС; реконфигурируемые вычислительные системы.

M.D. Chekina

THE PARALLEL-PIPELINED IMPLEMENTATION OF THE FRACTAL IMAGE COMPRESSION AND DECOMPRESSION FOR RECONFIGURABLE COMPUTING SYSTEMS

Fractal algorithms find an increasing number of areas of application - from computer graphics to modeling complex physical processes, but their software implementation requires significant computing power. Fractal image compression is characterized by a high degree of data compression with good quality of the reconstructed image. The aim of this work is to improve the performance of reconfigurable computing systems (RCS) when implementing fractal compression and decompression of images. The paper describes the developed methods of fractal compression and subsequent decompression of images, implemented in a parallel-pipeline method for RCS.

The main idea of parallel implementation of fractal image compression is reduced to parallel execution of pairwise comparison of domain and rank blocks. For best performance, the maximum number of pairs must be compared simultaneously. In the practical implementation of fractal image compression on the DCS, such critical resources as the number of input channels and the number of FPGA logical cells are taken into account. For the problem of fractal image compression, data channels are a critical resource; therefore, the parallel organization of computations is replaced by parallel-pipeline, after the performance reduction of the parallel computational structure is performed. Each operand goes into the computational structure sequentially (bit by bit) to save computational resources and reduce equipment downtime. To store the coefficients of the iterated functions system encoding the image, a data structure has been introduced that specifies the relation between the numbers of rank and domain blocks and the corresponding parameters. For the convenience of subsequent decompression, the elements of the array encoding the compressed image are ordered by the numbers of the rank blocks, which avoids double indirect addressing in the computational structure. Applying this approach for parallel-pipeline programs allows scaling computing structure to plurality programmable logic arrays (FPGAs). A practical implementation performed on a reconfigurable computer Tertius-2 containing eight FPGAs provides an acceleration of 15000 times compared to a universal multi-core processor and 18–25 times compared to existing solutions for FPGAs. The implementation of image decompression on a reconfigurable computer shows an acceleration of 380 times in comparison with the similar implementation for a multi-core general-purpose processor.

Fractals; fractal image compression; FPGA; reconfigurable computing systems.

Введение. Фрактальные алгоритмы находят применение в самых разнообразных классах задач: графике, сжатии изображений, анализе финансовых рынков, моделировании сложных физических процессов. Для эффективной программной реализации фрактальных алгоритмов требуются значительные вычислительные мощности. Стандартные методы и средства для многопроцессорных систем не обеспечивают удовлетворительную эффективность, т.к. специфика фрактальных алгоритмов подразумевает обмен большими массивами данных между вычислительными устройствами, из-за чего невозможно эффективное масштабирование параллельных программ. Реализации фрактальных алгоритмов для многопроцессорных систем на основе универсальных процессоров или графических ускорителей обеспечивают меньшую производительность, чем системы, построенные на основе ПЛИС [1].

Фрактальное сжатие изображений является одной из самых востребованных задач на основе фрактальных методов. В его основе лежит обнаружение самоподобных участков в изображении. Впервые возможность применения систем итерируемых функций (СИФ) к проблеме сжатия изображения была исследована М. Барнсли и А. Слоуном [2]. Жакен А. представил метод фрактального кодирования, использующий доменные и ранговые блоки квадратной формы, покрывающие всё изображение. Этот подход стал основой для большинства методов фрактального кодирования и был усовершенствован Ю. Фишером [3], и рядом других исследователей

Приложения, реализующие фрактальное сжатие изображений для традиционных многопроцессорных систем, обладают рядом недостатков. Вычислительные системы на универсальных процессорах требуют больших накладных расходов на передачу данных между процессами, а также их синхронизацию, что приводит к простою части вычислительного оборудования [4]. Графические ускорители обладают большой вычислительной мощностью, но не могут функционировать как самостоятельные устройства. При выполнении фрактального алгоритма на системе, основным вычислителем которой является графический ускоритель, требуется постоянный обмен данными с центральным процессором и оперативной памятью. Пропускная способность этих каналов существенно ниже, чем частота работы графического ускорителя, что приводит к снижению производительности системы [5–8].

Существующие реализации фрактального сжатия изображений разработаны для реконфигурируемых акселераторов на основе только одной ПЛИС, которая не может функционировать как самостоятельное устройство, и требуют для работы наличия универсального процессора. Кроме того, все известные реализации не предусматривают возможность масштабирования алгоритмов на многокристальную систему [9–13].

Помимо описанных выше недостатков, как правило, параллельно реализуют только сжатие изображения, оставляя его декомпрессию для выполнения на персональном компьютере. Это объясняется асимметричностью алгоритма фрактального сжатия изображения, компрессия данных занимает в десятки раз больше времени, чем их декомпрессия, но для изображений с высоким разрешением обратная операция также будет происходить длительное время.

Реконфигурируемые вычислительные системы (РВС), объединяющие в единый ресурс множество ПЛИС [14], обладают высокой реальной производительностью и имеют большой потенциал для реализации фрактальных алгоритмов, но отсутствует инструментарий для работы с самоподобными структурами на РВС.

Метод реализации фрактального сжатия изображений для РВС параллельно-конвейерным способом. Фрактальная архивация основана на том, что с помощью трехмерных аффинных преобразований [15] изображение представляется в компактной форме. При этом изображение одновременно разбивается на два множества: неперекрывающихся ранговых блоков и перекрывающихся доменных блоков. Ранговые блоки могут быть одинакового размера, хотя более эффективно использовать адаптивное разбиение с переменным размером блоков. Это дает возможность плотно заполнять ранговыми блоками меньшего размера части изображения, содержащие мелкие детали.

После разбиения для каждого рангового блока перебирают все доменные блоки для поиска максимального соответствия между ними. Домены сжимают до размеров рангового блока и выполняют операции изменения ориентации. Далее вычисляют оптимальные значения коэффициентов преобразования для наилучшего соответствия ранговому блоку. Сначала вычисляется сдвиг по яркости:

$$v = \left(\sum_{i=1}^{m} \sum_{j=1}^{m} r_{ij} - u \sum_{i=1}^{m} \sum_{j=1}^{m} d_{ij} \right) / m^2 ,$$

здесь m – длина стороны рангового и усредненного доменного блока в пикселях, d_{ij} – значение яркости пикселя усредненного доменного блока, r_{ij} – значение яркости пикселя рангового блока, u – коэффициент сжатия яркости.

Затем для определения соответствия двух блоков между ними находят расстояние

$$E(R,D) = \sum_{i=1}^{m} \sum_{j=1}^{m} \left(u d_{ij} + v - r_{ij} \right)^2 .$$

Если для некоторого домена после применения к нему преобразования при соответствующих коэффициентов системы итерируемых функций (СИФ) его значение E(R,D) не превышает заданной допустимой погрешности, рассматриваемый ранговый блок считают покрытым данным доменным блоком – производится сохранение подходящих коэффициентов СИФ, и происходит переход к обработке следующего ранга.

Основная сложность фрактального сжатия заключается в том, что для нахождения соответствующих доменных блоков требуется полный перебор всех возможных пар, что требует больших затрат времени и вычислительного ресурса.

Общая идея параллельной реализации фрактального сжатия изображений сводится к параллельному выполнению попарного сравнения доменных и ранговых блоков.

Согласно представленному алгоритму, для выполнения фрактального сжатия изображения необходимо выполнить четыре базовых операции над всеми парами ранговых и доменных блоков: сжатие доменного блока до размеров рангового (W), поворот сжатого блока (F_i, где i=0..7), сдвиг по яркости (L), вычисление расстояния между преобразованным доменным блоком и ранговым (E).

При практической реализации фрактального сжатия изображений на РВС следует учитывать критические ресурсы: количество входных каналов и количество логических ячеек ПЛИС. Реализация параллельно-конвейерной программы проводилась на реконфигурируемом компьютере Терциус-2 [16], выпускаемом НИЦ суперЭВМ и нейрокомпьютеров, содержащем 8 ПЛИС XCKU095, каждая из которых обладает ресурсом в 1176000 LUT, и универсальный процессор Intel Core i5-6600K Skylake. Данный вычислительный блок содержит 16 блоков распределенной памяти с 32-битными каналами.

Для задачи фрактального сжатия изображения критическим ресурсом являются каналы данных, поэтому необходимо перейти от параллельной организации вычислений к параллельно-конвейерной, выполнив редукцию производительности параллельной вычислительной структуры [17]. Коэффициент редукции производительности параллельной вычислительной структуры по каналам для задачи фрактального сжатия изображений можно выразить формулой

$$K_c = \frac{BN_{im}n_D}{n_R C},$$

здесь N_{im} – количество пикселей в исходном изображении, n_D – количество пикселей в доменном блоке, n_R – количество пикселей в ранговом блоке, C – количество доступных каналов, B – количество бит, кодирующих один пиксель изображения.

При реализации задачи на реконфигурируемом компьютере Терциус-2 с пословной подачей элементов доменных блоков значение коэффициента редукции составит K_c= 62500.

При всех преимуществах параллельной подачи всех элементов массива такой способ имеет ряд недостатков: параллельная подача всего доменного блока удобна только при фиксированном размере всех доменных блоков, если использовалось динамическое разбиение изображения, например, с помощью четверичного дерева, то может возникнуть потребность также и в динамическом выборе ширины канала для входных данных.

При пословной подаче элементов доменных блоков возникнет простой оборудования из-за исключения из работы тех доменов, для которых нашлось соответствие с ранговыми, что диктует необходимость изменить способ подачи данных. Если поразрядно (побитово) подавать значения яркости каждого из пикселей доменного блока с общим количеством точек 64, четыре одновременно подаваемых доменных блока займут 256 каналов.

При осуществлении последовательной (побитовой) подачи элементов массива пикселей коэффициент редукции производительности вычислительной структуры составит K_c= 1500000.

Операции сжатия и поворота можно провести один раз перед началом конвейера, подавая для дальнейших вычислений модифицированные доменные блоки D_i" (рис. 1).

В начале каждого конвейера будут проведены операции сжатия и смены ориентации, для каждой модификации вычислены сдвиг по яркости и расстояние.

Выполняя побитовую последовательную подачу данных для каждого доменного блока, мы сможем одновременно осуществлять обработку 256 доменных блоков.

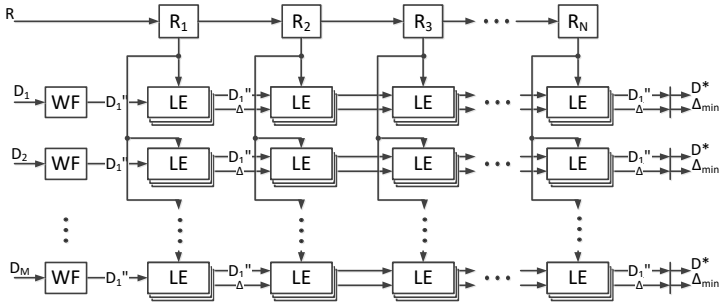


Рис. 1. Вычислительная структура с выносом блоков сжатия и поворота в начало конвейера

При такой подаче данных появляется возможность использовать одноразрядные арифметические блоки, не требующие больших затрат оборудования. При таких параметрах блок сжатия-поворота WF будет занимать 15 LUT, а 8 блоков LE займут 2120 LUT. Ресурс ПЛИС Kintex UltraScale XCKU095 – 1176000 LUT, на одном кристалле можно будет разместить не более 550 подобных вычислительных единиц.

Если осуществляется параллельная подача каждого восьмибитного слова, то возможна одновременная подача только 32-х доменных блоков. Арифметические блоки для работы с восьмибитными словами требуют большего ресурса (WF – 105 LUT, LE – 12904 LUT), поэтому на аналогичной ПЛИС возможно разместить не более 90 вычислительных единиц.

Доменные блоки могут быть пропущены через конвейер несколько раз. После нахождения совпадения домен исключается из потока. В этом случае при пословной подаче операндов будет простаивать большая часть оборудования, чем при использовании последовательной побитовой подачи.

Для хранения коэффициентов СИФ предлагается структура данных $X=(N_r,N_d,i,v)$, где N_r – номер рангового блока, N_d – номер доменного блока, i – поворотный коэффициент, v – сдвиг по яркости. В одном элементе такой структуры будет храниться информация о соответствии между конкретными ранговым и доменным блоками, что значительно облегчит впоследствии восстановление изображения по коэффициентам СИФ. Процесс формирования структуры X показан на рис. 2.

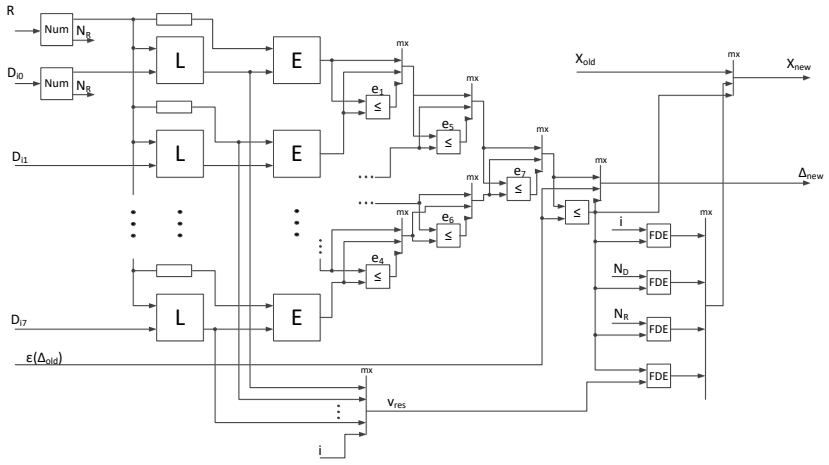


Рис. 2. Вычислительная структура, формирующая массив X, содержащий коэффициенты СИФ

Структура X ставит в соответствие друг другу массивы данных ранговых и доменных блоков. При этом соответствие не является биективным: одному ранговому блоку может соответствовать несколько доменных блоков, но не наоборот.

Метод параллельной реализации для РВС декомпрессии изображения сжатого фрактальным способом. Декомпрессия изображения, сжатого фрактальным способом, производится за счет итерационного применения найденных ранее коэффициентов системы итерируемых функций к произвольному начальному изображению. В качестве начального может быть взято любое изображение, в соответствии с теоремой о сжимающем отображении [18] итерационный процесс будет сходиться к неподвижному изображению. Изображение из коэффициентов СИФ восстанавливается следующим образом:

1. Создаются два произвольных изображения одинакового размера A и B.
2. На изображение A накладывается сетка ранговых блоков, аналогичная той, на которую было разбито сжатое изображение. С изображением B проводится аналогичная операция, но на него накладывается сетка из доменных блоков.
3. Для каждого доменного блока области B проводится соответствующее аффинное преобразование ранговых областей изображения A, описанное коэффициентами СИФ. Результат помещается в область B. После преобразования получается совершенно новое изображение.
4. Данные из области B копируются в область A.
5. Шаги 3 и 4 повторяются до тех пор, пока изображения A и B не станут неразличимыми.

Номера соответствующих друг другу ранговых и доменных блоков содержатся вместе с коэффициентами СИФ в структурах X, множество которых подается на вход процедуры по восстановлению изображения. Для дальнейшей работы необходимо выделить некоторую рабочую область в памяти ПЛИС или внешней памяти, где будет производиться итерационная декомпрессия изображения.

Предполагается, что входные данные X для процедуры декомпрессии, полученные в ходе процедуры сжатия изображения, не упорядочены, поэтому возникает проблема обращения к конкретной области памяти устройства, а также наложения сеток доменных и ранговых блоков на рабочую область. Для решения этой проблемы, как правило, используется косвенная адресация: адреса нужных ранговых и доменных блоков вычисляются, исходя из их номеров, содержащихся во входных данных X.

Традиционная реализация процедуры декомпрессии не может быть реализована на РВС в силу невозможности одновременной записи и чтения из нее потока доменных и ранговых блоков. Информационные потоки доменных и ранговых блоков будут постоянно прерываться для осуществления процедуры записи в память результата. На рис. 3 показано, что при традиционной реализации процедуры декомпрессии блоки RAM, содержащие доменные и ранговые блоки, должны выступать как приемник и как источник данных.

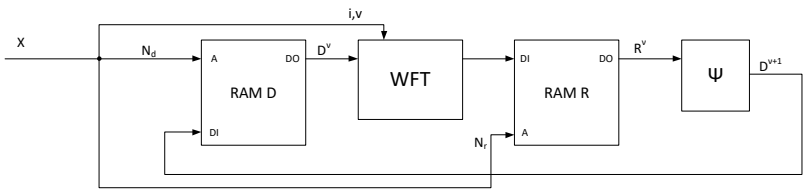


Рис. 3. Тривиальная реализация процедуры декомпрессии изображения

Так, из RAM D элементы доменных блоков поступают в вычислительную структуру WFT (рис. 4), осуществляющую аффинные преобразования.

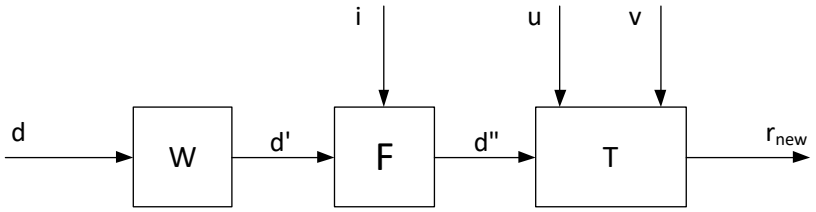


Рис. 4. Составляющие блока WFT

Здесь блоки W и F соответствуют операциям сжатия и поворота. На вход блока F подается один из коэффициентов СИФ – номер поворота доменного блока. В блоке T осуществляется обратный сдвиг по яркости, структура которого показана на рис. 5.

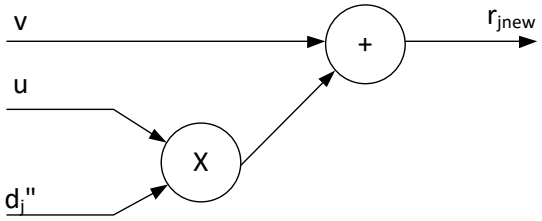


Рис. 5. Вычислительная структура блока T

Преобразованные данные записываются в RAM R, которое, в свою очередь, является источником данных для получения $D^{\nu+1}$ – элементов доменного блока для нового временного слоя итерационного процесса.

Через вычислительную структуру будет проходить не поток операндов, а единичные операнды. Это, в свою очередь, приводит к увеличению скважности и падению реальной производительности системы. Подобная вычислительная структура практически не поддается распараллеливанию из-за невозможности параллельного доступа к одной памяти вследствие косвенной адресации.

Для такой структуры характерна высокая скважность порядка 300 тактов, и скорость обработки данных снижается в 300 раз. Для ее снижения вычислительную структуру, производящую декомпрессию изображения, можно разделить на два кадра, первый из которых реализует функцию поиска значений рангового блока на основе предыдущих значений доменного $R^{\nu} = F\left(X, D^{\nu-1}\right)$, а второй кадр обеспечивает обновление доменных блоков на основе полученных ранговых $D^{\nu} = \Psi\left(X, R^{\nu}\right)$, здесь ν – номер временного слоя в итерационном процессе. При реализации такого подхода по две памяти для ранговых и доменных блоков используются либо как приемник, либо как источник данных. Такая структура соответствует классической формулировке алгоритма восстановления изображения (рис. 6).

Номер доменного блока N_d, содержащийся в X, формирует адрес чтения из памяти RAM D, и элементы доменных блоков из нее поступают в функцию WFT вместе с коэффициентами (i,v) СИФ, также содержащимися в X.

После преобразования данные записываются в память RAM R по адресу, сформированному согласно поступившему на вход номеру рангового блока N_r.

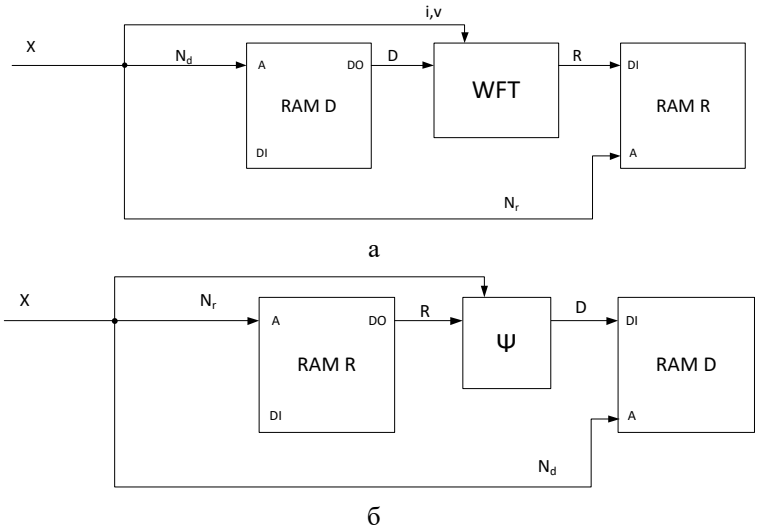


Рис. 6. Вычислительная структура с двумя проходами а) прямой ход – поиск элементов ранговых блоков, б) обратный ход – перенос элементов ранговых блоков в доменные

Следует отметить, что в данном варианте применяется двойная косвенная адресация для блоков RAM, содержащих ранговые и доменные блоки восстанавливаемого изображения, поэтому распараллелить вычислительную структуру по данным становится практически невозможно из-за неизбежности одновременных обращений к одному каналу памяти.

Однако можно предварительно выполнить сортировку элементов массива X, представляющих собой набор $\left(N_r, N_d, i, v\right)$, по номерам ранговых блоков N_r, упростив исходный набор из четырех параметров до трех $X_R = \left(N_d, i, v\right)$. В этом случае мы перейдем от двойной косвенной адресации к одинарной косвенной адресации. Это позволит улучшить распараллеливание по данным. На рис. 7 показана вычислительная структура, использующая в качестве входных данных отсортированные X_R.

