МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное
учреждение высшего образования
ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ
Институт компьютерных технологий и информационной безопасности
Кафедра интеллектуальных и многопроцессорных систем

ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСТВЕННОСТЬЮ
НИЦ СУПЕР–ЭВМ И НЕЙРОКОМПЬЮТЕРОВ

На правах рукописи

Дордопуло Алексей Игоревич

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ ТЕХНОЛОГИИ РЕСУРСОНЕЗАВИСИМОГО ПРОГРАММИРОВАНИЯ ГИБРИДНЫХ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

Специальность: 2.3.5 – Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей, технические науки

АВТОРЕФЕРАТ
диссертации на соискание ученой степени

доктора технических наук

Научный консультант:
доктор технических наук,
профессор **И.И. Левин**

Таганрог – 2023 г.

Работа выполнена в ООО «Научно–исследовательский центр супер–ЭВМ и нейрокомпьютеров» (НИЦ супер–ЭВМ и нейрокомпьютеров) и Федеральном государственном автономном образовательном учреждении высшего образования «Южный федеральный университет» (ЮФУ), в Институте компьютерных технологий и информационной безопасности, на кафедре «Интеллектуальные и многопроцессорные системы» (ИМС).

НАУЧНЫЙ КОНСУЛЬТАНТ: доктор технических наук, профессор, директор НИЦ супер–ЭВМ и нейрокомпьютеров **Левин Илья Израилевич**

ОФИЦИАЛЬНЫЕ ОППОНЕНТЫ: доктор физико-математических наук, профессор, член–корреспондент РАН, директор Научно–исследовательского вычислительного центра МГУ им. М.В. Ломоносова, **Воеводин Владимир Валентинович**

доктор технических наук, профессор департамента программной инженерии факультета компьютерных наук Высшей школы экономики, **Легалов Александр Иванович**

доктор физико-математических наук, профессор кафедры системного программирования Высшей школы электроники и компьютерных наук Южно-Уральского государственного университета **Цымблер Михаил Леонидович**

Защита диссертации состоится «06» октября 2023 г. в 14^{00} на заседании диссертационного совета ЮФУ 801.02.06 при Южном федеральном университете по адресу: г. Таганрог, ул. Чехова, 2, корп. “И”, комн. 347.

С диссертацией можно ознакомиться в библиотеке и на сайте Южного федерального университета http://hub.sfedu.ru/diss/.

Автореферат разослан «__» _____ 2023 г.

Ученый секретарь диссертационного совета кандидат технических наук, доцент **А.П. Кухаренко**

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы исследования. Современные исследования в химии и медицине, гео– и астрофизике, энергетике и космонавтике, экономике и социологии невозможны без высокопроизводительных вычислений (high performance computing, HPC), которые входят в число приоритетных направлений развития науки и техники большинства ведущих стран мира. Более 30 лет рост производительности основных компонентов вычислительных систем обеспечивался достижениями микроэлектроники. В соответствии с эмпирическим законом Мура об удвоении числа транзисторов в микросхемах каждые 1,5 года улучшение технологии производства микропроцессоров повышало быстродействие устройств. Исчерпание технологических возможностей миниатюризации транзисторов, наряду с известными проблемами фон–Неймановской процессорной архитектуры, такими как «бутылочное горло» и «стена памяти», существенно замедлило привычные темпы роста производительности, что, в свою очередь, актуализировало проблему эффективной реализации вычислений. Эффективность реализации задачи в вычислительной системе оценивается отношением *реальной*, достигаемой при ее решении, производительности к теоретически возможной, *пиковой.* Эффективность реализации задачи характеризует соответствие архитектуры вычислительной системы структуре задачи и для большинства многопроцессорных вычислительных систем (МВС) не превышает 10–15%, для ряда практических задач снижаясь еще больше – до нескольких процентов или даже долей процента.

Одним из перспективных способов повышения эффективности реализации широкого круга задач на высокопроизводительных вычислительных системах стало объединение процессоров и специализированных вычислительных компонент в гибридную вычислительную систему (ГВС). Наглядным подтверждением этому служат последние редакции списка самых производительных мировых суперкомпьютеров