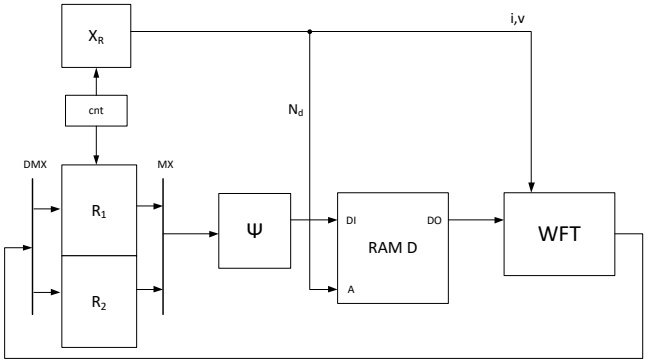


Рис. 7. Использование упорядоченных значений X_R

На вход вычислительной структуры поступают элементы X_R, которые синхронизированы с ранговыми блоками, хранящимися во внешней памяти устройства. Номер доменного блока N_d формирует адрес чтения в двухпортовой памяти RAM D, и элементы соответствующего доменного блока вычитываются для преобразования функцией WFT. После выполнения всех операций преобразованные данные записываются во внешнюю память. Использование двухпортовой памяти позволяет выполнять независимо операции записи и чтения данных. Содержимое обновленных ранговых блоков вычитываются из внешней памяти и поступает на вход RAM D для осуществления следующих итераций.

Подобный подход позволяет избежать двойной косвенной адресации, которая использовалась для ранговых и доменных блоков, перейдя к одинарной – только для доменных блоков. Поскольку ранее было проведено упорядочивание элементов X, поступающих на выход схемы, то новые ранговые блоки будут появляться в том порядке, в котором они расположены внутри изображения, что избавляет от необходимости использовать для них косвенную адресацию, оставляя ее применение только для доменных блоков.

Очевидным способом сортировки X является сортировка массива в потоке, преобразовывая структуры вида $X = (N_r, N_d, i, v)$ к виду $X_R = (N_d, i, v)$, т.е. номер элемента нового массива X будет соответствовать номеру рангового блока. Но такой подход потребует дополнительных операций и приведет к лишним временным затратам, поэтому целесообразно осуществлять сортировку непосредственно при формировании X во время кодирования изображения.

Выше описана вычислительная структура X, формирующая соответствие между коэффициентами СИФ и номерами ранговых и доменных блоков. При этом элементы X не были упорядочены по какому-либо признаку. Поскольку ранговые блоки поступают в вычислительную схему по порядку, то возможно формирование элементов X соответственно номерам ранговых блоков, тогда номер каждого из элементов X будет идентичен номеру рангового блока, который был использован при формировании данного элемента.

За счет использования КРП возможно организовать до 256 потоков данных, осуществляя их подачу побитно. При этом для взаимодействия с памятью, содержащей в себе доменные блоки, придется осуществлять буферизацию параллельно идущих потоков данных. На рис. 8 показана вычислительная структура для нескольких параллельно обрабатываемых ранговых блоков. Из X, поступающего на вход, выделяются номера доменных блоков N_d. Они управляют выбором доменного блока, поступающего в блоки WFT для дальнейших преобразований. Также из X на входы блоков WFT поступают коэффициенты i и v. После преобразования данные записываются в соответствующие ранговые блоки. Затем порядок чтения и записи меняется, и теперь память, содержащая обновленные данные ранговых блоков, становится источником. Из нее данные поступают на вход памяти, хранящей доменные блоки для их обновления.

Процедура восстановления изображения, сжатого фрактальным способом, с трудом поддается распараллеливанию на РВС. Предложенный выше подход не является оптимальным решением этой задачи, поэтому поиск максимально оптимального способа ее распараллеливания может стать предметом дальнейших исследований.

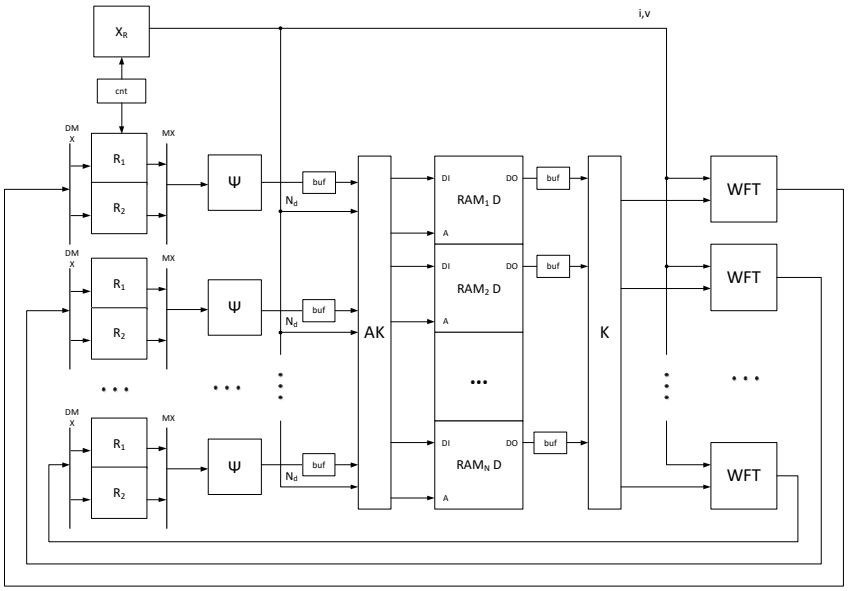


Рис. 8. Распараллеливание процедуры декомпрессии изображения

Оценка производительности фрактального сжатия и декомпрессии изображений на РВС. Время работы алгоритма фрактального сжатия изображений для РВС Терциус-2 можно оценить по формуле

$$T = \frac{1}{\mu U}\left(Ld + \left(Lt_{WF} + QLt_{LE} \right) N_D \frac{N_R}{QCF} \right),$$

где Lt_{WF} – латентность блока сжатия и поворота, Lt_{LE} – латентность блоков, определяющих сдвиг по яркости и расстояние между доменным и ранговым блоками, N_D - количество доменных блоков, N_R – количество ранговых блоков, F – количество ПЛИС, Ld – количество тактов, необходимое для загрузки одного доменного блока, Q – количество блоков LE в конвейере, μ – частота работы ПЛИС, U – коэффициент использования оборудования, показывающий соотношение времени полной загрузки вычислительного ресурса к общему времени работы.

При прохождении каждого доменного блока через вычислительную структуру он будет исключён из конвейера при совпадении с заданной погрешностью с ранговым блоком. Вероятность данного события оценивается как 0,5, что соответствует значению коэффициента использования оборудования U=0,5. Если доменный блок проходит через вычислительную структуру несколько раз для разных групп ранговых блоков, то вероятность того, что он будет исключён, и коэффициент простоя оборудования примет значение в интервале от 0,5 до 1, что в среднем даст значение U=0,75.

Для рассматриваемого примера время сжатия 4-мегапиксельного изображения на реконфигурируемом компьютере Терциус-2 при пословной подаче восьмибитных операндов массива составит около 7,5 секунд, а при последовательной побитовой подаче данных для той же задачи – приблизительно 1,5 секунды, за счет многократного увеличения числа блоков конвейерной обработки.

Фрактальное сжатие аналогичного изображения задачи на универсальном 4-ядерном процессоре Intel Core i7-4770 с тактовой частотой 3,50 GHz будет выполняться более шести часов (23000 секунд). Реализация фрактального сжатия

изображений для реконфигурируемого компьютера показывает ускорение в 15000 раз по сравнению с аналогичной реализацией для многоядерного универсального процессора.

При проведении сравнения предложенного подхода с существующими однокристальными реализациями фрактального сжатия [1, 5-9] ускорение составит 18-25 раз при тех же параметрах.

Выполним оценку времени, которое займет декомпрессия цветного четырехмегапиксельного изображения в формате RGB, разбитого на 250 000 квадратных ранговых блоков со стороной 4 пикселя. Время работы можно оценить по формуле

$$T = 1,3t\left(Ld + \frac{3LtN_R I}{CF} \right),$$

где t – величина обратная частоте работы ПЛИС, Ld – количество тактов, необходимое для загрузки одной порции входных данных, N_R – количество ранговых блоков, Lt – латентность вычислительной схемы, I – количество итераций, требуемых для восстановления изображения, C – количество каналов для подачи данных, F – количество задействованных ПЛИС. Для примера, указанного выше, оценочное время работы на реконфигурируемом компьютере Терциус-2 составит примерно 0,003 секунды, тогда как время выполнения этой же задачи на универсальном четырехъядерном процессоре Intel Core i7-4770 составит примерно 1,15 секунд. Таким образом, реализация декомпрессии изображения на РВС показывает ускорение в 380 раз по сравнению с аналогичной реализацией для многоядерного универсального процессора.

Заключение. Описанный выше подход к реализации фрактального сжатия изображений позволяет получать параллельно-конвейерные программы для РВС различных конфигураций, а также дает большие возможности для масштабирования алгоритма, позволяя наращивать производительность вычислительной системы с увеличением количества кристаллов ПЛИС.

Данный подход можно использовать для реализации на реконфигурируемых вычислительных системах других фрактальных задач – построения фрактальных ландшафтов, фрактального анализа природных систем и финансовых рынков.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Doris Chen*. Deshanand Singh Fractal Video Compression in OpenCL: An Evaluation of CPUs, GPUs, and FPGAs as Acceleration Platforms // 2013 18th Asia and South Pacific Design Automation Conference. – P. 297-304.
2. *Barnsley M., Hurd L.P.* Fractal Image Compression. – Wellsley Massachusetts: A.K. Peters Ltd, 1993. – 224 p.
3. *Fisher Y.* Fractal Image Compression: Theory and Application. – Springer-Verlag, New York, 1995.
4. *Бойченко И.В., Кулбаев С.С., Немеров А.А., Голенков В.В.* Эксперимент по фрактальному сжатию rgb изображений на вычислительном кластере // Известия Томского политехнического университета. – 2012. – Т. 321, № 5. – С. 87-92.
5. *Munesh Singh Chauhan, Ashish Negi, Prashant Singh Rana.* Fractal Image Compression using Dynamically Pipelined GPU Clusters // Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS, 2012), December 28-30, 2012.
6. *Mohammed Ismail B., Eswara Reddy B., Bhaskara Reddy T*. Cuckoo inspired fast search algorithm for fractal image encoding // Journal of King Saud University – Computer and Information Sciences, 2016.
7. *Md. Enamul Haque, Abdullah Al Kaisan, Mahmudur R Saniat, and Aminur Rahman*. GPU Accelerated Fractal Image Compression for Medical Imaging in Parallel Computing Platform // arXiv:1404.0774v1 [cs.DC] 3 Apr 2014.

8. *Munesh Singh Chauhan, Ashish Negi.* Fractals Image Rendering and Compression using GPUs // International Journal of Digital Information and Wireless Communications (IJDIWC) 2(1): 1-6 The Society of Digital Information and Wireless Communications, 2012. – ISSN 2225-658X.
9. *A-M.H.Y. Saad, M.Z.* Abdullah High-speed implementation of fractal image compression in low cost FPGA, Microprocessors and Microsystems 47. August 2016. – Doi: 10.1016/j.micpro.2016.08.004.
10. *A-M.H.Y. Saad, M.Z.* Abdullah High-Speed Fractal Image Compression Featuring Deep Data Pipelining Strategy // IEEE Access PP(99):1-1 · November 2018. – Doi: 10.1109/ACCESS.2018.2880480.
11. *Son T.N., Hung O.M., Xuan D.T., Tran V.L., Dzung N.T., Hoang T.M*. Implementation of Fractal image compression on FPGA // 4th International Conference on Communications and Electronics ICCE 2012, online IEEExplorer, Hue City, Vietnam. 1-3 Aug. 2012. – P. 339-344.
12. *Padmavati S.* Vaibhav Meshram FPGA Implementation for Fractal Quadtree Image Compression International Journal of Computer Sciences and Engineering. – Oct. 2018. – Vol. 6, Issue-10.
13. *Thai Nam Son Tran V Long, Hoang Manh Thang, Nguyen Tien Dzung*. Efficient implementation of a fractal color image compression on FPGA // 2013 International Conference of Soft Computing and Pattern Recognition (SoCPaR). – 2013. – Doi: 10.1109/SOCPAR.2013.7054124.
14. Thai Nam Son Thang Manh Hoang, Nguyen Tien Dzung Nguyen Hoang Giang Fast FPGA Implementation of YUV-based Fractal Image Compression // 2014 IEEE Fifth International Conference on Communications and Electronics (ICCE). – 2014. – Doi: 10.1109/CCE.2014.6916745.
15. *Гузик В.Ф., Каляев И.А., Левин И.И.* Реконфигурируемые вычислительные системы / под общ. ред. И.А. Каляева. – Ростов-на-Дону: Изд-во ЮФУ, 2016. – 472 с.
16. *Беклемишев Д.В.* Курс аналитической геометрии и линейной алгебры. – М.: Высш. шк., 1998. – 320 с.
17. *Левин И.И., Дордопуло А.И., Сорокин Д.А., Каляев З.В., Доронченко Ю.И.* Реконфигурируемые компьютеры на основе плис Xilinx Virtex Ultrascale // В сб.: Параллельные вычислительные технологии (ПаВТ'2019). Короткие статьи и описания плакатов XIII Международной научной конференции. – 2019. – С. 288-298.
18. *Дордопуло А.И., Левин И.И.* Методы редукции вычислений для программирования гибридных реконфигурируемых вычислительных систем // XII мультиконференция по проблемам управления (МКПУ-2019): Матер. конференции: в 4 т. – 2019. – С. 78-82.
19. *Колмогоров А.Н., Фомин С.В.* Элементы теории функций и функционального анализа. – 4-е изд. – М.: Наука, 1976. – 544 с.
20. *Левин И.И., Пелипец А.В.* Эффективная реализация распараллеливания на реконфигурируемых системах // Вестник компьютерных и информационных технологий. – 2018. – № 8. – С. 11-16.

REFERENCES

1. *Doris Chen*. Deshanand Singh Fractal Video Compression in OpenCL: An Evaluation of CPUs, GPUs, and FPGAs as Acceleration Platforms, *2013 18th Asia and South Pacific Design Automation Conference*, pp. 297-304.
2. *Barnsley M., Hurd L.P.* Fractal Image Compression. Wellsley Massachusetts: A.K. Peters Ltd, 1993, 224 p.
3. *Fisher Y.* Fractal Image Compression: Theory and Application. Springer-Verlag, New York, 1995.
4. *Boychenko I.V., Kulbaev S.S., Nemerov A.A., Golenkov V.V.* Eksperiment po fraktal'nomu szhatiyu rgb izobrazheniy na vychislitel'nom klastere [Experiment on fractal compression of rgb images on a computing cluster], *Izvestiya Tomskogo politekhni-eskogo universiteta* [Bulletin of the Tomsk Polytechnic University], 2012, Vol. 321, No. 5, pp. 87-92.
5. *Munesh Singh Chauhan, Ashish Negi, Prashant Singh Rana.* Fractal Image Compression using Dynamically Pipelined GPU Clusters, *Proceedings of the Second International Conference on Soft Computing for Problem Solving (SocProS, 2012), December 28-30, 2012.*
6. *Mohammed Ismail B., Eswara Reddy B., Bhaskara Reddy T*. Cuckoo inspired fast search algorithm for fractal image encoding, *Journal of King Saud University – Computer and Information Sciences, 2016*.
7. *Md. Enamul Haque, Abdullah Al Kaisan, Mahmudur R Saniat, and Aminur Rahman*. GPU Accelerated Fractal Image Compression for Medical Imaging in Parallel Computing Platform, *arXiv:1404.0774v1 [cs.DC] 3 Apr 2014*.

8. *Munesh Singh Chauhan, Ashish Negi.* Fractals Image Rendering and Compression using GPUs, *International Journal of Digital Information and Wireless Communications (IJDIWC) 2(1): 1-6 The Society of Digital Information and Wireless Communications, 2012*. ISSN 2225-658X.
9. *A-M.H.Y. Saad, M.Z.* Abdullah High-speed implementation of fractal image compression in low cost FPGA, Microprocessors and Microsystems 47. August 2016. Doi: 10.1016/j.micpro.2016.08.004.
10. *A-M.H.Y. Saad, M.Z.* Abdullah High-Speed Fractal Image Compression Featuring Deep Data Pipelining Strategy, *IEEE Access PP(99):1-1 · November 2018*. Doi: 10.1109/ACCESS.2018.2880480.
11. *Son T.N., Hung O.M., Xuan D.T., Tran V.L., Dzung N.T., Hoang T.M*. Implementation of Fractal image compression on FPGA, *4th International Conference on Communications and Electronics ICCE 2012, online IEEExplorer, Hue City, Vietnam. 1-3 Aug. 2012*, pp. 339-344.
12. *Padmavati S.* Vaibhav Meshram FPGA Implementation for Fractal Quadtree Image Compression, *International Journal of Computer Sciences and Engineering*, Oct. 2018, Vol. 6, Issue-10.
13. *Thai Nam Son Tran V Long, Hoang Manh Thang, Nguyen Tien Dzung*. Efficient implementation of a fractal color image compression on FPGA, *2013 International Conference of Soft Computing and Pattern Recognition (SoCPaR)*, 2013. Doi: 10.1109/SOCPAR.2013.7054124.
14. Thai Nam Son Thang Manh Hoang, Nguyen Tien Dzung Nguyen Hoang Giang Fast FPGA Implementation of YUV-based Fractal Image Compression, *2014 IEEE Fifth International Conference on Communications and Electronics (ICCE)*, 2014. Doi: 10.1109/CCE.2014.6916745.
15. *Guzik V.F., Kalyaev I.A., Levin I.I.* Rekonfiguriruemye vychislitel'nye sistemy [Reconfigurable computing systems], ed. by I.A. Kalyaeva. Rostov-on-Don: Izd-vo YUFU, 2016, 472 p.
16. *Beklemishev D.V.* Kurs analiticheskoy geometrii i lineynoy algebry [A course in analytic geometry and linear algebra]. Moscow: Vyssh. shk., 1998, 320 p.
17. *Levin I.I., Dordopulo A.I., Sorokin D.A., Kalyaev Z.V., Doronchenko Yu.I.* Rekonfiguriruemye komp'yutery na osnove plis Xilinx Virtex Ultrascale [Xilinx Virtex Ultrascale PLD-based reconfigurable computers], *V sb.: Parallel'nye vy-chislitel'nye tekhnologii (PaVT'2019). Korotkie stat'i i opisaniya plakatov XIII Mezh-dunarodnoy nauchnoy konferentsii* [Parallel computational technologies (PCT 2019)], 2019, pp. 288-298.
18. *Dordopulo A.I., Levin I.I.* Metody reduktsii vychisleniy dlya programmirovaniya gibridnykh rekonfiguriruemykh vychislitel'nykh sistem [Computation reduction methods for programming hybrid reconfigurable computing systems], XII mul'tikonferentsiya po problemam upravleniya (MKPU-2019): Mater. konferentsii [Materials of the XII multiconference on management problems 2019]: in 4 vol., 2019, pp. 78-82.
19. *Kolmogorov A.N., Fomin S.V.* Elementy teorii funktsiy i funktsional'nogo analiza [Elements of the theory of functions and functional analysis]. 4th ed. Moscow: Nauka, 1976, 544 p.
20. *Levin I.I., Pelipets A.V.* Effektivnaya realizatsiya rasparallelivaniya na rekonfiguriruemykh sistemakh [Efficient Parallel Execution on Reconfigurable Systems], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Herald of computer and information technologies], 2018, No. 8, pp. 11-16.

Статью рекомендовал к опубликованию д.т.н. Э.В. Мельник.

Чекина Мария Дмитриевна – Южный федеральный университет; e-mail: chekina @superevm.ru; 347900, г. Таганрог, пер. Итальянский, 106; тел.: +79281541526; аспирант.

Chekina Marija Dmitrievna – Southern Federal University; e-mail: chekina @superevm.ru; 106, Italyansky lane, Taganrog, 347900, Russia; phone: +79281541526; graduate student.

Раздел IV. Проблемно-ориентированные и встраиваемые вычислительные системы

УДК 621.398 DOI 10.18522/2311-3103-2020-7-143-151

О.В. Ершова, Е.В. Кириченко, М.С. Кочерга, Е.А. Семерников

МАСШТАБИРОВАНИЕ ЦЕЛОЧИСЛЕННЫХ ДАННЫХ В РВС ПРИ ВЫЧИСЛЕНИИ РАДИОЛОКАЦИОННОГО ДАЛЬНОСТНО-СКОРОСТНОГО ПОРТРЕТА

Данная статья посвящена вопросу предотвращения переполнений разрядной сетки в высокопроизводительных реконфигурируемых вычислительных системах (РВС) на основе ПЛИС, приводящих к фатальным ошибкам обработки данных в процессе получения радиолокационного дальностно-скоростного портрета (ДСП) цели. Кратко рассмотрены существующие способы решения данной проблемы, и предложена методика априорного определения количества точек масштабирования в конвейерно-параллельных вычислительных структурах, формирующих радиолокационный ДСП цели. Данная методика позволяет заранее определить необходимое количество масштабирований на всех этапах обработки целочисленных данных и предотвратить переполнения при вычислении БПФ (ОБПФ) во всех возможных ситуациях. Рассмотрен алгоритм получения ДСП из исходной сигнальной матрицы (ИСМ) на примере радиолокационной системы (РЛС) непрерывного излучения с линейной частотной модуляцией (ЛЧМ). Приведены формулы, позволяющие рассчитать максимально допустимое значение (в используемой разрядной сетке) амплитуды преобразумых сигналов на всех этапах получения ДСП и количество итераций с масштабированием в процедурах БПФ (ОБПФ). Представлен численный пример расчета количества масштабирований для всех этапов алгоритма формирования ДСП, в котором определено необходимое число итераций с масштабированием при вычислении быстрой свертки и доплеровской скорости (с учетом умножения на оконную функцию), позволяющее предотвратить возможный выход значений сигнала за пределы разрядной сетки. В результате установлено, что предлагаемый способ расчета количества масштабирований позволяет избежать чрезмерного падения уровня сигнала на выходе обработки и снизить отношение ошибок цифровой обработки к уровню сигнала дальностно-скоростной матрицы.

Масштабирование; дальностно-скоростной портрет; быстрое преобразование Фурье; реконфигурируемые вычислительные системы; цифровая обработка сигналов; линейная частотная модуляция.

O.V. Ershova, E.V. Kirichenko, M.S. Kocherga, E.A. Semernikov

SCALING OF INTEGER DATA IN RECONFIGURABLE COMPUTER SYSTEMS WHEN CALCULATING A RADAR RANGE-VELOCITY PORTRAIT

This article deals with the question of preventing overflows of the bit grid in high-performance reconfigurable computing systems based on FPGAs, leading to fatal data processing errors when obtaining a radar range-velocity portrait of the target. The known methods of solving this problem are briefly considered. A new method for a priori determination of the number of scaling points in pipeline-parallel computing structures that form the target's radar range-velocity portrait is proposed. This technique allows to determine in advance the required number of scaling points at all stages of integer data processing and to prevent overflows when calculating the

FFT (IFFT) in all possible situations. An algorithm of forming of range-velocity portrait from an initial signal matrix is considered by the example of a continuous-wave radar system with linear frequency modulation (LFM). Formulas for calculating the maximum value of the amplitude of the converted signals at all stages of obtaining the range-velocity portrait and the number of iterations with scaling in the FFT (IFFT) procedures are given. A numerical example of calculating the number of scaling points for all stages of the algorithm of range-velocity portrait formation is presented. In the example the required number of iterations with scaling is determined when calculating the fast convolution and Doppler velocity (taking into account multiplication by the window function). It allows to prevent signal values from going beyond the bounds of the bit grid. As a result, it was found that the proposed method for calculating the number of scaling points avoids an excessive drop in the signal level at the processing output and reduces the ratio of digital processing errors to the signal level of the range-velocity matrix.

Scaling; range-velocity portrait; fast Fourier transform; reconfigurable computer systems; digital signal processing; linear frequency modulation.

Введение. Высокопроизводительные реконфигурируемые вычислительные системы (РВС) на основе ПЛИС все чаще находят применение в радиолокации, где требуется обрабатывать большие объемы данных с высоким темпом оцифровки в реальном масштабе времени [1–4].

Одним из примеров такого применения являются РЛС непрерывного излучения с линейной частотной модуляцией (ЛЧМ), широко используемые в современной радиолокации для обнаружения движущихся объектов [5–12]. В таких системах в качестве излучаемого сигнала используется периодическая последовательность непрерывно излучаемых ЛЧМ-импульсов с периодом T_p. Соответственно эхо-сигнал, принимаемый от объекта, также будет представлять собой периодическую последовательность ЛЧМ-импульсов, каждый из которых имеет сдвиг несущей частоты на величину доплеровского смещения частоты и временную задержку τ относительно излученного импульса. Закон изменения частоты излучаемого и принимаемого сигналов представлен на рис 1.

Основными задачами обработки данных в таких РЛС являются обнаружение и оценка параметров движущихся объектов в координатах дальность-скорость [12].

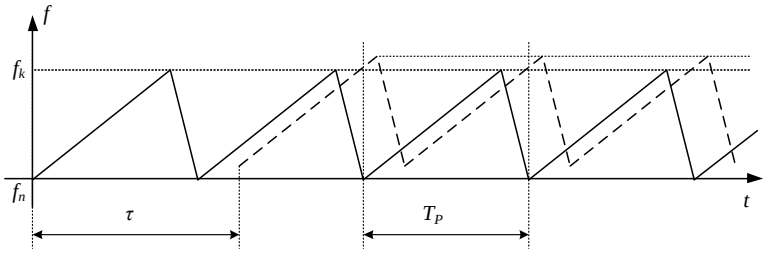


Рис. 1. Закон изменения частоты излучаемого (сплошная линия) и принимаемого (пунктир) сигналов

Для решения задачи оценки дальности и скорости объекта формируется исходная сигнальная матрица (ИСМ), каждая строка которой представляет собой отсчеты отраженного сигнала на интервале приема, равном T_p, а количество строк матрицы рассчитывается из условия обеспечения требуемых характеристик обнаружения. В процессе когерентной обработки принимаемых сигналов с использованием алгоритмов БПФ и ОБПФ происходит преобразование ИСМ в дальностно-скоростной портрет (ДСП), анализируя который можно обнаруживать объекты и оценивать их дальности и скорости.

Поскольку формирование дальностно-скоростного портрета происходит в режиме реального времени и с высокой частотой дискретизации сигналов, то для ускорения расчетов цифровая обработка сигналов производится в целочисленной арифметике. В то же время, согласно теореме Парсеваля, при вычислении дискретного быстрого преобразования Фурье происходит увеличение энергии сигнала. Поэтому в процессе получения ДСП возникает вопрос о предотвращении переполнений разрядной сетки, приводящих к фатальным ошибкам обработки [13, 14].

В общем случае проблема недопущения переполнения решается либо введением обязательного масштабирования на каждом этапе преобразования Фурье, либо с помощью приемов, сводящихся к гибридной плавающей точке [15–19]. Первый способ решения проблемы приводит к возрастанию ошибок цифровой обработки сигналов и уменьшению отношения сигнал/шум на выходе системы обработки. Второй способ требует контроля переполнений на каждой арифметической операции, что снижает быстродействие и усложняет аппаратное обеспечение системы обработки. Более подробно с этими способами можно познакомиться в [20].

Оптимальным решением было бы определение заранее минимального количества масштабирований на всех этапах обработки данных, необходимого и достаточного для отсутствия переполнений. Решению данного вопроса в процессе получения ДСП посвящена эта статья.

Постановка задачи. Для решения вопроса о необходимом и достаточном количестве масштабирований при формировании ДСП рассмотрим алгоритм его получения из ИСМ. Суть алгоритма заключается в следующем. Сначала производится согласованная фильтрация сигналов в каждой строке ИСМ методом быстрой свертки с применением алгоритмов БПФ и ОБПФ. Затем для каждого канала дальности (каждого столбца преобразуемой матрицы) производится узкополосная доплеровская фильтрация с помощью БПФ, которому предшествует умножение столбца на оконную функцию.

Таким образом, преобразование ИСМ в ДСП содержит следующие основные этапы обработки.

1. БПФ по всем строкам ИСМ.
2. Умножение на эталонную функцию $CF(i)$ всех строк матрицы после этапа 1.
3. ОБПФ по всем строкам матрицы после этапа 2.
4. Умножение всех столбцов матрицы на оконную функцию $W(n)$.
5. БПФ по всем столбцам.

Ниже на рис. 2 показана блок-схема преобразования ИСМ в ДСП.

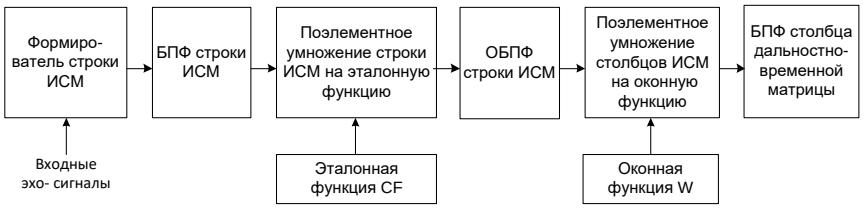


Рис. 2. Блок схема формирования ДСП

В данной статье при решении вопроса о количестве масштабирований на различных этапах формирования ДСП будем исходить из следующих допущений:

♦ зондирующий сигнал является непрерывным периодическим сигналом с периодом повторения T_p, в каждом периоде которого излучается импульс с линейной частотной модуляцией (ЛЧМ) в полосе частот ΔF;

♦ каждая строка сигнальной матрицы содержит сумму отраженных от разных объектов непрерывных сигналов на интервале повторения T_p; при этом каждый отдельный эхосигнал в строке в общем случае состоит из двух частей, представляющих собой окончание и начало двух смежных периодов отраженных ЛЧМ-импульсов (см. рис. 1);

♦ длина строки сигнальной матрицы N_1 согласована с длительностью периода излучения ЛЧМ-импульсов и соответветствует обрабатываемому диапазону дальностей;

♦ одиночному отраженному ЛЧМ сигналу в отсутствие помех в ДСП соответствует точечный отклик как по координате дальности, так и по координате скорости.

Количество итераций с масштабированием в процедурах БПФ и ОБПФ рассчитывается, исходя из требования, что амплитуда преобразумых сигналов на всех этапах получения ДСП не должна превышать максимально допустимое значение в используемой разрядной сетке. При расчете будем опираться на ранее написанные статьи [21, 22], в которых рассматривались вопросы определения неоходимого количества этапов с масштабированием при вычислении БПФ и при быстрой свертке.

Оценка количества итераций с масштабированием при вычислении быстрой свертки по дальности (этапы 1-3). Ранее авторами в статьях [21, 22] было показано, что количество масштабирований в процедурах БПФ и ОБПФ быстрой свертки зависит от энергии сигнала, отношения полосы частот сигнала к частоте дискретизации и от размера БПФ. Однако при расчете не учитывалось влияние энергии спектра эталонного сигнала, с которым производится свертка. Оценим влияние этого фактора на количество итераций с масштабированием.

Для оценки уровня получаемого спектра на этапе 1 исходим из допущений, что размер как принятого, так и излученного сигнала в отсчетах соответствует N_1 – длине строки сигнальной матрицы, а спектр принятого сигнала равномерно распределен в полосе частот ΔF. Значение N_1 равно степени двойки. Энергия принятого сигнала в строке, вычисленная во временной области, после БПФ, согласно теореме Персеваля, возрастает в N_1 раз

$$N_1 \cdot E_0 = E_1, \tag{1}$$

где E_0, E_1 – энергии сигнала до и после БПФ, измеренные во временной и в частотных областях соответственно.

Так как нашей задачей является получение количества масштабирований, достаточное для предотвращения переполнений при вычислении БПФ (ОБПФ) во всех возможных ситуациях, то для получения верхних оценок уровня сигнала на выходе БПФ (ОБПФ) в равенство (1) следует подставлять максимально возможные на этом этапе значения входного сигнала.

Обозначим M_p максимальные уровни сигнала, получаемые в отсутствии масштабирований на p-м этапе обработки, а A_1, A_2 – максимально возможные уровни вещественных сигналов, представимых в разрядной сетке для данных ИСМ и данных ДСП соответственно. Тогда, учитывая, что спектр сигнала сосредоточен в полосе частот ΔF, из уравнения (1) получим

$$N_1 \cdot (N_1 \cdot A_1^2) = \frac{\Delta F}{F_d} \cdot N_1 \cdot M_1^2$$

и далее

$$M_1 = \sqrt{\frac{F_d}{\Delta F} \cdot N_1} \cdot A_1,$$

где F_d – частота дискретизации.

Так как значение M_1 показывает средний уровень спектра в полосе, то возникает необходимость учета возможной неравномерности спектра. Для этого можно ввести корректирующий множитель $K > 1$, который для ЛЧМ сигнала можно принять равным 1,2. Учитывая фактор неравномерности спектра, можно определить количество итераций БПФ с масштабированием для блока 1

$$R_1 = min\left\{log_2(N_1),\ \ max\left[0,\ \ ceil\left[log_2\left(\frac{K \cdot \sqrt{\frac{F_d}{\Delta F} N_1} \cdot A_1}{A_2}\right)\right]\ \right]\right\}. \quad (2)$$

Максимальноое значение энергии сигнала в строке после БПФ с количеством масштабирований R_1 станет равным $N_1 \cdot N_1 \cdot A_1^2 \cdot \frac{1}{2^{2 \cdot R_1}}$.

Для расчета количества итераций ОБПФ с масштабированием R_3 сначала требуется определить максимально возможное значение на выходе свертки. Этому случаю соответствует полное тождество отраженного и излучаемого сигналов, в результате чего на выходе свертки получается единственный пик, местоположение которого соответствует дальности обнаружения объекта. Уровень этого пика можно рассчитать из неравенства Коши–Буняковского. Из этого неравенства следует, что величина квадрата модуля произвольного m-го отсчета $D(m)$ в строке после ОБПФ с учетом умножения спектра $S(i)$ на эталонную функцию $CF(i)$ не может превосходить значение произведения их энергий

$$|D(m)|^2 = \left|\sum_{i=0}^{N_1-1} CF(i) \cdot e^{\frac{j2\pi mi}{N_1}} \cdot S(i)\right|^2 \leq \sum_{i=0}^{N_1-1} |CF(i)|^2 \cdot \frac{N_1^2 \cdot A_1^2}{2^{2 \cdot R_1}}.$$

Будем считать, что значения эталонной функции $CF(i)$ нормированы к единице. Влияние эталонной функции можно учесть с помощью корректирующего множителя C_{CF}

$$C_{CF} = \sum_{i=0}^{N_1-1} |CF(i)|^2.$$

Таким образом, в отсутствие масштабирований максимально достижимый уровень сигнала после свертки по дальности на этапе 3 можно определить из соотношения

$$M_3 = \frac{N_1 \cdot \sqrt{C_{CF}} \cdot A_1}{2^{R_1}}.$$

Определив M_3, можно легко рассчитать количество итераций с масштабированием в ОБПФ согласно формуле

$$R_3 = min\left\{log_2(N_1),\ \ max\left(0,\ \ ceil\left[log_2\left(\frac{N_1 \cdot A_1 \cdot \sqrt{C_{CF}}}{A_2 \cdot 2^{R_1}}\right)\right]\right)\right\}. \quad (3)$$

В результате масштабирования ОБПФ максимально возможное значение модуля сигнала в преобразуемой матрице после этапа 3 составит $\frac{N_1 \cdot \sqrt{C_{CF}} \cdot A_1}{2^{(R_1+R_3)}}$.

Оценка количества итераций с масштабированием при определении доплеровской скорости. Доплеровская скорость объекта рассчитывается путем вычисления преобразования Фурье для столбцов матрицы ДСП. При этом в целях лучшего разрешения по частоте и уменьшения негативного эффекта подавления сильным сигналом слабых сигналов на этапе 4 (см. рис. 1) следует произвести умножение сигнала в столбце на оконную функцию $W(n), n = \overline{0, N_2 - 1}$. Значения оконной функции нормированы к единице.

В статьях [21, 22] авторами рассматривался вопрос определения необходимого и достаточного количества итераций с масштабированием при вычислении БПФ, но без учета умножения входного сигнала на оконную функцию. Здесь произведем оценку количества итераций с масштабированием, во-первых, с учетом умножения на оконную функцию и, во-вторых, основываясь на ранее полученном максимально возможном значении сигнала в столбце.

Определим максимально возможное значение модуля спектральной составляющей матрицы ДСП после БПФ по столбцу (этап 5), воспользовавшись неравенством Коши – Буняковского. Ранее было показано, что максимально возможное значение модуля сигнала в преобразуемой матрице после блока 3 составит $M_3 = \frac{N_1 \cdot \sqrt{C_{CF}} \cdot A_1}{2^{(R_1+R_3)}}$. Из неравенства Коши – Буняковского следует, что квадрат модуля спектральной составляющей $Y(k)$ на этапе 5 после БПФ по столбцу, значения которого равны $P(n) \cdot W(n)$, не может превосходить значение произведения их энергий:

$$|Y(k)|^2 = \left| \sum_{n=0}^{N_2-1} P(n) \cdot e^{j2\pi kn/N_2} \cdot W(n) \right|^2 \le C_W \cdot N_2 \cdot \frac{N_1^2 \cdot C_{CF} \cdot A_1^2}{2^{2\cdot(R_1+R_3)}},$$

где C_W – энергия оконной функции:

$$C_W = \sum_{n=1}^{N_2} |W(n)|^2.$$

Отсюда можно рассчитать количество масштабирований на этапе 5 вычисления доплеровской скорости

$$R_5 = min\left\{ log_2(N_2),\ max\left[0,\ ceil\left[log_2\left(\frac{N_1 \cdot A_1 \cdot \sqrt{N_2 \cdot C_{CF} \cdot C_W}}{A_2 \cdot 2^{R_1+R_3}}\right)\right]\right]\right\} \quad (4)$$

Пример. В качестве примера был произведен расчет количества итераций с масштабированием согласно формулам (2) - (4) при следующих параметрах: $A_1 = 2^{15},\ A_2 = 2^{17},\ N_1 = 2^{14},\cdot N_2 = 2^{10},\ \Delta F/F_d = 0.8$, оконная функция – окно Хэмминга. Был получен следующий результат: $R_1 = 6, R_3 = 13, R_5 = 9$, позволяющий гарантированно предотвратить возможный выход значений сигнала за пределы разрядной сетки. Сравнивая полученное суммарное количество итераций с масштабированием, равное 28, с количеством масштабированией на каждой итерации БПФ и ОБПФ, равным 38, видим, что предлагаемый способ расчета количества масштабирований позволяет избежать чрезмерного падения уровня сигнала на выходе обработки в 1024 раз и тем самым снизить отношение ошибок цифровой обработки к уровню сигнала дальностно-скоростной матрицы.

Заключение. Предлагаемая методика, используемая при организации высокопроизводительных систем ЦОС, позволяет априори определить количество точек масштабирования (деления на 2) в конвейерно-параллельных вычислительных структурах, формирующих матрицу для обнаружения подвижных объектов в координатах дальность – скорость.

Такое априорное определение необходимого и достаточного количества масштабирований на всех этапах обработки данных является оптимальным решением проблемы недопущения переполнений разрядной сетки, приводящих к фатальным ошибкам обработки.

Применение предложенной методики в системах ЦОС позволяет не только исключить появление избыточных масштабирований, ведущих к чрезмерному падению уровня сигнала на выходе обработки и, соответственно, к уменьшению отношения сигнал/шум, но также избежать снижения быстродействия и усложнения аппаратного обеспечения системы обработки.

Данная методика была использована на практике в реальном изделии и дала положительные результаты.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Каляев И.А., Левин И.И.* Многопроцессорные вычислительные системы (суперкомпьютеры): состояние и перспективы // Вестник компьютерных и информационных технологий. – 2004. – № 6. – С. 25-44.
2. *Дмитренко Н.Н., Каляев И.А., Левин И.И., Семерников Е.А.* Развитие аппаратурной платформы реконфигурируемых вычислительных систем // Тр. Международной суперкомпьютерной конференции: Научный сервис в сети Интернет: суперкомпьютерные центры и задачи. (Новороссийск, 20-25 сентября 2010 г.). – М.: Изд-во МГУ, 2010. – С. 315-320.
3. *Каляев И.А., Левин И.И., Семерников Е.А., Шмойлов В.И.* Реконфигурируемые мультиконвейерные вычислительные структуры. – Ростов-на-Дону: Изд-во ЮНЦ РАН, 2009. – 320 с.
4. *Дордопуло А.И., Каляев И.А., Левин И.И., Семерников Е.А.* Высокопроизводительные реконфигурируемые вычислительные системы нового поколения // Тр. 13-й Международной суперкомпьютерной конференции: Научный сервис в сети Интернет: экзафлопсное будущее. (Новороссийск, 19-24 сентября 2011 г.). – М.: Изд-во МГУ, 2011. – С.42-49.
5. Радиолокационные сигналы для аэрокосмических, оборонных и автомобильных РЛС: сайт. – URL: https://www.astrosoft.ru/articles/radar/radiolokatsionnye-signaly-dlya-aerokosmicheskikh-oboronnykh-i-avtomobilnykh-rls/ (дата обращения: 20.10.2020).
6. *Asuzu P., Thompson C.* A more exact linear FMCW radar signal model for simultaneous range-velocity estimation // IEEE Radar Conference (RadarConf18). – 2018. – P. 7-11.
7. *Skolnik, Merrill I.* Introduction of Radar Systems, McGraw-Hill Inc. – ISBN - 978007118189, 1962.
8. *Stove A.G.* Linear FMCW radar techniques // Radar and Signal Processing, IEE Proceedings F. – Oct. 1992. – Vol. 139, Issue 5. – P. 343-350.
9. Дальностно-скоростной портрет (обнаружение) в автомобильных РЛС с непрерывным ЛЧМ-сигналом: сайт. – URL: https://www.astrosoft.ru/articles/radar/ dalnostno-skorostnoy-portret-obnaruzhenie-v-avtomobilnykh-rls-s-nepreryvnym-lchm-signalom/ (дата обращения: 08.11.2020).
10. *Folster F., Rohling H., Lubbert U.* An Automotive radar network based on 77GHz FMCW sensors // Radar Conference, 2005 IEEE International, 9-12 May 2005. – P. 871-876.
11. *Mende R., Rohling H.* A High-Performance AICC radar and results with an experimental vehicle // Radar 97 (Conf. Publ. No. 449), 14.-16.Oct 1997. – P. 21-25.
12. *Geroleo F.G., Brandt-Pearce M.* Detection and Estimation of Multi-Pulse LFMCW Radar Signals // IEEE Transactions on Aerospace and Electronic Systems. – Jan. 2012. – Vol. 48, Issue 1. – P. 405-418.
13. *Титов Д.А., Василевский В.В., Косых А.В.* Цифровая обработка сигналов: Методические указания к лабораторным работам. (Омск-2011). – URL: https://omgtu.ru/general_ information/faculties/radio_engineering_department/department_quot_radio_devices_and_diagnostic_systems_quot/educational-materials/Digital_signal_processing/Methodical_instructions_to_laboratory_works.pdf. (дата обращения: 08.11.2020).
14. Реализация целочисленного БПФ на ПЛИС: сайт. – URL: https://habr.com/ru/post/420517/ (дата обращения: 08.11.2020).

15. *Рабинер Л., Гоулд Б.* Теория и применение цифровой обработки сигналов. – М.: Мир, 1978. – 848 с.
16. *Лайонс Р.* Цифровая обработка сигналов. – М.: Бином-Пресс, 2011. – 654 с.
17. *Оппенгейм А., Шафер Р.* Цифровая обработка сигналов. – М.: Техносфера, 2012. – 1048 с.
18. *Айфичер Э.С., Джервис Б.У.* Цифровая обработка сигналов: практический подход. – 2-е изд. – М.: Вильямс, 2016. – 992 с.
19. *Смит С.* Цифровая обработка сигналов. Практическое руководство для инженеров и научных работников: перев. с англ. – М.: Додэка XXI, 2008. – 720 с.
20. LogiCORE IP Fast Fourier Transform v9.0. Product Guide. Vivado Design Suite. PG109, October 4, 2017. – URL: http://www.xilinx.com/support/documentation/ip_documentation/xfft/v9_0/pg109-xfft.pdf. (дата обращения: 08.11.2020).
21. *Ершова О.В., Кириченко Е.В., Семерников Е.А., Чкан А.В.* Управление масштабированием данных в процедуре согласованной фильтрации в зависимости от энергии сигнала // Матер. 8-й Всероссийской мультиконференции по проблемам управления МКПУ. – Ростов-на-Дону: Изд-во ЮФУ, 2015. – Т. 3. – С. 92-96.
22. *Ершова О.В., Кириченко Е.В., Семерников Е.А., Чкан А.В.* Масштабирование данных с фиксированной точкой в процедуре быстрой свертки // Радиотехника. – 2015. – № 4. – С. 66-72.

REFERENCES

1. *Kalyaev I.A., Levin I.I.* Mnogoprotsessornye vychislitel'nye sistemy (superkomp'yutery): sostoyanie i perspektivy [Multiprocessor computing systems (supercomputers): state and prospects], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Bulletin of computer and information technologies], 2004, No. 6, pp. 25-44.
2. *Dmitrenko N.N., Kalyaev I.A., Levin I.I., Semernikov E.A.* Razvitie apparaturnoy platformy rekonfiguriruemykh vychislitel'nykh sistem [Development of a hardware platform for reconfigurable computing systems], *Tr. Mezhdunarodnoy super-komp'yuternoy konferentsii: Nauchnyy servis v seti Internet: superkomp'yuternye tsentry i zadachi. (Novorossiysk, 20-25 sentyabrya 2010 g.)* [Proceedings of the International Supercomputer Conference: Scientific service on the Internet: supercomputer centers and tasks (Novorossiysk, September 20-25, 2010)]. Mosow: Izd-vo MGU, 2010, pp. 315-320.
3. *Kalyaev I.A., Levin I.I., Semernikov E.A., SHmoylov V.I.* Rekonfiguriruemye mul'tikonveyernye vychislitel'nye struktury [Reconfigurable multicore computational structures]. Rostov-on-Don: Izd-vo YuNTS RAN, 2009, 320 p.
4. *Dordopulo A.I., Kalyaev I.A., Levin I.I., Semernikov E.A.* Vysokoproizvoditel'nye rekonfiguriruemye vychislitel'nye sistemy novogo pokoleniya [High-performance reconfigurable computing systems of a new generation], *Tr. 13-y Mezhdunarodnoy superkomp'yuternoy konferentsii: Nauchnyy servis v seti Internet: ekzaflopsnoe budushchee. (Novorossiysk, 19-24 sentyabrya 2011 g.)* [Proceedings of the 13th International Supercomputer Conference: Scientific Service on the Internet: Exaflops Future. (Novorossiysk, September 19-24, 2011)]. Moscow: Izd-vo MGU, 2011, pp. 42-49.
5. Radiolokatsionnye signaly dlya aerokosmicheskikh, oboronnykh i avtomobil'nykh RLS: sayt [Radar signals for aerospace, defense and automotive radars: website]. Available at: https://www.astrosoft.ru/articles/radar/radiolokatsionnye-signaly-dlya-aerokosmicheskikh-oboronnykh-i-avtomobilnykh-rls/ (accessed 20 October 2020).
6. *Asuzu P., Thompson C.* A more exact linear FMCW radar signal model for simultaneous range-velocity estimation, *IEEE Radar Conference (RadarConf18)*, 2018, pp. 7-11.
7. *Skolnik, Merrill I.* Introduction of Radar Systems, McGraw-Hill Inc. ISBN - 978007118189, 1962.
8. *Stove A.G.* Linear FMCW radar techniquesб *Radar and Signal Processing, IEE Proceedings F*, Oct. 1992, Vol. 139, Issue 5, pp. 343-350.
9. Dal'nostno-skorostnoy portret (obnaruzhenie) v avtomobil'nykh RLS s nepreryvnym LCHM-signalom: sayt [Range-velocity portrait in car radars with continuous chirp signal: website]. Available at: https://www.astrosoft.ru/articles/radar/ dalnostno-skorostnoy-portret-obnaruzhenie-v-avtomobilnykh-rls-s-nepreryvnym-lchm-signalom/ (accessed 08 November 2020).
10. *Folster F., Rohling H., Lubbert U.* An Automotive radar network based on 77GHz FMCW sensors, *Radar Conference, 2005 IEEE International, 9-12 May 2005*, pp. 871-876.

11. *Mende R., Rohling H.* A High-Performance AICC radar and results with an experimental vehicle, *Radar 97 (Conf. Publ. No. 449), 14.-16.Oct 1997*, pp. 21-25.
12. *Geroleo F.G., Brandt-Pearce M.* Detection and Estimation of Multi-Pulse LFMCW Radar Signals, *IEEE Transactions on Aerospace and Electronic Systems*, Jan. 2012, Vol. 48, Issue 1, pp. 405-418.
13. *Titov D.A., Vasilevskiy V.V., Kosykh A.V.* TSifrovaya obrabotka signalov: Metodicheskie ukazaniya k laboratornym rabotam (Omsk-2011) [Digital signal processing: Methodological instructions for laboratory work. (Omsk-2011)]. Available at: https://omgtu.ru/general_ information/faculties/radio_engineering_department/department_quot_radio_devices_and_diagnostic_systems_quot/educational-materials/Digital_signal_processing/Methodical_instructions_to_laboratory_works.pdf. (accessed 08 November 2020).
14. Realizatsiya tselochislennogo BPF na PLIS: sayt [Implementation of integer FFT on FPGA: website]. Available at: https://habr.com/ru/post/420517/ (accessed 08 November 2020).
15. *Rabiner L., Gould B.* Teoriya i primenenie tsifrovoy obrabotki signalov [Theory and application of digital signal processing]. Moscow: Mir, 1978, 848 p.
16. *Layons R.* Tsifrovaya obrabotka signalov [Digital signal processing]. Moscow: Binom-Press, 2011, 654 p.
17. *Oppengeym A., Shafer R.* TSifrovaya obrabotka signalov [Digital signal processing]. Moscow: Tekhnosfera, 2012, 1048 p.
18. *Ayficher E.S., Dzhervis B.U.* TSifrovaya obrabotka signalov: prakticheskiy podkhod [Digital signal processing: a practical approach]. 2nd ed. Moscow: Vil'yams, 2016, 992 p.
19. *Smit S.* Tsifrovaya obrabotka signalov. Prakticheskoe rukovodstvo dlya inzhenerov i nauchnykh rabotnikov [Digital signal processing. A Practical Guide for Engineers and Scientists]: transl. from engl. Moscow: Dodeka XXI, 2008, 720 p.
20. LogiCORE IP Fast Fourier Transform v9.0. Product Guide. Vivado Design Suite. PG109, October 4, 2017. Available at: http://www.xilinx.com/support/documentation/ip_ documentation/ xfft/v9_0/pg109-xfft.pdf. (accessed 08 November 2020).
21. *Ershova O.V., Kirichenko E.V., Semernikov E.A., Chkan A.V.* Upravlenie masshtabirovaniem dannykh v protsedure soglasovannoy fil'tratsii v zavisimosti ot energii signala [Data scaling control in the matched filtering procedure depending on the signal energy], *Mater. 8-y Vserossiyskoy mul'tikonferentsii po problemam upravleniya MKPU* [Proceedings of the 8th All-Russian multiconference on management problems of the MCPU]. Rostov-on-Don: Izd-vo YuFU, 2015, Vol 3, pp. 92-96.
22. *Ershova O.V., Kirichenko E.V., Semernikov E.A., Chkan A.V.* Masshtabirovanie dannykh s fiksirovannoy tochkoy v protsedure bystroy svertki [Scaling of fixed-point data in a fast convolution procedure], *Radiotekhnika* [Radio engineering], 2015, No. 4, pp. 66-72.

Статью рекомендовал к опубликованию д.т.н. Э.В. Мельник.

Ершова Ольга Владимировна – ООО "НИЦ супер-ЭВМ и нейрокомпьютеров"; e-mail: ershova150681@mail.ru; 347900, г. Таганрог, пер. Итальянский, 106; тел.: 88634368177; программист 1 категории.

Кириченко Елена Викторовна – e-mail: e.v.kirichenko@yandex.ru; научный сотрудник.

Кочерга Максим Сергеевич – e-mail: regul105@list.ru; ведущий конструктор.

Семерников Евгений Андреевич – e-mail: semernikov@mvs.tsure.ru; к.т.н., зав. отделом ЦОС.

Ershova Olga Vladimirovna – "Scientific research center of supercomputers and neurocomputers" CoLtd; e-mail: ershova150681@mail.ru; 106, Italyansky lane, Taganrog, 347900, Russia; phone: +78634368177; programmer.

Kirichenko Elena Victorovna – e-mail: e.v.kirichenko@yandex.ru; research engineer.

Kocherga Maksim Sergeevich – e-mail: regul105@list.ru; leading engineer.

Semernikov Evgeniy Andreevich – e-mail: semernikov@mvs.tsure.ru; cand. of eng. sc.; head of department of Digital Signal Processing.

УДК 004.021 DOI 10.18522/2311-3103-2020-7-152-163

А.В. Чкан

ПОВЫШЕНИЕ РЕАЛЬНОЙ ПРОИЗВОДИТЕЛЬНОСТИ РВС ПРИ РЕШЕНИИ ЗАДАЧ ЦИФРОВОЙ ОБРАБОТКИ ИЗОБРАЖЕНИЙ С ИСПОЛЬЗОВАНИЕМ БЫСТРОГО ПРЕОБРАЗОВАНИЯ ФУРЬЕ

Рассматриваются вопросы цифровой обработки изображений больших размерностей в реальном масштабе времени с помощью реконфигурируемых вычислительных систем (РВС) на базе ПЛИС. РВС относятся к классу высокопроизводительных многопроцессорных вычислительных систем, но при этом обладают программируемой архитектурой, позволяющей конфигурировать структуру вычислительной системы, оптимально подстраивая её под алгоритмы решаемой задачи. В то же время оптимизация вычислительной структуры задачи сводится к разработке и реализации параллельных алгоритмов, соответствующих специфике используемой архитектуры РВС. Всё это позволяет эффективно использовать РВС для решения широкого класса задач цифровой обработки сигналов. Предложены способы повышения удельной и реальной производительности РВС при решении задач цифровой обработки изображений с использованием быстрого преобразования Фурье (БПФ). На примере процедуры фильтрации изображений в частотной области рассмотрены основные вычислительные этапы и способы их оптимизации, основанные на свойствах алгоритма БПФ. Применение оптимизации позволяет существенно сократить как объем вычислений, так и объем задействованных аппаратных ресурсов ПЛИС, и повысить производительность РВС для задач обработки изображений. Освобожденные в результате оптимизации вычислительной структуры ресурсы ПЛИС могут быть использованы для дополнительного распараллеливания вычислений и ускорения обработки поступающих данных. Показаны преимущества представления данных в формате с фиксированной запятой при выполнении расчётов на РВС. Использование фиксированной запятой позволяет не только повысить удельную и реальную производительность вычислительной системы по сравнению с плавающей запятой в силу свойств формата, но и использовать произвольную разрядность данных, что является актуальным для большинства задач цифровой обработки сигналов. Рассмотрено решение проблемы переполнения разрядной сетки при использовании формата с фиксированной запятой с помощью масштабирования разрядности данных.

Цифровая обработка сигналов; цифровая обработка изображений; быстрое преобразование Фурье; реконфигурируемые вычислительные системы; фиксированная запятая; масштабирование разрядности.

A.V. Chkan

IMPROVING REAL PERFORMANCE OF RCS WHEN SOLVING DIGITAL IMAGE PROCESSING TASKS USING FAST FOURIER TRANSFORM

The article discusses the issues of digital processing of images of large dimensions in real time using reconfigurable computer systems (RCS) on the base of programmable logic arrays (FPGAs). RCS belongs to the class of high-performance multiprocessor computing systems that have a programmable architecture that allows configuring the structure of a computer system and optimally adjusting it to the algorithms of the solved task. At the same time, optimization of the computational structure of the task reduced to the development and implementation of parallel algorithms corresponding to the specifics of the RCS architecture used. All this allows to effectively using RCS to solve a wide class of digital signal processing tasks. Offered are methods of increasing specific and real performance of RCS when solving digital image processing problems using fast Fourier transform (FFT). Using the example of a procedure for filtering images in the frequency domain, the main computational steps and methods for optimizing them based on the properties of the FFT algorithm are discussed. The use of optimization allows to significantly reducing both the amount of computation and the amount of hardware resources of the FPGA and

increase the performance of RCS for image processing tasks. The FPGA resources freed because of the optimization of the computational structure can be uses to further parallelize calculations and accelerate the processing of incoming data. The advantages of presenting data in fixed-point format when performing calculations on RCS are showed. The use of a fixed point allows not only to increase the specific and real performance of a computer system compared to a floating point due to the properties of the format, but also to use arbitrary data bit capacity, which is relevant for most digital signal processing tasks. The solution to the problem of overflow of the bit grid when using the fixed-point format using data bit scaling is discussed.

Digital signal processing; digital image processing; fast Fourier transform; reconfigurable computing systems; fixed point; bit scaling.

Введение. Цифровая обработка изображений больших размерностей в реальном масштабе времени является актуальной современной задачей [1–5], решение которой зачастую становится невозможным без высокопроизводительных многопроцессорных вычислительных систем [6-8]. Одними из активно развивающихся и перспективных направлений в области разработки МВС являются реконфигурируемые вычислительные системы (РВС) на базе ПЛИС [6, 7, 9]. Программируемая архитектура РВС даёт возможность конфигурировать собственную структуру, подстраивая её под алгоритмы решаемой задачи. При этом вычислительная структура самой задачи оптимизируется с точки зрения максимального распараллеливания информационных потоков в соответствии с особенностями используемой архитектуры РВС [10, 11]. Такой подход позволяет эффективно использовать РВС для построения высокопроизводительных систем цифровой обработки сигналов и изображений. Повышение производительности РВС, в свою очередь, зависит не только от технологических и архитектурных подходов построения системы, но и от математической оптимизации алгоритмической базы, используемой для решения поставленной задачи [6, 7].

Цифровая обработка изображений нередко связана с использованием дискретного преобразования Фурье (ДПФ) [1–5]. Использование свойств ДПФ применительно к обработке изображений позволяет существенно сократить объём вычислений и повысить реальную производительность вычислительных систем.

Другим важным аспектом, влияющим на производительность вычислительных систем, является выбор формата представления данных: с плавающей запятой или с фиксированной запятой [12–15]. Известно, что основными достоинствами использования формата с фиксированной запятой являются простота аппаратной реализации арифметических операций [13] и, как следствие, значительное уменьшение задействованного аппаратного ресурса ПЛИС в сравнении с использованием формата с плавающей запятой при реализации одной и той же вычислительной задачи. При этом удельная [16] и реальная производительности вычислительных систем на базе с фиксированной запятой существенно повышается, что связано как с более быстрым выполнением арифметических операций в формате с ФЗ, так и с возможностью размещения на сэкономленном аппаратном ресурсе ПЛИС дополнительных вычислительных структур. Ещё одним положительным моментом представления данных в формате с фиксированной запятой является возможность оперировать с данными произвольной разрядности на каждом из вычислительных этапов, что также приводит к экономии аппаратных ресурсов за счёт гибкой настройки системы по разрядностям входов и выходов.

Основными недостатками устройств на базе с фиксированной запятой являются [12, 15]:

- ограниченный динамический диапазон представления чисел;
- необходимость контроля промежуточных результатов арифметических операций на предмет возможных переполнений разрядной сетки;

♦ рост погрешности вычислений, связанной с округлением/усечением результатов арифметических операций при приведении их к требуемой разрядности представления данных на различных вычислительных этапах.

Поэтому использование формата с фиксированной запятой требует решения следующих задач:

♦ исключения ошибок переполнения разрядной сетки в процессе выполнения арифметических операций;

♦ снижения погрешности вычислений за счёт минимизации потерь значащих разрядов при округлении/усечении промежуточных результатов арифметических операций.

Важность решения этих задач связана с тем, что ошибки, возникающие на промежуточных этапах вычислений вследствие итерационного характера большинства алгоритмов ЦОС, могут приводить к накоплению вычислительных погрешностей, существенно снижающих точность получаемых решений.

Таким образом, основными способами повышения удельной и реальной производительности РВС при обработке изображений с помощью ДПФ являются:

♦ оптимизация вычислений на основе свойств ДПФ;

♦ представление данных в формате с фиксированной запятой.

Рассмотрим данные способы более подробно.

Оптимизация вычислений в задачах цифровой обработки изображений при использовании ДПФ. Покажем оптимизацию вычислений в задачах цифровой обработки изображений с использованием ДПФ на примере одной из распространённых и вычислительно трудоемких процедур – фильтрации изображений в частотной области [1–2, 17]. Данная процедура предназначена для улучшения, восстановления или изменения изображения в соответствии с поставленной целью и включает в себя следующие вычислительные этапы:

♦ расчёт спектра входного изображения $\dot{f}_{вх}(x,y)$ с помощью прямого двумерного ДПФ;

♦ умножение полученного спектра $\dot{F}(u,v)$ входного изображения на коэффициенты частотного фильтра $\dot{H}(u,v)$;

♦ расчёт обратного двумерного ДПФ $\dot{f}_{вых}(x,y)$ для полученных произведений $\dot{F}(u,v)\cdot\dot{H}(u,v)$.

Двумерный сигнал описывается двумерным рядом Фурье [1, 12, 13, 18]:

$$\dot{f}(x,y)=\frac{1}{M\,N}\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\dot{F}(u,v)\cdot\exp\left(j2\pi\left(\frac{u\,x}{M}+\frac{v\,y}{N}\right)\right), \qquad (1)$$

где $\dot{f}(x,y)$ – двумерное изображение; x = [0,…, M – 1] и y = [0,…, N – 1] – пространственные координаты; u = [0,…, M – 1] и v = [0,…, N – 1] – частотные координаты изображения.

ДПФ для такого сигнала имеет вид:

$$\dot{F}(u,v)=\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}\dot{f}(x,y)\cdot\exp\left(-j2\pi\left(\frac{u\,x}{M}+\frac{v\,y}{N}\right)\right). \qquad (2)$$

Обратное двумерное ДПФ определяется формулой (1).

Расчёт ДПФ для двумерного сигнала по формулам (1) и (2) осуществляется с помощью алгоритмов быстрого преобразования Фурье (БПФ) [19–21] сначала по строкам, а затем по столбцам двумерной матрицы изображений. На практике ре-

комендуется работать с квадратными изображениями, размер стороны которых равен степени двойки [1-2]. Для этого матрица изображения, в случае необходимости, дополняется достаточным количеством нулей. Далее, без нарушения общности, будут рассматриваться квадратные матрицы изображений.

Из формулы (2) видно, что для получения спектра изображения размером N x N потребуется расчёт $2 \cdot N$ алгоритмов БПФ. Реализация всей процедуры фильтрации изображений в частотной области потребует $4 \cdot N$ выполнений алгоритма БПФ (прямого и обратного БПФ).

Однако количество выполнений алгоритма БПФ при реализации процедуры фильтрации изображений можно существенно сократить, если учитывать тот факт, что все исходные данные (строки или столбцы матрицы изображения) представляют собой действительные последовательности, для которых справедливо свойство симметрии ДПФ [12, 13]

$$\mathrm{Re}[X(k)] = \mathrm{Re}[X(N-k)],$$
$$\mathrm{Im}[X(k)] = -\mathrm{Im}[X(N-k)].$$

Это свойство имеет важное практическое приложение для повышения вычислительной производительности в том случае, когда необходимо получить ДПФ нескольких действительных последовательностей одинаковой длины. Так, сгруппировав эти последовательности в пары, можно в одной процедуре ДПФ получить ДПФ двух последовательностей сразу. Рассмотрим применение этого свойства на примере двух действительных периодических последовательностей $x_1(n)$ и $x_2(n)$ с периодом N, n = 0, 1, …, N – 1. Сформируем комплексную последовательность $z(n)$ следующим образом:

$$z(n) = x_1(n) + j \cdot x_2(n).$$

Выполним процедуру ДПФ от последовательности $z(n)$:

$$Z(k) = \sum_{n=0}^{N-1} [x_1(n) + j \cdot x_2(n)] \cdot W_N^{nk}, \quad k = 0..N-1.$$

На основании свойства линейности получим:

$$Z(k) = X_1(k) + j \cdot X_2(k),$$

где $X_1(k)$ и $X_2(k)$ – спектры действительных последовательностей $x_1(n)$ и $x_2(n)$. Тогда для $k = 1,...,(N/2-1)$:

$$X_1(k) = \frac{1}{2} \cdot \left[Z(k) + Z\left(N-k\right) \right],$$
$$X_2(k) = -j \cdot \frac{1}{2} \cdot \left[Z(k) - Z\left(N-k\right) \right] \cdot$$

Поскольку действительные части $X_1(k)$ и $X_2(k)$ симметричны, а мнимые – антисимметричны, то их можно разделить, используя операции сложения и вычитания

$$\mathrm{Re}[X_1(k)] = \frac{1}{2}\left(\mathrm{Re}[Z(k)] + \mathrm{Re}[Z(N-k)]\right),$$
$$\mathrm{Im}[X_1(k)] = \frac{1}{2}\left(\mathrm{Im}[Z(k)] - \mathrm{Im}[Z(N-k)]\right),$$

$$\mathrm{Re}[X_2(k)] = \frac{1}{2}\left(\mathrm{Im}[Z(k)] + \mathrm{Im}[Z(N-k)]\right),$$

$$\mathrm{Im}[X_2(k)] = -\frac{1}{2}\left(\mathrm{Re}[Z(k)] - \mathrm{Re}[Z(N-k)]\right).$$

Для нулевой ($n = 0$) и центральной ($n = N / 2$) точек:

$$\mathrm{Re}[X_1(0)] = \mathrm{Re}[Z(0)]; \qquad \mathrm{Im}[X_1(0)] = 0;$$

$$\mathrm{Re}[X_2(0)] = \mathrm{Im}[Z(0)]; \qquad \mathrm{Im}[X_2(0)] = 0;$$

$$\mathrm{Re}[X_1(N/2)] = \mathrm{Re}[Z(N/2)]; \ \mathrm{Im}[X_1(N/2)] = 0;$$

$$\mathrm{Re}[X_2(N/2)] = \mathrm{Im}[Z(N/2)]; \qquad \mathrm{Im}[X_2(N/2)] = 0.$$

Вторую половину спектров $X_1(k)$ и $X_2(k)$ для $k = (N/2+1), ..., (N-1)$ можно получить по формулам

$$X_1(k) = X_1^*(N-k); \ \ X_2(k) = X_2^*(N-k),$$

где X_1^* и X_2^* – комплексно-сопряженные значения спектров X_1 и X_2.

Вычисление двумерного ДПФ по строкам матрицы входного изображения с помощью описанной выше процедуры позволит сократить количество операций сложения и умножения на данном этапе более чем на 48% для $N \geq 128$ [12]. При этом число расчётов алгоритма БПФ для строк сократится вдвое и составит $N/2$.

Следующим этапом обработки является расчёт двумерного ДПФ по столбцам матрицы изображения, где рассчитываемые данные будут представлены уже в комплексном виде.

В силу симметричности получаемых данных реальной части и антисимметричности мнимой расчёт алгоритма БПФ нужно выполнить только для (N/2+1) столбцов матрицы изображения. Остальные (N/2–1) столбцов, с (N/2+1) по N–1, будут являться расположенной в обратном порядке перевёрнутой копией столбцов с N/2–1 по 1-й и могут быть получены по формулам

$$X(0, N-k) = X^*(0, k), \quad k = \left[1:(N/2-1)\right];$$

$$X(N-n, N-k) = X^*(n, k), \quad n = [1:(N-1)], \quad k = \left[1:(N/2-1)\right].$$

В результате общее число выполнений алгоритма БПФ для расчёта спектра входного изображения составит: $(N + N)/2 + 1 = N + 1$, что практически в два раза меньше прямого подхода.

Выполнение обратного двумерного БПФ, осуществляемого после умножения полученного спектра изображения на действительные коэффициенты частотного фильтра, тоже может быть оптимизировано на этапе работы алгоритма обратного БПФ для строк изображения. В силу симметричности получаемых при проходе по строкам данных реальной части и антисимметричности мнимой расчёт алгоритма обратного БПФ нужно выполнить только для (N/2+1) строк матрицы изображения. Остальные (N/2–1) строк будут являться точной копией строк со 1-й по N/2–1, расположенных в обратном порядке (с N–1 по N/2 + 1) и могут быть получены по формулам

$$x(N-n, k) = x^*(n, k), \quad n = [1:(N/2-1)], \quad k = \left[0:(N-1)\right],$$

где x^* – комплексно-сопряженные значения получаемого изображения x.

Выполнение обратного БПФ для столбцов матрицы изображения приводит к получению отфильтрованного по частоте исходного изображения и симметричностью данных не обладает. Поэтому на этом этапе потребуется N вычислений алгоритма обратного БПФ.

Таким образом, общее число расчётов алгоритма БПФ для процедуры фильтрации изображений в частотной области составит

$$N + 1 + N / 2 + 1 + N = 2.5 \cdot N + 2$$

вместо $4 \cdot N$ при прямом подходе, что позволит повысить производительность РВС примерно в 1.6 раза для данной процедуры.

Ещё одной возможностью оптимизации БПФ является упрощение базовых операций первой (для БПФ с прореживанием по времени [12, 13]) или последней (для БПФ с прореживанием по частоте [12, 13]) итерации БПФ в связи с тем, что комплексные поворачивающие множители $\dot{W}_i$ для каждой базовой операции данных итераций БПФ представляют собой единицы в реальной части и нули в мнимой.

Так, общая формула для базовой операции БПФ с прореживанием по времени определяется формулами

$$\dot{A}_{i+1} = \dot{A}_i + \dot{B}_i \cdot \dot{W}_l,$$
$$\dot{B}_{i+1} = \dot{A}_i - \dot{B}_i \cdot \dot{W}_l.$$

Или для реальных (Re) и мнимых (Im) частей

$$\begin{aligned}
\mathrm{Re}\left[\dot{A}_{i+1}\right] &= \mathrm{Re}\left[\dot{A}_i\right] + \mathrm{Re}\left[\dot{B}_i\right] \cdot \mathrm{Re}\left[\dot{W}_l\right] - \mathrm{Im}\left[\dot{B}_i\right] \cdot \mathrm{Im}\left[\dot{W}_l\right], \\
\mathrm{Im}\left[\dot{A}_{i+1}\right] &= \mathrm{Im}\left[\dot{A}_i\right] + \mathrm{Re}\left[\dot{B}_i\right] \cdot \mathrm{Im}\left[\dot{W}_l\right] + \mathrm{Im}\left[\dot{B}_i\right] \cdot \mathrm{Re}\left[\dot{W}_l\right], \\
\mathrm{Re}\left[\dot{B}_{i+1}\right] &= \mathrm{Re}\left[\dot{A}_i\right] - \mathrm{Re}\left[\dot{B}_i\right] \cdot \mathrm{Re}\left[\dot{W}_l\right] + \mathrm{Im}\left[\dot{B}_i\right] \cdot \mathrm{Im}\left[\dot{W}_l\right], \\
\mathrm{Im}\left[\dot{B}_{i+1}\right] &= \mathrm{Im}\left[\dot{A}_i\right] - \mathrm{Re}\left[\dot{B}_i\right] \cdot \mathrm{Im}\left[\dot{W}_l\right] - \mathrm{Im}\left[\dot{B}_i\right] \cdot \mathrm{Re}\left[\dot{W}_l\right].
\end{aligned} \tag{3}$$

где $\dot{W}_l = \exp\left(-j\frac{2\pi l}{N}\right)$, $i, l \in [0..N/2-1]$, – комплексные поворачивающие множители. Таким образом, базовая операция включает в себя 8 операций умножений, 4 операции сложения и 4 операции вычитания.

После оптимизации для первой итерации БПФ с прореживанием по времени при $\mathrm{Re}\left[\dot{W}_l\right] = 1$ и $\mathrm{Im}\left[\dot{W}_l\right] = 0$ формулы (3) примут вид:

$$\begin{aligned}
\mathrm{Re}\left[\dot{A}_{i+1}\right] &= \mathrm{Re}\left[\dot{A}_i\right] + \mathrm{Re}\left[\dot{B}_i\right], \\
\mathrm{Im}\left[\dot{A}_{i+1}\right] &= \mathrm{Im}\left[\dot{A}_i\right] + \mathrm{Im}\left[\dot{B}_i\right], \\
\mathrm{Re}\left[\dot{B}_{i+1}\right] &= \mathrm{Re}\left[\dot{A}_i\right] - \mathrm{Re}\left[\dot{B}_i\right], \\
\mathrm{Im}\left[\dot{B}_{i+1}\right] &= \mathrm{Im}\left[\dot{A}_i\right] - \mathrm{Im}\left[\dot{B}_i\right],
\end{aligned} \tag{4}$$

что составит всего 2 операции сложения и 2 операции вычитания для каждой из $N/2$ базовых операций. Общее сокращение вычислительных операций составит $8 \cdot N / 2 = 4 \cdot N$ операций умножения, $4 \cdot N / 2 = 2 \cdot N$ операций сложения/вычитания.

Для алгоритма БПФ с прореживанием по частоте формулы базовой операции имеют вид:

$$\dot{A}_{i+1} = \dot{A}_i + \dot{B}_i,$$
$$\dot{B}_{i+1} = \left(\dot{A}_i - \dot{B}_i\right) \cdot \dot{W}_l.$$

Или для реальных (Re) и мнимых (Im) частей:

$$\begin{aligned}
&\mathrm{Re}\left[\dot{A}_{i+1}\right]=\mathrm{Re}\left[\dot{A}_i\right]+\mathrm{Re}\left[\dot{B}_i\right],\\
&\mathrm{Im}\left[\dot{A}_{i+1}\right]=\mathrm{Im}\left[\dot{A}_i\right]+\mathrm{Im}\left[\dot{B}_i\right],\\
&\mathrm{Re}\left[\dot{B}_{i+1}\right]=\left(\mathrm{Re}\left[\dot{A}_i\right]-\mathrm{Re}\left[\dot{B}_i\right]\right)\cdot\mathrm{Re}\left[\dot{W}_l\right]-\left(\mathrm{Im}\left[\dot{A}_i\right]-\mathrm{Im}\left[\dot{B}_i\right]\right)\cdot\mathrm{Im}\left[\dot{W}_l\right],\\
&\mathrm{Im}\left[\dot{B}_{i+1}\right]=\left(\mathrm{Re}\left[\dot{A}_i\right]-\mathrm{Re}\left[\dot{B}_i\right]\right)\cdot\mathrm{Im}\left[\dot{W}_l\right]+\left(\mathrm{Im}\left[\dot{A}_i\right]-\mathrm{Im}\left[\dot{B}_i\right]\right)\cdot\mathrm{Re}\left[\dot{W}_l\right].
\end{aligned} \tag{5}$$

Такая базовая операция включает в себя 4 операции умножения, 3 операции сложения и 5 операций вычитания.

После оптимизации для последней итерации БПФ с прореживанием по частоте при $\mathrm{Re}\left[\dot{W}_l\right]=1$ и $\mathrm{Im}\left[\dot{W}_l\right]=0$ формулы (5) примут вид формул (4), что сократит количество вычислительных операций: на $2\cdot N$ операций умножения и на $2\cdot N$ операций сложения/вычитания.

Как видно из полученных формул, такая оптимизация также вносит свой вклад в рост общей производительности при выполнении алгоритма БПФ.

Использование формата с фиксированной запятой для представления информации. В работе [22] были представлены методы создания параллельно-конвейерных программ для алгоритмов быстрого преобразования Фурье (БПФ) при реализации их на реконфигурируемых вычислительных системах (РВС) с использованием данных в формате с фиксированной запятой. Применение методов позволяет значительно сокращать аппаратные затраты ПЛИС и повышать удельную производительность РВС при решении различных задач цифровой обработки сигналов.

Первый метод основан на априорном поиске итераций БПФ, требующих масштабирования результатов арифметических операций, с целью исключения возможных переполнений разрядной сетки. В методе применяется управляемое масштабирование (битовый сдвиг), с помощью которого разрядности данных на выходах базовых операций БПФ всегда преобразуются к разрядностям входов, что снижает задействованные аппаратные ресурсы ПЛИС и значительно повышает удельную производительность РВС [23]. Использование метода сокращает число необоснованных операций масштабирования и связанных с ними погрешностей вычислений, однако не всегда обеспечивает требуемую точность в задачах с большим числом входных отсчётов.

Второй метод программной реализации БПФ отличается поэтапным увеличением разрядности представления результатов базовых операций на один бит на каждой ступени вычислительного конвейера. Значения, получаемые на выходе последней ступени конвейера, масштабируются с целью приведения текущей разрядности данных к разрядности входов первой ступени для расчёта последующих итераций БПФ на одной и той же вычислительной структуре. Метод позволяет повысить точность получаемых решений за счёт использования вычислительных блоков ПЛИС увеличенной разрядности, однако требует дополнительных аппаратных затрат в сравнении с методом априорного поиска.

Появление новых методов реализации алгоритмов быстрого преобразования Фурье на РВС позволило разработать методику создания структурно-процедурных алгоритмов и параллельно-конвейерных программ для решения задач цифровой обработки сигналов и изображений, использующих БПФ для данных с фиксированной запятой, которая включает следующие этапы:

1) синтез общей функциональной структуры реализуемого алгоритма решаемой задачи;

2) расстановка базовых функциональных элементов алгоритма;

3) выбор варианта масштабируемой разрядности для функциональных элементов, позволяющего обеспечить заданную точность вычислений:

- управляемое масштабирование результата текущей операции;
- поэтапное увеличение разрядности для представления результата текущей операции;
- комбинированный вариант масштабируемой разрядности;

4) создание унифицированной вычислительной структуры для выбранного варианта масштабируемой разрядности.

Под унифицированной вычислительной структурой понимается структура, топология которой не меняется при организации различных информационных потоков.

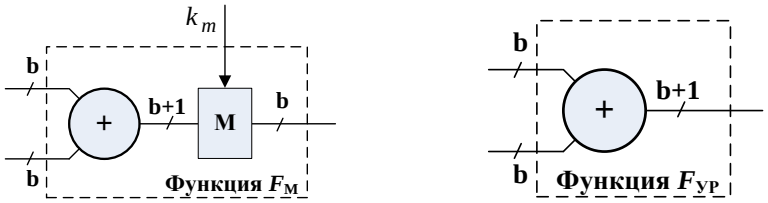
На первом этапе создания параллельно-конвейерного алгоритма выполняется построение общего информационного графа решаемой задачи, представляющего собой множество соединенных между собой вычислительных узлов (функциональных элементов). Узлами информационного графа могут являться как элементарные арифметические операции (сумматоры, умножители и т.п.), так и необходимые вычислительные алгоритмы.

На втором этапе в зависимости от требований, предъявляемых к аппаратным ресурсам вычислительной системы, определяется число ступеней вычислительного конвейера и число конвейеров, необходимых для реализации алгоритма, после чего осуществляется расстановка базовых функциональных элементов.

На третьем этапе в зависимости от требований, предъявляемых к точности получаемых решений, выбирается тип масштабируемой разрядности, который будет применяться для результатов вычислений в конвейерных вычислительных блоках. Сначала с помощью программного моделирования проверяется, обеспечивает ли использование управляемого масштабирования заданную точность вычислений. Если точность вычислений удовлетворяет заданной, то для реализации на РВС выбирается алгоритм с данным типом масштабируемой разрядности, как обеспечивающий более высокую удельную производительность. Если управляемое масштабирование не позволяет обеспечить заданную точность, то рассматривается комбинированный вариант масштабируемой разрядности, использующий как управляемое масштабирование, так и поэтапное увеличение разрядности.

Если использование комбинированного варианта не позволяет обеспечить требуемую точность вычислений, то рассматривается вариант с поэтапным увеличением разрядности. Если и он не удовлетворяет заданной точности, то выбирается реализация алгоритма в формате с плавающей запятой, дающая максимальную точность вычислений.

На рис. 1 в качестве примера приведены две конвейерные функции $F_{\text{М}}$ и $F_{\text{УР}}$, реализующие соответствующий тип масштабируемой разрядности. Функции выполняют операцию суммирования двух b-разрядных чисел. Конвейерная функция $F_{\text{М}}$ (рис. 1,а) осуществляет приведение (b+1)-разрядного результата суммирования к b-разрядному значению с помощью блока управляемого масштабирования М, который в зависимости от подаваемого на него управляющего сигнала k_m выбирает b разрядов числа, либо со старшего (масштабирование с отбрасыванием младшего значащего разряда), либо со следующего за старшим (без масштабирования). Конвейерная функция $F_{\text{УР}}$ (рис. 1,б) осуществляет увеличение разрядности для результата суммирования, сохраняя все (b+1) значащих разряда.



а б

Рис. 1. Конвейерные функции: а – с управляемым масштабированием; б – с увеличением разрядности

На четвёртом этапе для выбранного варианта масштабируемой разрядности разрабатывается унифицированная вычислительная структура, подразумевающая возможность реализации алгоритма или подобных алгоритмов на заданном ограниченном числе ступеней вычислительного конвейера. К подобным алгоритмам можно отнести прямой и обратный БПФ, прямой и обратный двумерный БПФ (для обработки двумерных сигналов и изображений) и т.п. Если структура будет неунифицированной, то для реализации на одной вычислительной структуре подобных алгоритмов, выполняемых друг за другом, потребуется реконфигурирование ПЛИС, что является крайне нежелательным. Ввиду того что реализация алгоритма с масштабируемой разрядностью уже обладает унифицированной вычислительной структурой (разрядности данных на входах и выходах функциональных блоков совпадают), обеспечение унификации необходимо только для реализаций алгоритмов с поэтапным увеличением разрядности и с комбинированным вариантом масштабируемой разрядности. Для этого разрядности данных на выходах последней ступени вычислительного конвейера должны быть приведены к разрядности входов первой ступени конвейера, что достигается применением операции масштабирования. Представленный подход позволяет реализовать на одной и той же вычислительной структуре с заданным ограниченным числом ступеней конвейера расчёт всего алгоритма, а также подобных алгоритмов.

Подобный подход позволяет использовать все преимущества формата с фиксированной запятой для алгоритмов БПФ и полностью исключить ситуации, связанные с переполнением разрядной сетки.

Заключение. Достоинствами предложенного подхода цифровой обработки изображений на РВС с использованием алгоритмов БПФ являются:

- значительное сокращение числа арифметических операций и увеличение удельной и реальной производительности реконфигурируемых вычислительных систем;
- возможность проектирования подстраиваемой вычислительной структуры алгоритма под имеющийся аппаратный ресурс;
- высокая точность получаемых решений для данных с фиксированной запятой.

Следует отметить, что представленные способы оптимизации задач цифровой обработки изображений могут быть применены и для ряда других задач, использующих БПФ, где входные данные представляют собой действительные последовательности.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Gonzalez R.C., Woods R.E.* Digital Image Processing. – 4th ed., Pearson, 2018.
2. *Gonzalez R.C., Woods,R.E., and Eddins S.L.* Digital Image Processing Using MATLAB. – 3rd ed., Gatesmark Publishing, Knoxville, TN, 2009.
3. *Castleman K.R.* Digital Image Processing, Pearson Education, 2007.
4. *Jain A.K.* Fundamentals of Digital Image Processing, Prentice Hall, 1989.
5. Методы компьютерной обработки изображений / под ред. В.А. Сойфера. – 2-е изд. испр. – М.: Физматлит, 2003. – 784 с.
6. *Каляев И.А., Левин И.И., Семерников Е.А., Шмойлов В.И.* Реконфигуриуемые мультиконвейерные вычислительные структуры / под общ. ред. И.А. Каляева. – 2-е изд., перераб. и доп. – Ростов-на-Дону: ЮНЦ РАН, 2009. – 344 с.
7. *Гузик В.Ф., Каляев И.А., Левин И.И.* Реконфигурируемые вычислительные системы: учеб. пособие / под общ. ред. И.А. Каляева. – Таганрог: ЮФУ, 2016. – 472 с.
8. *Воеводин В.В., Воеводин Вл.В.* Параллельные вычисления: учеб. пособие. – СПб.: БХВ-Петербург, 2004. – 608 с.
9. *Зотов В.Ю.* Проектирование встраиваемых микропроцессорных систем на основе ПЛИС фирмы Xilinx. – М.: Горячая линия - Телеком, 2006. – 520 с.
10. *Артамонов Д.С., Путря М.Г.* Метод оптимизации вычислительного процесса на реконфигурируемых вычислительных средах // Информационные технологии и вычислительные системы. – М.: ИСА РАН, 2010. – № 3. – С. 19-26.
11. *Дордопуло А.И., Левин И.И.* Программное обеспечение для решения вычислительно-трудоёмких задач на реконфигурируемых вычислительных системах // Матер. 2-ой Международной научной конференции «Суперкомпьютерные системы и их применение» (SSA'2008). – Минск: ОИПИ НАН Беларуси, 2008. – С. 50-54.
12. *Лайонс Р.* Цифровая обработка сигналов: пер с англ. / под ред. А.А. Бритова. – 2 изд. – М.: Бином-Пресс, 2006. – 656 с.
13. *Сергиенко А.Б.* Цифровая обработка сигналов: учеб. пособие. – 3-е изд. – СПб.: БХВ-Петербург, 2011. – 768 с.
14. *Орлов С.А., Цилькер Б.Я.* Организация ЭВМ и систем: учебник для вузов. – 2-е изд. – СПб.: Питер, 2011. – 688 с.
15. *Солонина А.И., Улахович Д.А., Яковлев Л.А.* Алгоритмы и процессоры цифровой обработки сигналов. – СПб.: БХВ – Петербург, 2002. – 464 с.
16. *Бовкун А.В.* Автоматизированные методы повышения удельной производительности прикладных задач для реконфигурируемых вычислительных систем // Известия ЮФУ. Технические науки. – 2012. – № 4. – С. 204-210.
17. *Фисенко В.Т., Фисенко Т.Ю.* Компьютерная обработка и распознавание изображений: учеб. пособие. – СПб.: СПбГУ ИТМО, 2008. – 192 с.
18. *Грузман И.С. и др.* Цифровая обработка изображений в информационных системах: учебное пособие. – Новосибисрк: Изд-во НГТУ, 2000. – 168 с.
19. *Блейхут Р.* Быстрые алгоритмы цифровой обработки сигналов: пер. с англ. – М.: Мир, 1989. – 448 с.
20. *Айфичер Э.С., Джервис Б.У.* Цифровая обработка сигналов: практический подход: пер. с англ. – 2-е изд. – М.: Вильямс, 2008. – 992 с.
21. *Oppenheim Alan V., Ronald W. Schafer, John R.* Buck Discrete Time Signal Processing. – 2nd ed. – Prentice Hall, 1998.
22. *Чкан А.В., Михайлов Д.В.* Применение масштабируемой разрядности данных при программной реализации быстрого преобразования Фурье на реконфигурируемых вычислительных системах // Вестник компьютерных и информационных технологий. – 2018. – № 1 (163). – С. 44-50.
23. *Ершова О.В., Кириченко Е.В., Семерников Е.А., Чкан, А.В.* Масштабирование данных с фиксированной точкой в процедуре быстрой свертки // Радиотехника. – 2015. – № 4. – С. 66-72.

REFERENCES

1. *Gonzalez R.C., Woods R.E.* Digital Image Processing. 4th ed., Pearson, 2018.
2. *Gonzalez R.C., Woods,R.E., and Eddins S.L.* Digital Image Processing Using MATLAB. 3rd ed., Gatesmark Publishing, Knoxville, TN, 2009.
3. *Castleman K.R.* Digital Image Processing, Pearson Education, 2007.
4. *Jain A.K.* Fundamentals of Digital Image Processing, Prentice Hall, 1989.
5. Metody komp'yuternoy obrabotki izobrazheniy [Methods of computer image processing], ed. by V.A. Soyfera. 2-e izd. Moscow: Fizmatlit, 2003, 784 p.
6. *Kalyaev I.A., Levin I.I., Semernikov E.A., Shmoylov V.I.* Rekonfiguriuemye mul'tikonveyernye vychislitel'nye struktury [Reconfigurable multipipeline computing structures], under the general ed. I.A. Kalyaeva. 2-e ed. Rostov-on-Don: YuNTS RAN, 2009, 344 p.
7. *Guzik V.F., Kalyaev I.A., Levin I.I.* Rekonfiguriruemye vychislitel'nye sistemy: ucheb. posobie [Reconfigurable computing systems: tutorial], under the general. ed. I.A. Kalyaeva. Taganrog: YuFU, 2016, 472 p.
8. *Voevodin V.V., Voevodin Vl.V.* Parallel'nye vychisleniya: ucheb. posobie [Parallel calculations:: tutorial]. St. Petersburg: BKhV-Peterburg, 2004, 608 p.
9. *Zotov V.Yu.* Proektirovanie vstraivaemykh mikroprotsessornykh sistem na osnove PLIS firmy Xilinx [Designing embedded microprocessor systems based on FPGAs of Xilinx]. Moscow: Goryachaya liniya - Telekom, 2006, 520 p.
10. *Artamonov D.S., Putrya M.G.* Metod optimizatsii vychislitel'nogo protsessa na rekonfiguriruemykh vychislitel'nykh sredakh [Method of optimizing the computing process on reconfigurable computing environments], *Informatsionnye tekhnologii i vychislitel'nye sistemy* [Information technologies and computing systems]. Moscow: ISA RAN, 2010, No. 3, pp. 19-26.
11. *Dordopulo A.I., Levin I.I.* Programmnoe obespechenie dlya resheniya vychislitel'no-trudoemkikh zadach na rekonfiguriruemykh vychislitel'nykh sistemakh [Software for solving computationally time-consuming problems on reconfigurable computing systems], *Mater. 2-oy Mezhdunarodnoy nauchnoy konferentsii «Superkomp'yuternye sistemy i ikh primenenie» (SSA'2008)* [Materials of the 2nd International Scientific Conference "Supercomputer Systems and their Application" (SSA '2008)]. Minsk: OIPI NAN Belarusi, 2008, pp. 50-54.
12. *Layons R.* TSifrovaya obrabotka signalov [Digital Signal Processing]: transl. from engl., by ed. A.A. Britova. 2nd ed. Moscow: Binom-Press, 2006, 656 p.
13. *Sergienko A.B.* Tsifrovaya obrabotka signalov: ucheb. posobie [Digital signal processing: tutorial]. 3rd ed. St. Petersburg: BKhV-Peterburg, 2011, 768 p.
14. *Orlov S.A., Tsil'ker B.Ya.* Organizatsiya EVM i sistem: uchebnik dlya vuzov [Organization of computers and systems: tutorial for universities]. 2nd ed. St. Petersburg: Piter, 2011, 688 p.
15. *Solonina A.I., Ulakhovich D.A., Yakovlev L.A.* Algoritmy i protsessory tsifrovoy obrabotki signalov [Algorithms and processors of digital signal processing]. St. Petersburg: BKhV – Peterburg, 2002, 464 p.
16. *Bovkun A.V.* Avtomatizirovannye metody povysheniya udel'noy proizvoditel'nosti prikladnykh zadach dlya rekonfiguriruemykh vychislitel'nykh sistem [Automated methods for increasing the specific performance of applications for reconfigurable computing systems], *Izvestiya YuFU. Tekhnicheskie nauki* [Izvestiya SFedU. Engineering Sciences], 2012, No. 4, pp. 204-210.
17. *Fisenko V.T., Fisenko T.Yu.* Komp'yuternaya obrabotka i raspoznavanie izobrazheniy: ucheb. posobie [Computer processing and recognition of images: tutorial]. St. Petersburg: SPbGU ITMO, 2008, 192 p.
18. *Gruzman I.S. i dr.* TSifrovaya obrabotka izobrazheniy v informatsionnykh sistemakh: ucheb. posobie [Digital image processing in information systems: tutorial]. Novosibisrk: Izd-vo NGTU, 2000, 168 p.
19. *Bleykhut R.* Bystrye algoritmy tsifrovoy obrabotki signalov [Fast digital signal processing algorithms]: transl. from engl. Moscow: Mir, 1989, 448 p.
20. *Ayficher E.S., Dzhervis B.U.* TSifrovaya obrabotka signalov: prakticheskiy podkhod [Digital signal processing: a practical approach]: transl. from engl. 2nd ed. Moscow: Vil'yams, 2008, 992 p.
21. *Oppenheim Alan V., Ronald W. Schafer, John R.* Buck Discrete Time Signal Processing. 2nd ed. Prentice Hall, 1998.

22. *Chkan A.V., Mikhaylov D.V.* Primenenie masshtabiruemoy razryadnosti dannykh pri programmnoy realizatsii bystrogo preobrazovaniya Fur'e na rekonfiguriruemykh vychislitel'nykh sistemakh [Application of scalable data bitness in software implementation of fast Fourier transformation on reconfigurable computing systems], *Vestnik komp'yuternykh i informatsionnykh tekhnologiy* [Herald of Computer and Information Technologies], 2018, No. 1 (163), pp. 44-50.
23. *Ershova O.V., Kirichenko E.V., Semernikov E.A., Chkan, A.V.* Masshtabirovanie dannykh s fiksirovannoy tochkoy v protsedure bystroy svertki [Scaling of fixed point data in the fast convolution procedure], *Radiotekhnika* [Radio engineering], 2015, No. 4, pp. 66-72.

Статью рекомендовал к опубликованию д.т.н. Э.В. Мельник.

Чкан Андрей Викторович – ООО "НИЦ супер-ЭВМ и нейрокомпьютеров"; e-mail: chkan_andrey@mail.ru; 347900, г. Таганрог, пер. Итальянский, 106; тел.: 88634368177; к.т.н.; научный сотрудник отдела ЦОС.

Chkan Andrey Victorovich – "Scientific research center of supercomputers and neurocomputers" CoLtd; e-mail: chkan_andrey@mail.ru; 106, Italyansky lane, Taganrog, 347900, Russia; phone: +78634368177; cand. of eng. sc.; research associate of department of Digital Signal Processing.

УДК 004.421 DOI 10.18522/2311-3103-2020-7-163-171

Н.И. Витиска, Н.А. Гуляев, В.В. Селянкин

ОПТИМИЗАЦИЯ ПРОЕКТИРОВАНИЯ МНОГОКАНАЛЬНОЙ СИСТЕМЫ С ИСПОЛЬЗОВАНИЕМ ЛОГИЧЕСКОГО СИНТЕЗА ДЛЯ ПОВЫШЕНИЯ КАЧЕСТВА ОБЪЕМНОЙ ВИЗУАЛИЗАЦИИ

Рассматривается задача оптимизации проектирования многоканальных систем, используемых для прямой объемной визуализациис целью повышения качества её результата. Объемная визуализация широко используется в современных системах компьютерной визуализации, моделирования, симуляции, технического зрения, при этом отличается необходимостью обработки больших объемов данных для возможности получения высокого качества результата. Задача оптимизации проектирования многоканальных систем для объемной визуализации рассматривается с точки зрения достижения необходимого качества синтезируемого изображения при минимальных затратах. В работе предлагается метод логического синтеза таких систем, позволяющего получить оптимальные соотношения качества-затрат в зависимости от требуемых параметров постановки задачи. Предлагаемый метод позволяет достигать качества, близкого к результатам полного перебора, но требующего значительно меньший объем вычислений. Для каждого канала системы определяется набор переменных, оптимизация которых обеспечит качество результата визуализации. На основе параметров оптимизации строится переключательная функция с помощью диаграммы Вейча. Данный подход осуществляется программным путём в каждом канале распределённой системы в реальном масштабе времени, что задаёт общую схему такой методики. В процессе выполнения работы проводились экспериментальные исследования зависимости точности решения и объема вычислений для прямой объемной визуализации в каждом канале распределённой системы. Разработана методика оптимального синтеза изображений при условии выравнивания качества воспроизведения в небольшой группе каналов распределённой системы.

Оптимизация; многоканальные системы; логический синтез; объемная визуализация качество визуализации; трассировка лучей.

N.I. Vitiska, N.A. Gulyaev, V.V. Selyankin

MULTICHANNEL SYSTEM DESIGN OPTIMIZATION USING LOGICAL SYNTHESIS FOR QUALITY IMPROVEMENT OF VOLUME VISUALIZATION

The paper reviewes a problem of optimization and quality improvement of development and design of multi-channel systems which perform direct volume visualization. Volume visualization is widely used in modern computer graphics and visualization systems. Volume visualization is well-know for its requrements – it demands large amounts of data to be processed to produce a high quality result. The optimization problem is considered as a quality-cost dependence, where the target is to achieve is the required quality level at minimal cost. The paper proposes a method for logical synthesis of such systems, which allows to obtain optimal quality-cost ratios depending on the required parameters. The proposed method allows to achieve a quality level, that is close to results of a full-search solutions, but it requires a significantly smaller amount of calculations. For each channel of the system, a set of variables is defined, the optimization of which will ensure the quality of the resulting images. Based on the optimization parameters, a switching function is constructed using a Veitch diagram. This approach is implemented programmatically in each channel of the distributed system in real time, what sets the general scheme of the method. In described study, experimental research of relatationship between the accuracy of the solution and the amount of calculations of direct volume visualization in each channel of a distributed system was performed. A method for optimal image synthesis based equalizing the playback quality in a small group of channels in a distributed system was developed.

Optimization; multi-channel systems; logical synthesis; volume visualization; visualization quality; ray tracing.

Введение. Системы реального времени передачи и обработки данных широко используются в решении задач визуализации, моделирования, симуляции, технического зрения. Примерами таких задач могут быть задачи синтеза и анализа изображений, выделение и восстановление объектов и объёмных феноменов в медицине и дефектоскопии, визуализация в системах виртуальной реальности, обучающих системах и виртуальных тренажёрах. Отличительными особенностями таких систем являются наличие большого объема данных и требования максимально возможного качества и скорости их обработки.

Высокая стоимость таких систем передачи и обработки данных требует разработки программно-аппаратных решений, позволяющих передавать по одной линии связи одновременно большого числа сообщений. С этой целью используют многоканальные системы, которые в своем развитии идут от аналоговых технологий к аналогово-цифровой реализации с постепенным увеличением цифровых решений. Получение заданных характеристик каналов, обеспечивающих высокое качество передачи информации, при проектировании цифровых и аналоговых систем передачи требует принятия таких программно-аппаратных решений, которые дают допустимые результаты по затратам и качеству.

Поскольку решение большинства современных задач в технических системах тесно связано с процедурой визуализации, которая может выступать как часть какой-то разработки, либо являться самостоятельной задачей, решаемой системой, то от эффективности визуализации, обеспечивающей необходимую точность и объем временных затрат, либо вычислительных ресурсов, зависит функционирование всей системы в целом. Стоит отметить, что современные программные системы нередко характеризуются несоответствием используемых инструментов визуализации требованиям задач. Например, актуальные в настоящее время задачи в областях интерактивной графики, виртуальной и дополненной реальности, системах моделирования, проектирования и дизайна требуют всё большего качества и реалистичности синтезируемых изображений.

Ряд задач визуализации в различных областях техники в настоящее время решается методами прямой объёмной визуализации. Ее применение может обеспечивать высокие показателей качества и степени реалистичности по сравнению с другими технологиями визуализации. Однако ее использование связано с необходимостью значительных стоимостных затрат и вычислительных ресурсов. В связи с этим прямая объемная визуализация требует разработки методов оптимизации [1–3]. Использование гибридных методов прямой объёмной визуализации и компромиссных оптимизационных решений является перспективным оптимизационным решением благодаря большой гибкости. При этом исследования в этом направлении имеют своей целью поиск вариантов оптимального соотношения точности и времени визуализации. Практическое применение прямой объёмной визуализации в настоящее время ограничено по сравнению с другими парадигмами визуализации, однако, в области применения данной парадигмы наиболее широкое распространение получили методы рендеринга, основанные на трассировке лучей. Этот метод является одним из наиболее высокоточных методов рендеринга, однако он также является и одним из наиболее вычислительно сложных методов рендеринга. Поэтому исследования, ориентированные на повышение скорости метода трассировки лучей в рамках прямой объёмной визуализации, в настоящее время являются актуальными.

Важным вопросом в контексте повышения производительности объёмной визуализации является аппаратная и системная составляющая, т.е. возможность повышения производительности возможно и за счёт подбора наиболее подходящих для данной задачи аппаратных комплексов и операционных систем [4]. Действительно, реализация прямой объёмной визуализации аппаратными средствами общего назначения (CPU) может быть недостаточно производительной, специализированные вычислительные модули (например, GPU) могут не являться достаточно гибкими для решения всех задач в рамках объёмной визуализации [5]. С другой стороны, применение многоканальных распределённых систем может быть более подходящим вариантом, так как существует возможность варьировать архитектуру таких систем, в том числе – для приведения в соответствие с потребностями объёмной визуализации [6].Обычно используются методы оптимизации, ориентированные на решение достаточно ситуативных задач – задач сокращения количества вычислений за счёт исключения из обработки фрагментов объёмного изображения, оказывающих малое влияние на результат [7].Представляют интерес и значительно более стратегически ориентированные подходы, которые ставят целью решение более фундаментальных задач – задач проектирования систем, имеющих возможность решения конкретной задачи в конкретных условиях с учётом практических нюансов – стоимости оборудования, повышения отказоустойчивости и так далее [8–10]. Одной из таких задач является задача оптимизации проектирования многоканальных систем, используемых для прямой объемной визуализации.

Постановка задачи. Обычно для каждого канала определяется набор переменных, оптимизация которых обеспечит качество получаемых изображений при определённой динамике их формирования. В рассматриваемой постановке задачи оптимизации число исследуемых параметров k, удовлетворяющих качеству получаемого изображения, предполагается не более двух, т.е. $n = 2^k$, где$k = [1; 2]$, а число оптимизируемых параметров n не более4-х. Поэтому на каждом уровне оптимизации строится двухпараметрическая целевая функция. В качестве выбора одного или четырёх параметров оптимизации может быть построена переключательная функция с помощью диаграммы Вейча. Например, из переменных $x_1 \dots x_4$ строится конституэнта единицы, по которой программа оптимизирует параметры x без инверсий. В свою очередь параметры с инверсиями должны оптимизироваться на втором уровне.

Данный подход осуществляется программным путём в каждом канале распределённой системы в реальном масштабе времени, что задаёт общую схему методики. При проектировании многоканальных распределённых систем следует решать проблемы с качеством синтезируемых изображений одновременно во всех каналах. Поэтому для выбора наилучшего решения рассмотрим следующий пример, который относился бы к системе, например с16-ю каналами. Будем предполагать, что система характеризуется по каждому каналу четырьмя параметрами X_1, X_2, X_3, X_4 и соответствующей стоимостью обеспечения качества *CT*, которое будет достигнуто в данном канале для целевой функции оптимизации в виде переключательной функции из диаграммы Вейча (см. табл. 1). В ней каждая клетка соответствует конституэнте единицы, совпадающей с номером канала, т.е., например, $F = X_1X_2X_3X_4$ для 15-того канала. Поэтому для каждого канала сформулируем две постановки задачи (ПЗ) оптимизации:

Первая постановка:
$$F_1 = \text{СТ} \to min;$$
$$\text{ВК}_i \geq \text{ВК}_\text{з}$$

Вторая постановка:
$$F_2 = \text{ВК}_{\text{общ}} \to max;$$
$$\text{СТ} \leq \text{СТ}_\text{з}$$

Таким образом, при первой постановке задачи производится отбор таких вариантов реализации качества в каналах, чтобы система являлась самой дешёвой при условии, что качество в каждом канале будет по каждой переменной не ниже заданного. В свою очередь, во второй постановке необходимо достигать максимума по каждой переменной X при условии, что стоимость будет не выше заданной.

Известны два метода решения подобного рода задач [11–20]:

а) перебор всевозможных вариантов;

б) решение задачи оптимизации.

Таблица 1

Диаграмма Вейча для переключательной функции

X_1 \ X_2			
12	13	9	8
14	15	11	10
6	7	3	2
4	5	1	0

X_3

X_4

Если предположить, что общее качество работы канала есть произведение качеств ВК_i:

$$\text{ВК}_{\text{общ.}} = \text{вк}_1\text{вк}_2\text{вк}_3; \tag{1.1}$$
$$\text{СТ}_{\text{общ.}} = \text{ст}_1 + \text{ст}_2 + \text{ст}_3,$$

где $вк_1$,$вк_2$, $вк_3$ – вероятность подключения достаточного количества программно-аппаратных средств с целью получения заданного качества переменных x_1, x_2, x_3,x_4, $ст_1$, $ст_2$, $ст_3$ – стоимость программно-аппаратных средств.

Таблица 2

Варианты показателей качества ПАС

Варианты	Показатели качества программно-аппаратных средств		
	М1	М2	М3
1	$вк_1^1 = 0{,}95$ $ст_1^1 = 15000$	$вк_1^2 = 0{,}99$ $ст_1^2 = 30000$	$вк_1^3 = 0{,}95$ $ст_1^3 = 15000$
2	$вк_2^1 = 0{,}99$ $ст_2^1 = 30000$	——— ———	$вк_2^3 = 0{,}99$ $ст_2^3 = 25000$
3	$вк_3^1 = 0{,}9$ $ст_3^1 = 20000$	———	———

Будем считать, что в многоканальной системе возможно подключение трёх программно-аппаратных средств (ПАС), каждый из которых реализуется несколькими способами, как показано в табл. 2. В ней нижний индекс параметра соответствует варианту реализации, а верхний – виду программно-аппаратных средств. Требуется выбрать наилучший вариант реализации системы с учётом показателей качества по каждому каналу и соответствующих параметровx_1, x_2,x_4.

Решение методом полного перебора. Рассмотрим возможность решения сформулированной выше задачи методом полного перебора вариантов многоканальной системы. Для этого в табл. 3 сведём возможные варианты системы и её характеристики по *вк* и стоимости *ст*, рассчитанные согласно формулам (1.1). Общее число вариантов реализации многоканальной системы составляет: $NB = 3 \times 1 \times 2 = 6$, где $3, 1, 2$ – число способов возможной реализации соответствующих (с первого по третий) программно-аппаратных средств многоканальной системы (см. табл. 2). В таблицах 4,5 сведены лучшие варианты реализации многоканальной системы по критериям качества объёмных изображений и стоимости программно-аппаратных средств, обеспечивающих это качество.

Таблица 3

Возможные варианты системы и её характеристики

Вариант системы	Способ реализации ПАСв системе			Показатели качества изображений в каналах		Оптимальный вариант
	1	2	3	*ст*	*вк*	
1	1	1	1	60000	0,893	2-я ПЗ
2	2	1	1	75000	0,931	———
3	3	1	1	65000	0,846	———
4	1	1	2	70000	0,931	1-я ПЗ
5	2	1	2	85000	0,970	———
6	3	1	2	75000	0,882	———

Лучшие варианты системы по качеству результата приведены в табл. 4.

Таблица 4

Лучшие варианты системы по качеству

Варианты системы	Показатели	
	СТ	*ВК*
2	75000	0,931
4	70000	0,931
5	85000	0,970

Лучшие варианты системы по критерию стоимости приведены в табл. 5.

Таблица 5

Лучшие варианты системы по стоимости

Варианты системы	Показатели	
	СТ	*ВК*
1	60000	0,893
3	65000	0,846

Известно [11], что в общем случае, при выборе оптимальной структуры системы, состоящей из L компонентов (в нашем случае ПАС), где каждый l-й ПАС может быть реализован в m_l-х вариантах, общее число вариантов реализации NB составляет$NB = m_1.m_{2\ldots}m_L$. Тогда, если все ПАС реализуются одинаковым числом способов, т.е. $m_1 = m_2 = \cdots = m_L$, то $NB = m^L$.

Число возможных вариантов реализации многоканальной системы при определённых значениях L ($L = 5, 10$) и m ($m = 2, 6$) (см. табл. 2) показывает, что для реальной системы из 10 ПАС, где каждый ПАС может быть реализован одним из 6 возможных вариантов, имеется 60 млн. вариантов. Если принять, что расчёт одного варианта реализации требует 0,1 мин., то общее время, необходимое для расчёта всех вариантов, составит около 5 лет.

Решение задачи оптимизации. Выбор оптимальной структуры системы выполняется путём решения задачи оптимизации. Здесь считается оптимальной структура системы, использующая дешёвые ПАС, на базе которых решается задача оптимизации. Для решения задачи оптимизации составим математическую модель с использованием булевых переменных $y_m^l \epsilon \{0, 1\}$, где l – номер варианта; m– номер варианта реализации ПАС, определяемой в результате решения задачи оптимизации.

Причём принимаем то, что$y_m^l = 1$, если длял – го ПАС принят -й вариант его решения, $y_m^l = 0$ – в противном случае. Когда используется одновариантная реализация ПАС системы, то приходим к следующей математической модели:

$$y_1^1 \veebar y_2^1 \veebar y_3^1 = 1;\ y_1^2 = 1;\ y_1^3 \veebar y_2^3 = 1, \qquad (1.2)$$

где $\veebar$ – знак операции дизъюнкции, а верхний индекс указывает на номер ПАС системы.

Стоимость каждого ПАС в зависимости от принятого варианта реализации составляет (значения cm_m^i взяты из табл. 2):

$$\begin{aligned} \text{ст}_1 &= 15000y_1^1 + 30000y_2^1 + 20000y_3^1; \\ \text{ст}_2 &= 30000y_1^2; \\ \text{ст}_3 &= 15000y_1^3 + 25000y_2^3; \end{aligned} \qquad (1.3)$$

Аналогично для вероятности подключения достаточного количества ПАС запишем такую математическую модель:

$$\begin{aligned} \text{вк}_1 &= 0{,}95y_1^1 + 0{,}99y_2^1 + 0{,}90y_3^1; \\ \text{вк}_2 &= 0{,}99y_1^2; \\ \text{вк}_3 &= 0{,}90y_1^3 + 0{,}99y_2^3; \end{aligned} \qquad (1.4)$$

Так как параметры систем зависят от параметров ПАС согласно (1.1) , то эти ограничения условно обозначим через (1.5), а все в совокупности ограничения можно объединить подобщим номером (1.6), записав его как

(1.6)= (1.2)+(1.3)+(1.4) +(1.5).

Задачу оптимизации можно сформулировать следующим образом:

Первая постановка:	Вторая постановка:
$F_1 = \text{СТ} \rightarrow min;$	$F_2 = \text{ВК} \rightarrow max;$
Ограничения (1.6);	Ограничения (1.6);
$\text{ВК} \geq 0{,}93$	$\text{СТ} \geq 65000$

Полученные системы представляют собой задачу нелинейного программирования, так как зависимости для вероятностей являются нелинейными. Поэтому её можно решать как нелинейную задачу, которая реализуется на ЭВМ значительно быстрее по сравнению с вариантом полного перебора.

Выводы. Объектом исследования является с одной стороны автоматизация прямой объемной визуализации в многоканальных распределенных системах реального времени, а с другой стороны оптимизация их проектирования с учётом стоимостных затрат на программно-аппаратные средства, обеспечивающих необходимый уровень качества результата. Следовательно, полученные результаты в работе позволяют заменить полный перебор при проектировании многоканальных систем на решение задачи нелинейного программирования, добиваясь тем самым минимизации временных затрат и ресурсов ЭВМ.

В процессе выполнения работы проводились экспериментальные исследования зависимости точности решения и объема вычислений для прямой объемной визуализации в каждом канале распределённой системы. Разработана методика оптимального синтеза изображений при условии выравнивания качества воспроизведения в небольшой группе каналов распределённой системы.

Благодарность. Работа выполнена при финансовой поддержке РФФИ, грант № 18- 07 -00733(а).

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Витиска Н.И., Гуляев Н.А., Данилов И.Г.* О проблемах и задачах обработки и организации данных при объёмной визуализации на распределённых системах // Известия ЮФУ. Технические науки. – 2019. – № 4. – С. 153-163.
2. *Витиска Н.И. и др.* Оптимизация прямой объёмной визуализации с программируемым управлением качества: монография. – М.: Изд-во «Перо», 2020. – 82 с.
3. *Vitiska N., Selyankin V., Gulyaev N.* An Approach to Optimization of Ray-Tracing In Volume Visualization Based on Properties of Volume Elements // Proceedings of the Third International Scientific Conference «Intelligent Information Technologies for Industry» (IITI'18). – Springer International Publishing, 2019. – Vol. 1. – P. 148-158.
4. *Strassburger S., Schulze T., Fujimoto R.* Future trends in distributed simulation and distributed virtual environments: results of a peer study // Proc. 40th Conferenceon Winter Simulation. – 2008. – P. 777-785.
5. *Морозов Б.Б. и др.* Особенности построения систем визуализации на базе распределенной мультимедийной среды // Вестник НГУ. Серия: Физика. – 2013. – № 4 (8). – С. 118-124.
6. *Морозов Б.Б. и др.* Построение распределённой мультимедийной виртуальной среды с многоканальной визуализацией медиаданных на графических акселераторах // Матер. 25-й Международной конференции по компьютерной графике и зрению ГрафиКон – 2015. – 2013. – С. 166-170.
7. *Витиска Н.И., Гуляев Н.А.* Исследование и разработка метода сокращения вычислений при трассировке лучей в объёмной визуализации // Информатизация и связь. – 2018. – №. 6. – С. 44-52.
8. *Слядников Е.Е.* Моделирование распределенных информационно-телекоммуникационных систем с пакетной передачей данных // Известия Томского политехнического университета. – 2008. – № 5. – С. 55-60.
9. *Калиниченко С.В., Хомоненко А.Д.* Модель оценки оперативности функционирования распределённых автоматизированных систем при интеграции данных // Бюллетень результатов научных исследований: электронный научный журнал. – 2012. – № 5 (4). – С. 47-57.

10. *Барский А.Б., Мельник Д.И., Пирожник В.В.* Модель управления качеством вычислительных средств распределённой многоканальной системы массового обслуживания // Информационные технологии. – 2019. – № 6 (25). – С. 358-367.
11. *Шелобаев С.И.* Математические методы и модели в экономике, финансах, бизнесе: учеб. пособие для вузов. – М.: ЮНИТИ-ДАНА, 2001. – 367 с.
12. *Арсеньев Ю.И., Шелобаев С.И.* Модели и методы оптимизации ресурсов субъектов рынка. – М.: Высшая школа , 1998.
13. *Арсеньев Ю.И., Шелобаев С.И.* Анализ, синтез и оптимизация социо-техно-экономических систем: надёжность, безопасность, эффективность, качество. – М.: Высшая школа, 1998.
14. *Meyhak H.* Simultane Gesamtplanung im mehrstufigen Mehrproduktunternehmen. – Mannheim, 1968.
15. *Zwehl W. von*. Programmierung Ganzzahlige // Handwörterbuch der Wissenschaften. – Stuttgart u.a, 1988.
16. *Таха Х.* Введение в исследование операций. – М.: Мир, 1985. – Т. 1, Т. 2.
17. *Эддоус М., Стэнсфилд Р.* Методы принятия решений. – М.: Аудит, ЮНИТИ, 1997.
18. *Бешелев С.Д., Гурвич Ф.Г.* Математико-статистические методы экспертных оценок. – М.: Статистика, 1980.
19. *Бункин В.А., Курицкий Б.Я., Сокуренко Ю.А.* Справочник по оптимизационным задачам в АСУ. – Л.: Машиностроение, 1984.
20. *Арсеньев Ю.И., Минаев В.С.* Управление рисками. – М.: Высшая школа, 1997.

REFERENCES

1. *Vitiska N.I., Gulyaev N.A., Danilov I.G.* O problemakh i zadachakh obrabotki i organizatsii dannykh pri ob"emnoy vizualizatsii na raspredelennykh sistemakh [On problems and tasks of data processing and organization in volumetric visualization on distributed systems], *Izvestiya YuFU. Tekhnicheskie nauki* [.Izvestiya SFedU. Engineering Sciences], 2019, No. 4, pp. 153-163.
2. *Vitiska N.I. i dr.* Optimizatsiya pryamoy ob"emnoy vizualizatsii s programmiruemym upravleniem kachestva: monografiya [Optimization of direct volumetric visualization with programmable quality control: monograph]. Moscow: Izd-vo «Pero», 2020, 82 p.
3. *Vitiska N., Selyankin V., Gulyaev N.* An Approach to Optimization of Ray-Tracing in Volume Visualization Based on Properties of Volume Elements, *Proceedings of the Third International Scientific Conference «Intelligent Information Technologies for Industry» (IITI'18)*. Springer International Publishing, 2019, Vol. 1, pp. 148-158.
4. *Strassburger S., Schulze T., Fujimoto R.* Future trends in distributed simulation and distributed virtual environments: results of a peer study, *Proc. 40th Conferenceon Winter Simulation*, 2008, pp. 777-785.
5. *Morozov B.B. i dr.* Osobennosti postroeniya sistem vizualizatsii na baze raspredelennoy mul'timediynoy sredy [Features of building visualization systems based on a distributed multimedia environment], *Vestnik NGU. Seriya: Fizika* [Bulletin of the NSU. Series: Physics], 2013, No. 4 (8), pp. 118-124.
6. *Morozov B.B. i dr.* Postroenie raspredelennoy mul'timediynoy virtual'noy sredy c mnogokanal'noy vizualizatsiey mediadannykh na graficheskikh akseleratorakh [Building a distributed multimedia virtual environment with multichannel visualization of media data on graphics accelerators], *Mater. 25-y Mezhdunarodnoy konferentsii po komp'yuternoy grafike i zreniyu GrafiKon – 2015* [Proceedings of the 25th International Conference on Computer Graphics and Vision GraphiCon-2015], 2013, pp. 166-170.
7. *Vitiska N.I., Gulyaev N.A.* Issledovanie i razrabotka metoda sokrashcheniya vychisleniy pri trassirovke luchey v ob"emnoy vizualizatsii [Research and development of a method for reducing calculations in ray tracing in volumetric visualization], *Informatizatsiya i svyaz'* [Informatization and communication], 2018, No. 6, pp. 44-52.
8. *Slyadnikov E.E.* Modelirovanie raspredelennykh informatsionno-telekommunikatsionnykh sistem s paketnoy peredachey dannykh [Modeling of distributed information and telecommunications systems with packet data transmission], *Izvestiya Tomskogo politekhnicheskogo universiteta* [Proceedings of the Tomsk Polytechnic University], 2008, No. 5, pp. 55-60.

9. *Kalinichenko S.V., Khomonenko A.D.* Model' otsenki operativnosti funktsionirovaniya raspredelennykh avtomatizirovannykh sistem pri integratsii dannykh [The model of estimation of efficiency of functioning of the distributed automated systems when integrating data], *Byulleten' rezul'tatov nauchnykh issledovaniy: elektronnyy nauchnyy zhurnal* [Bulletin of the results of scientific research: electronic scientific journal], 2012, No. 5 (4), pp. 47-57.
10. *Barskiy A.B., Mel'nik D.I., Pirozhnik V.V.* Model' upravleniya kachestvom vychislitel'nykh sredstv raspredelennoy mnogokanal'noy sistemy massovogo obsluzhivaniya [The model of quality management of computational means of a distributed multichannel queuing system], *Informatsionnye tekhnologii* [Information technologies], 2019, No. 6 (25), pp. 358-367.
11. *Shelobaev S.I.* Matematicheskie metody i modeli v ekonomike, finansakh, biznese: ucheb. posobie dlya vuzov [Mathematical methods and models in economics, finance, and business: a textbook for universities]. Moscow: YuNITI-DANA, 2001, 367 p.
12. *Arsen'ev Yu.I., Shelobaev S.I.* Modeli i metody optimizatsii resursov sub''ektov rynka [Models and methods of optimizing the resources of market subjects]. Moscow: Vysshaya shkola, 1998.
13. *Arsen'ev Yu.I., SHelobaev S.I.* Analiz, sintez i optimizatsiya sotsio-tekhno-ekonomicheskikh sistem: nadezhnost', bezopasnost', effektivnost', kachestvo [Analysis, synthesis and optimization of socio-techno-economic systems: reliability, safety, efficiency, quality]. Moscow: Vysshaya shkola, 1998.
14. *Meyhak H.* Simultane Gesamtplanung im mehrstufigen Mehrproduktunternehmen. Mannheim, 1968.
15. *Zwehl W. von*. Programmierung Ganzzahlige, *Handwörterbuch der Wissenschaften*. Stuttgart u.a, 1988.
16. *Takha Kh.* Vvedenie v issledovanie operatsiy [Introduction to the study of operations]. Moscow: Mir, 1985, Vol. 1, Vol. 2.
17. *Eddous M., Stensfild R.* Metody prinyatiya resheniy [Methods of decision-making]. Moscow: Audit, YuNITI, 1997.
18. *Beshelev S.D., Gurvich F.G.* Matematiko-statisticheskie metody ekspertnykh otsenok [Mathematical and statistical methods of expert assessments]. Moscow: Statistika, 1980.
19. *Bunkin V.A., Kuritskiy B.Ya., Sokurenko Yu.A.* Spravochnik po optimizatsionnym zadacham v ASU [Handbook of optimization problems in automated control systems]. Leningrad: Mashinostroenie, 1984.
20. *Arsen'ev Yu.I., Minaev V.S.* Upravlenie riskami [Risk management]. Moscow: Vysshaya shkola, 1997.

Статью рекомендовал к опубликованию д.т.н., профессор Л.К. Бабенко.

Витиска Николай Иванович – Общество с ограниченной ответственностью «Научно-исследовательский институт многопроцессорных вычислительных и управляющих систем»; e-mail: vit614294@rambler.ru; 347905, г. Таганрог, ул. Социалистическая, 150-г.; тел.: 88634614294; д.т.н.; в.н.с.

Гуляев Никита Андреевич – Южный федеральный университет; e-mail: m.yo.da@yandex.ru; 347928, г. Таганрог, пер. Некрасовский, 44; тел.: 88634371673; кафедра МОПЭВМ; ассистент.

Селянкин Владимир Васильевич – e-mail: selyankin@sfedu.ru; кафедра МОП ЭВМ; к.т.н.; доцент.

Vitiska Nikolaj Ivanovich – Scientific Research Institute of Multiprocessor Computer and Control Systems, Co Ltd.; e-mail: vit614294@rambler.ru; 150-g, Socialist street, Taganrog, 347905, Russia; phone: +78634614294; dr. of eng. sc.; leading researcher.

Gulyaev Nikita Andreevich – Southern Federal University; e-mail: m.yo.da@yandex.ru; 44, Nekrasovskiy, Taganrog, 347928, Russia; phone: +788634371673; the department of software engineering; cand. of eng. sc.; assistant professor.

Selyankin Wladimir Wasilevich – e-mail: selyankin@sfedu.ru; the department of software engineering; cand. of eng. sc.; associate professor.

Раздел V. Интеграция параллельных и гибридных распределенных вычислений

УДК 519.688 DOI 10.18522/2311-3103-2020-7-172-180

К.В. Герценбергер, А.И. Чеботов, И.Н. Александров, И.А. Филозова, Е.И. Александров

ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ СОСТОЯНИЙ ДЛЯ ОНЛАЙН И ОФЛАЙН ОБРАБОТКИ ДАННЫХ ЭКСПЕРИМЕНТАЛЬНЫХ УСТАНОВОК КОМПЛЕКСА NICA

Хранение, обработка и анализ экспериментальных и смоделированных данных являются неотъемлемой частью всех современных экспериментов физики высоких энергий. Эти задачи имеют важное значение в экспериментах комплекса NICA, строящегося в Объединенном институте ядерных исследований (ОИЯИ), из-за большой частоты взаимодействия и множественности частиц в событиях столкновения ионов, в связи с этим особенно актуальна автоматизация рассматриваемых процессов для комплекса NICA. Для решения поставленной задачи современные физические эксперименты используют информационные системы различной направленности, которые позволяют управлять потоками данных и обслуживать большое количество одновременных запросов на требуемую информацию от различных систем эксперимента и их пользователей. В статье описывается проектирование новой информационной системы на основе базы данных состояний, а также сопутствующие информационные сервисы для автоматизации хранения и обработки данных и информации об экспериментах проекта NICA. Разрабатываемая база данных состояний предназначена для хранения, поиска и использования различных параметров и информации о режимах работы систем эксперимента. База данных, реализуемая при помощи СУБД (системы управления базами данных) PostgreSQL, будет отвечать за предоставление хранимой информации для обработки данных событий и их физического анализа, а также за организацию прозрачного единого доступа и управление данными на протяжении всего жизненного цикла проводимых научных исследований. В статье показаны схема и цели создаваемой базы данных состояний, представлены её атрибуты, а также выделены ключевые аспекты разработки. Показано место базы данных состояний в архитектуре обработки потока данных эксперимента. Также в статье описана интеграция данной информационной системы с используемым программным обеспечением экспериментов. Начата разработка интерфейсов базы данных состояний для использования хранимых параметров и информации об эксперименте в задачах моделирования событий, обработки "сырых" данных, реконструкции и физического анализа.

Комплекс NICA; онлайн и офлайн обработка данных экспериментальных установок; информационные сервисы; информационная система; база данных состояний.

K.V. Gertsenberger, A.I. Chebotov, I.N. Alexandrov, I.A. Filozova, E.I. Alexandrov

DESIGN OF THE CONDITION DATABASE FOR ONLINE AND OFFLINE DATA PROCESSING IN EXPERIMENTAL SETUPS OF THE NICA COMPLEX

Storing, processing and analyzing of experimental and simulated data are an integral part of all modern high-energy physics experiments. These tasks are of particular importance in the experiments of the NICA project at the Joint Institute for Nuclear Research (JINR) due to the high interaction rate and particle multiplicity of ion collision events, therefore the task of automating the consid-

ered processes for the NICA complex has particular relevance. To solve the task, modern physics experiments use various information systems, which control experiment data flows and simultaneously service a large number of requests from various systems and collaboration members. The article describes the design of a new information system based on the Condition Database as well as related information services to automate storing and processing of data and information on the experiments. The Condition Database is aimed at storing, searching and using various parameters and operation modes of experiment systems. The system being implemented on the PostgreSQL DBMS will provide the information for event data processing and physics analysis and organize a transparent, unified access and data management throughout the life cycle of the scientific research. The article shows the scheme and purposes of the Condition Database and its attributes, key aspects of the design are highlighted. A place of the Condition Database in data processing flow is illustrated. The integration of the information system with experiment software systems is also presented. The development of the Condition Database interfaces has been started to use the stored information in event simulation, raw data processing, reconstruction and physics analysis tasks.

NICA complex; online and offline data processing; information system; information services; condition database.

Введение. Согласно реализуемой программе Объединенного института ядерных исследований (ОИЯИ, г. Дубна) по изучению столкновений тяжелых ионов в ближайшие годы будет завершено строительство ускорительного комплекса NICA [1] (Nuclotron-based Ion Collider Facility, коллайдерная установка для столкновения ионов на базе Нуклотрона) для столкновения частиц в диапазоне атомных масс A = 1–197 при энергиях в системе центра масс до 11 ГэВ для ионов золота Au^{79+} и до 27 ГэВ для протонов. Предусмотрены две точки взаимодействия частиц в накопительных кольцах коллайдера NICA (рис. 1), в каждой из которых будет расположен детектор, ориентированный на свою физическую программу исследований: детекторы MPD (MultiPurpose Detector) [2] и SPD (Spin Physics Detector) [3]. Многоцелевой детектор MPD оптимизирован для всестороннего изучения свойств горячей и плотной ядерной материи, образуемой в столкновениях тяжелых ионов, и поиска критической точки фазового перехода. Детектор спиновой физики SPD комплекса NICA строится для исследования спиновой структуры нуклонов на высокоинтенсивных поляризованных пучках легких ядер.

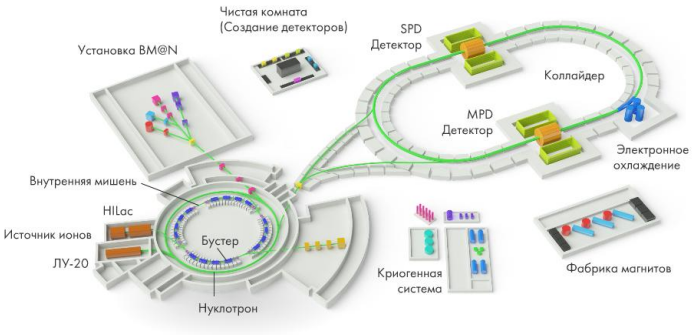


Рис. 1. Строящийся ускорительно-накопительный комплекс NICA

Кроме того, одним из основных элементов первого этапа проекта NICA является эксперимент на фиксированной мишени BM@N [4] (Baryonic Matter at Nuclotron, барионная материя на Нуклотроне), проводимый на пучках частиц, выводимых с Нуклотрона в экспериментальный зал. Технические сеансы эксперимента BM@N проводятся уже с 2015 года, а в 2018 году эксперимент был запущен с новой физической подпрограммой SRC [5] (Short Range Correlations, измерение двухнуклонных короткодействующих корреляций).

Для современных крупных научных исследований, таких как эксперименты проекта NICA, характерны длительность, сложность, высокая трудоемкость, большие временные затраты, оперирование большими объемами данных, получаемых в ходе эксперимента. Ожидается, что при беспрерывной работе в течении 4 – 6 месяцев в году и частоте столкновений 7 кГц только для эксперимента MPD будет набираться около 20 миллиардов событий (порядка 10-20 ПБ данных) в год. На данном этапе уже ведется работа с большим объемом, исчисляемым сотнями терабайт, смоделированных данных всех экспериментов проекта NICA и экспериментальных данных BM@N, только за последний сеанс которого было набрано сотни миллионов событий.

В этой связи особую актуальность приобретает задача автоматизации процессов сбора, хранения, обработки и анализа экспериментальных (а также моделированных) данных комплекса NICA. Автоматизация современного эксперимента невозможна без применения специализированного информационно-вычислительного обеспечения, позволяющего собирать, хранить и обрабатывать большое количество информации, управлять экспериментом в процессе его проведения, обслуживать одновременно большое количество оборудования установки и выполнять другие действия, необходимые для своевременного получения качественного физического результата. Неотъемлемой частью такого обеспечения для онлайн и офлайн обработки данных экспериментов являются базы данных различного назначения [6] и связанные с их использованием и поддержкой соответствующие информационные системы.

Проведенное авторами исследование [7] показало, что информационные системы используются во всех крупных экспериментах по столкновению частиц, и они стали важной частью программного обеспечения этих экспериментов, в частности экспериментов на Большом адронном коллайдере в CERN. Однако существующие решения по автоматизации процессов сбора, обработки и анализа данных физических экспериментов сильно зависят от специфики выполняемого эксперимента и являются их неотъемлемой частью.

Таким образом, в настоящее время практически ни один крупный эксперимент физики высоких энергий не обходится без создания и использования автоматизированных информационных систем, в связи с чем в рамках гранта РФФИ №18-02-40125 ведется разработка комплекса новых информационных систем для онлайн и офлайн обработки данных экспериментов проекта NICA, включающего, в том числе, базу данных состояний и условий работы систем эксперимента (далее – база данных состояний). Целью реализации таких систем комплекса и баз данных является повышение эффективности сбора, хранения, обработки и анализа данных и обеспечение членов коллаборации требуемыми информационными сервисами.

Текущая архитектура обработки данных экспериментов. Для понимания роли разрабатываемой базы данных состояний необходимо определить ее место в архитектуре онлайн и офлайн обработки данных эксперимента. Поток обработки получаемых с эксперимента данных организован следующим образом [8]. Система триггеров эксперимента выполняет онлайн отбор событий столкновения частиц согласно текущей физической программе. В соответствии с триггерными сигналами детекторы установки формируют фрагменты “сырых” (необработанных) данных, которые собираются Сборщиком Событий системы сбора данных (Data Acquisition System, DAQ) в события и затем поступают на временное хранилище – распределенную кластерную систему хранения, где полученные события используются в системах оператора, работающих в режиме онлайн, таких как: проверка качества необработанных данных, системе онлайн гистограммирования и графи-

ческом мониторе событий. С временного хранилища данные передаются в систему постоянного хранения. Здесь необработанные данные преобразуются в режиме офлайн в ROOT [9] формат и передаются на реконструкцию событий, после чего реконструированные данные используются в различных задачах физического анализа. Обработка экспериментальных (а также моделированных) данных выполняется на распределенных вычислительных системах проекта NICA. В рамках гранта разрабатывается комплекс информационных систем и соответствующий набор сервисов, которые обеспечат качественное управление, хранение и передачу информации различным подсистемам для дальнейшей обработки данных событий столкновения частиц.

Информация о сеансах эксперимента, о полученных экспериментальных и смоделированных файлах, а также различные параметры работы систем записываются в базу данных состояний как в онлайн, так и офлайн режиме. В дальнейшем эта сохраненная информация используется в различных алгоритмах обработки данных эксперимента, в том числе при моделировании работы установки, реконструкции полученных событий и их физическом анализе. Текущая схема обработки данных на примере действующего эксперимента BM@N проекта NICA показана на рис. 2.

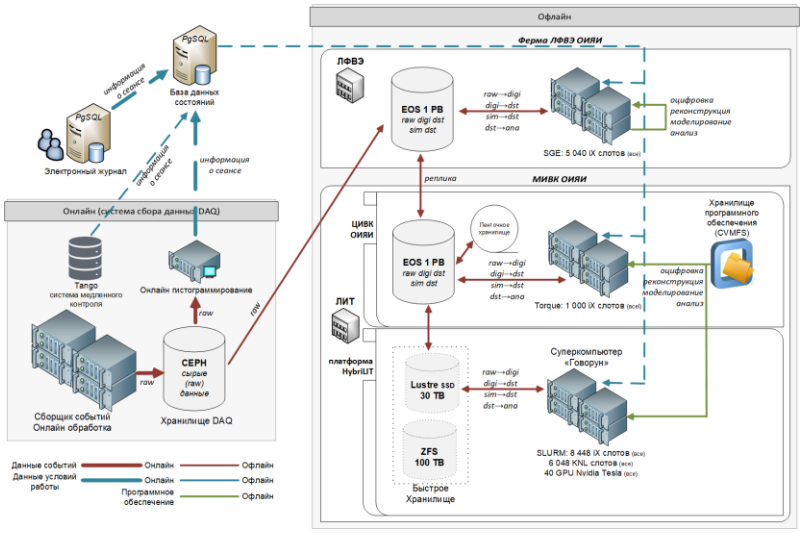


Рис. 2. Схема распределенной обработки экспериментальных данных

На рисунке представлено место разрабатываемой базы данных состояний в архитектуре онлайн и офлайн обработки данных событий. Часть информации, содержащая параметры проводимых сеансов, будет автоматически поступать в базу данных состояний из электронного журнала эксперимента [10]. Кроме того, система медленного контроля на базе программной среды Tango [11] также является источником параметров аппаратных подсистем, необходимых для дальнейшей обработки данных событий столкновения частиц. Как показано на рисунке, хранимая в базе данных состояний информация в дальнейшем используется при оцифровке необработанных данных событий, их реконструкции и физическом анализе, а также при моделировании работы установки на территориально-распределенных вычислительных системах Лаборатории физики высоких энергий и Лаборатории информационных технологий ОИЯИ (в настоящее время подключаются и ресурсы стран-участниц), включая современный суперкомпьютер «Говорун» [12].

Разработка базы данных состояний и режимов работы систем. Важной частью не только систем, работающих в режиме онлайн, но и задач, выполняемых офлайн, включая обработку и анализ полученных физических данных, являются информационные системы, построенные на современных базах данных и предлагающие различные пользовательские сервисы для прозрачного доступа и управления хранимыми данными и информацией о проводимом эксперименте.

База данных состояний (англ. condition database) направлена на хранение, обработку и использование параметров и режимов работы различных устройств и детекторов установки в офлайн (а возможно и онлайн) системах обработки данных эксперимента, в том числе в алгоритмах реконструкции и физического анализа событий столкновения частиц. Соответствующая информационная система решает также задачу удобного доступа и управления требуемыми параметрами подсистем установки для их учета на всех этапах обработки данных эксперимента.

В ходе проектирования была сформирована по методологии IDEF1x [13] следующая диаграмма базы данных состояний, представленная на рис. 3. В процессе разработки дизайна структура базы данных была максимально упрощена и приведена к общей схеме, которая может быть использована в разных физических экспериментах по столкновению частиц.

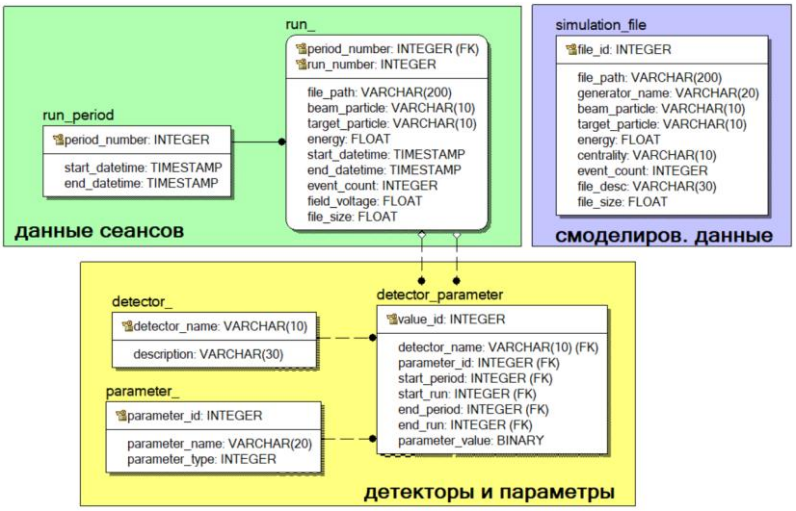


Рис. 3. Физическая модель разрабатываемой базы данных состояний

База данных состояний разрабатывается как центральное хранилище информации об условиях и режимах работы различных подсистем, а также параметров проводимого эксперимента. Эти данные необходимы для реконструирования и анализа записанных данных событий, моделирование работы детекторов, а также являются ключевым элементом на этапе обработки полученных экспериментальных данных. Соответствующая информационная система обеспечивает управление хранимыми данными и унифицированный доступ к ним для онлайн и офлайн систем обработки данных, гарантируя корректную многопользовательскую обработку, актуальность информации, к которой предоставляется доступ, согласованность и целостность данных, исключает многократное дублирование и использование устаревших данных. Кроме того, автоматическое регулярное резервное копирование хранимой информации гарантирует ее сохранность в случае программных ошибок или аппаратных сбоев.

Проведенный анализ показал, что база данных состояний для экспериментов проекта NICA будет иметь относительно небольшой размер, поэтому ее разработка ведется на реляционной СУБД (системе управления базами данных) PostgreSQL [14]. В архитектуре создаваемой базы данных можно выделить 3 компонентные части (рисунок 3): хранение информации о смоделированных данных, информации о проведенных сеансах эксперимента, включая данные по магнитному полю, типе частицы пучка и энергии, времени, количеству событий столкновений частиц и файлах экспериментальных данных, а также хранение параметров различного типа для аппаратных подсистем эксперимента. Выделены 4 группы хранимых параметров:

- конфигурационные данные, связанные с режимом работы детекторов, включая программируемые параметры внешней электроники;
- калибровочные данные, описывающие калибровку и выравнивание в соответствие с заданным расположением детекторов, которые обычно вычисляются с помощью специальных алгоритмов после сеансов;
- параметрические данные, описывающие состояние детекторных и аппаратных подсистем эксперимента;
- алгоритмические данные, задающие как сами алгоритмы обработки данных событий столкновения частиц, так и условия их работы.

Важным свойством параметров, хранимых в базах данных состояний и режимов работы систем в экспериментах физики высоких энергий, является то, что они меняются с течением времени [15]. Значение или набор значений параметра описывают состояние системы в течение ограниченного промежутка времени и используются только для анализа событий, полученных в этом интервале действия. В связи с этим хранимые параметры характеризуются привязкой к интервалу времени во время сеанса, в течение которого они действительны. Обычно в качестве соответствующего атрибута параметра используется временной период действия [16] или последовательный набор измерений – “ранов” (относительно небольших отрезков работы эксперимента, данные которых зачастую сохраняются в отдельном файле), а для выборки параметра передается соответствующий момент времени или номер “рана”. В спроектированной базе данных состояний для экспериментов проекта NICA период валидности параметра определяется набором “ранов”, для которых он действителен.

В настоящее время рассматривается реализация информационной системы на базе клиент-серверной модели, состоящей из центральной базы данных состояний и локальных пользовательских реплик (копий) [17]. В этом случае будет осуществляться онлайн и офлайн запись данных в централизованную базу данных, а для быстрого доступа к хранимой информации будут использоваться обновляемые реплики центральной базы данных состояний. Также это обеспечит возможность обрабатывать данные в программной среде эксперимента членами коллаборации при отсутствии связи с сервером базы данных, например, при отсутствии интернета.

Еще одной важной особенностью разрабатываемой базы данных состояний является то, что ее архитектура обеспечивает хранение параметров любого, заранее определенного типа в виде двоичных объектов, сохраняемых в одно поле базы данных (“*parameter_value*” на рис. 3) и сериализуемых через специализированный программный интерфейс.

Интеграция базы данных состояний с системами обработки данных эксперимента. Для обработки моделированных и экспериментальных данных экспериментов проекта NICA разрабатываются соответствующие программные пакеты [18] на языке программирования C++, базирующиеся на программном обеспечении FairRoot [19] коллаборации FAIR (института GSI Германии). FairRoot предоставляет общие классы и механизмы, используемые при решении задач в физиче-

ских экспериментах по столкновению частиц, программным пакетам экспериментов BM@N, MPD и SPD: BmnRoot, MPDRoot и SPDRoot соответственно. Основные задачи обработки данных в этих пакетах решаются при помощи реализованных макросов среды ROOT.

Программное обеспечение экспериментов должно иметь доступ к информации, хранимой в базе данных состояний, для использования ее в задачах оцифровки сигналов, моделирования, реконструкции и физического анализа событий столкновения частиц. Для интеграции базы данных состояний с программными пакетами экспериментов BmnRoot [20], MPDRoot и SPDRoot, базирующихся на среде CERN ROOT, в настоящее время ведется разработка специализированного программного интерфейса на языке C++, позволяющего использовать хранимую информацию при онлайн и офлайн обработке данных.

Кроме того, начата разработка пользовательского интерфейса в виде вебсервиса для упрощения просмотра и управления информацией об эксперименте членами коллаборации через интернет. Также необходима реализация удобного инструмента поиска, который обеспечит формирование критериев выбора и эффективный поиск в базе данных состояний информации, необходимой для обработки данных или физического анализа событий столкновения частиц.

Заключение. Разрабатываемая база данных состояний является важным компонентом реализуемого комплекса информационных систем и представляет собой централизованное хранилище, обеспечивающее единый доступ онлайн и офлайн системам эксперимента к хранимым параметрам для обработки и анализа данных экспериментальных установок комплекса NICA, а также управление членами коллаборации актуальной информацией об эксперименте. Такие системы вносят существенный вклад в решение задачи автоматизации сбора, хранения, обработки и анализа данных, а также являются необходимым элементом для успешной работы современных экспериментов физики высоких энергий и своевременного получения качественного физического результата.

Поддержка. Работа выполнена при финансовой поддержке РФФИ в рамках научного проекта №18-02-40125. Авторы выражают благодарность команде гетерогенной вычислительной платформы HybriLIT (ЛИТ, ОИЯИ) за предоставление программно-аппаратных ресурсов для работы.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *NICA Collaboration*. NICA White paper. Searching for a QCD mixed phase at the Nuclotron-based ion collider facility. – JINR, Dubna, 2014. – 334 p.
2. *MPD Collaboration*. The MultiPurpose Detector – MPD. Conceptual Design Report. – JINR, Dubna, 2012. – 259 p.
3. *Kouznetsov O., Savin I.* Spin Physics Experiments at NICA-SPD // Nuclear and Particle Physics Proceedings. – 2017. – Vol. 282–284. – P. 20-26.
4. *Baranov D., Kapishin M., Mamontova T., et al*. The BM@N experiment at JINR: status and physics program // KnE Energy & Physics. – 2018. – Vol. 3. – P. 291 -296.
5. *Galavanov A., Khabarov S., Kirushin, et al*. Studies of Short Range Correlations in inverse kinematics at BM@N at the NICA facility // Journal of Physics: Conference Series. – 2019. – Vol. 1390. – 012025.
6. *Shiers J.* Databases in High Energy Physics: A Critical Review // In: R. Brun, F. Carminati, G. Galli Carminati. From the Web to the Grid and Beyond. – Springer, 2011. – P. 225-266.
7. *Александров Е.И., Александров И.Н., Герценбергер К.В. и др*. Информационные системы для онлайн и офлайн обработки данных в современных экспериментах физики высоких энергий // Международный научный журнал «Современные информационные технологии и ИТ-образование», – 2019. – Т. 15, № 3. – С. 645-650.
8. *Baskakov A., Bazylev S., Fediunin A., Filippov I.* MPD Data Acquisition System: Technical Design Report. – JINR, Dubna, 2018. – 74 p.

9. *Brun R., Rademakers F.* ROOT – An Object Oriented Data Analysis Framework // Proceedings AIHENP'96 Workshop. – Nucl. Inst. & Meth. in Phys. Res. A. – 1997. – Vol. 389. – P. 81-86.
10. *Gertsenberger K., Moshkin A., Chebotov A.* Development of the Electronic Logbook for the BM@N Experiment at NICA // CEUR Workshop Proceedings of the 27th International Symposium Nuclear Electronics and Computing. – 2019. – Vol. 2507. – P. 175-179.
11. *Andreev V., Volkov V., Gorbachev E., et al*. TANGO standard software to control the Nuclotron beam slow extraction // Physics of Particles and Nuclei Letters. – 2016. – Vol. 13. – P. 605-608.
12. *Adam Gh., Bashashin M., Belyakov D., et al*. IT-ecosystem of the HybriLIT heterogeneous platform for high-performance computing and training of IT-specialists // CEUR Workshop Proceedings of the VIII International Conference «Distributed Computing and Grid-technologies in Science and Education». – 2018. –Vol. 2267. – P. 638–644.
13. *Serifi V., Dasic P., Jecmenica R., Labovic D.* Functional and information modeling of production using IDEF methods // Journal of Mechanical Engineering. – 2009. – Vol. 55. – P. 131-140.
14. *Obe R., Hsu L*. PostgreSQL: up and running. – O'Reilly Media, 2015. – 234 p.
15. *Laycock P., Dykstra D., Formica A., et al*. A Conditions Data Management System for HEP Experiments // Journal of Physics: Conference Series. – 2018. – Vol. 1085. – 032040.
16. *Rinaldi L., Formica A., Gallas E., et al*. Conditions evolution of an experiment in mid–life, without the crisis (in ATLAS) // EPJ Web of Conferences. – 2019. – Vol. 214. – 04052.
17. *Akishina E., Alexandrov E., Alexandrov I., et al*. Conceptual considerations for CBM databases // Communication of the Joint Institute for Nuclear Research. – 2014. – E10-2014-103.
18. *Gertsenberger K., Merts S., Rogachevsky O., Zinchenko A*. Simulation and analysis software for the NICA experiments // European Physical Journal A. – 2016. – Vol. 52 (8). – 214.
19. *Al-Turany M., Bertini D., Karabowicz R., et al*. The FairRoot framework // Journal of Physics: Conference Series. – 2012. – Vol. 396. – 022001.
20. *Batyuk P., Gertsenberger K., Merts S., Rogachevsky O*. The BmnRoot framework for experimental data processing in the BM@N experiment at NICA // EPJ Web of Conferences. – 2019. – Vol. 214. – 05027.

REFERENCES

1. *NICA Collaboration*. NICA White paper. Searching for a QCD mixed phase at the Nuclotron-based ion collider facility. JINR, Dubna, 2014, 334 p.
2. *MPD Collaboration*. The MultiPurpose Detector – MPD. Conceptual Design Report. JINR, Dubna, 2012, 259 p.
3. *Kouznetsov O., Savin I.* Spin Physics Experiments at NICA-SPD, *Nuclear and Particle Physics Proceedings*, 2017, Vol. 282–284, pp. 20-26.
4. *Baranov D., Kapishin M., Mamontova T., et al*. The BM@N experiment at JINR: status and physics program, *KnE Energy & Physics*, 2018, Vol. 3, pp. 291 -296.
5. *Galavanov A., Khabarov S., Kirushin, et al*. Studies of Short Range Correlations in inverse kinematics at BM@N at the NICA facility, *Journal of Physics: Conference Series*, 2019, Vol. 1390, 012025.
6. *Shiers J.* Databases in High Energy Physics: A Critical Review, *In: R. Brun, F. Carminati, G. Galli Carminati. From the Web to the Grid and Beyond*. Springer, 2011, pp. 225-266.
7. *Aleksandrov E.I., Aleksandrov I.N., Gertsenberger K.V. i dr.* Informatsionnye sistemy dlya onlayn i oflayn obrabotki dannykh v sovremennykh eksperimentakh fiziki vyokikh energiy [Information systems for online and offline data processing in modern high-energy physics experiments], *Mezhdunarodnyy nauchnyy zhurnal «Sovremennye informatsionnye tekhnologii i IT-obrazovanie»* [International scientific journal «Modern Information Technologies and IT-Education»], 2019, Vol. 15, No. 3, pp. 645-650.
8. *Baskakov A., Bazylev S., Fediunin A., Filippov I.* MPD Data Acquisition System: Technical Design Report. JINR, Dubna, 2018, 74 p.
9. *Brun R., Rademakers F.* ROOT – An Object Oriented Data Analysis Framework, *Proceedings AIHENP'96 Workshop. Nucl. Inst. & Meth. in Phys. Res. A*., 1997, Vol. 389, pp. 81-86.
10. *Gertsenberger K., Moshkin A., Chebotov A.* Development of the Electronic Logbook for the BM@N Experiment at NICA, *CEUR Workshop Proceedings of the 27th International Symposium Nuclear Electronics and Computing*, 2019, Vol. 2507, pp. 175-179.

11. *Andreev V., Volkov V., Gorbachev E., et al*. TANGO standard software to control the Nuclotron beam slow extraction, *Physics of Particles and Nuclei Letters*, 2016, Vol. 13, pp. 605-608.
12. *Adam Gh., Bashashin M., Belyakov D., et al*. IT-ecosystem of the HybriLIT heterogeneous platform for high-performance computing and training of IT-specialists, *CEUR Workshop Proceedings of the VIII International Conference «Distributed Computing and Grid-technologies in Science and Education»*, 2018, Vol. 2267, pp. 638–644.
13. *Serifi V., Dasic P., Jecmenica R., Labovic D*. Functional and information modeling of production using IDEF methods, *Journal of Mechanical Engineering*, 2009, Vol. 55, pp. 131-140.
14. *Obe R., Hsu L*. PostgreSQL: up and running. – O'Reilly Media, 2015. – 234 p.
15. *Laycock P., Dykstra D., Formica A., et al*. A Conditions Data Management System for HEP Experiments, *Journal of Physics: Conference Series*, 2018, Vol. 1085, 032040.
16. *Rinaldi L., Formica A., Gallas E., et al*. Conditions evolution of an experiment in mid–life, without the crisis (in ATLAS), *EPJ Web of Conferences*, 2019, Vol. 214, 04052.
17. *Akishina E., Alexandrov E., Alexandrov I., et al*. Conceptual considerations for CBM databases, *Communication of the Joint Institute for Nuclear Research*, 2014, E10-2014-103.
18. *Gertsenberger K., Merts S., Rogachevsky O., Zinchenko A*. Simulation and analysis software for the NICA experiments, *European Physical Journal A*, 2016, Vol. 52 (8), 214.
19. *Al-Turany M., Bertini D., Karabowicz R., et al*. The FairRoot framework, *Journal of Physics: Conference Series*, 2012, Vol. 396, 022001.
20. *Batyuk P., Gertsenberger K., Merts S., Rogachevsky O*. The BmnRoot framework for experimental data processing in the BM@N experiment at NICA, *EPJ Web of Conferences*, 2019, Vol. 214, 05027.

Статью рекомендовал к опубликованию д.ф.-м.н. В.С. Пантуев.

Герценбергер Константин Викторович – Объединенный институт ядерных исследований; e-mail: gertsen@jinr.ru; 141980, г. Дубна, Московская обл., ул. Жолио-Кюри, 6; тел.: +74962165343; Лаборатория физики высоких энергий им. В.И. Векслера и А.М. Балдина; к.т.н.; начальник группы.

Чеботов Александр Игоревич – e-mail: chebotov@jinr.ru; тел.: +74962165343; Лаборатория физики высоких энергий им. В.И. Векслера и А.М. Балдина; инженер-программист.

Александров Игорь Николаевич – e-mail: aleksand@jinr.ru; Лаборатория информационных технологий; к.ф.-м.н.; начальник сектора.

Филозова Ирина Анатольевна – e-mail: fia@jinr.ru; Лаборатория информационных технологий; начальник группы.

Александров Евгений Игоревич – e-mail: aleksand@jinr.ru; Лаборатория информационных технологий; научный сотрудник.

Gertsenberger Konstantin Viktorovich – Joint Institute for Nuclear Research; e-mail: gertsen@jinr.ru; 6, Joliot-Curie St, Dubna, Moscow Region, 141980, Russia; phone: +74962165343; Veksler and Baldin Laboratory of High Energy Physics; cand. of eng. sc.; head of group.

Chebotov Alexander Igorevich – e-mail: chebotov@jinr.ru; phone: +74962165343; Veksler and Baldin Laboratory of High Energy Physics; software engineer.

Alexandrov Igor Nikolaevich – e-mail: aleksand@jinr.ru; Laboratory of Information Technologies; cand. of phys.-math. sc.; head of sector.

Filozova Irina Anatolyevna – e-mail: fia@jinr.ru; Laboratory of Information Technologies; head of group.

Alexandrov Evgeny Igorevich – e-mail: alexand@jinr.ru; Laboratory of Information Technologies; researcher.

ПРАВИЛА ОФОРМЛЕНИЯ РУКОПИСЕЙ

1. Объем статьи должен быть не менее 12 и не более 18 страниц. Формат (А 4). Редактор ***Word 7 for Windows***, шрифт Times New Roman, размер 14, интервал 1,5. Авторы представляют в редакцию 1 экз. статьи и идентичный электронный вариант.

2. Названию статьи предшествует индекс УДК, соответствующий заявленной теме.

3. Текст статьи начинается с названия статьи (на русском и английском языках), фамилии, имени и отчества автора (полностью) и снабжается аннотацией на русском и английском языках объёмом ***не менее 250-300 слов***. В тексте аннотации указывается цель, задачи исследования и краткие выводы. В аннотации ***не следует*** давать ссылки на номер публикации в списке литературы к статье. После аннотаций приводятся ключевые слова (словосочетания), несущие в тексте основную смысловую нагрузку (на русском и английском языках).

4. В тексте статьи следует использовать минимальное количество таблиц и иллюстраций. Рисунок должен иметь объяснения значений всех компонентов, порядковый номер, название, расположенное под рисунком. В тексте на рисунок дается ссылка. Таблица должна иметь порядковый номер, заголовок, расположенный над ней. Данные таблиц и рисунков не должны дублировать текст. Формулы должны быть набраны ***в редакторе формул Word 7 for Windows.***

5. Цитаты тщательно сверяются с первоисточником и визируются автором на обратной стороне последней страницы: "Цитаты и фактический материал сверены". Подпись, дата.

6. Наличие пристатейного библиографического списка на русском и английском языках обязательно. ***Ссылок должно быть не менее 20-ти,*** из них на зарубежные источники – не менее 35 %. В тексте ссылки должны быть в квадратных скобках.

Примеры оформления литературы: а) для книг: фамилия, инициалы автора(ов), полное название книги, место, год издания, страницы; б) для статей: фамилия и инициалы автора(ов), полное название сборника, книги, газеты, журнала, где опубликована статья, место и год издания (сборника, книги), номер (для журнала), год и дата (для газеты), выпуск, часть (для сборника), страницы, на которых опубликована статья. Иностранная литература оформляется по тем же правилам.

Ссылки на неопубликованные работы не допускаются.

7. Рукопись должна быть тщательно вычитана. Редакционная коллегия оставляет за собой право при необходимости сокращать статьи, редактировать и отсылать авторам на доработку.

8. Статьи сопровождаются сведениями об авторе(ах) (фамилия, имя, отчество, ученое звание, должность, место работы, адрес, электронный адрес и номер телефона) на русском и английском языках.

9. Плата с аспирантов за публикацию рукописей не взымается.

Адрес журнала в Интернете: http://izv-tn.tti.sfedu.ru/.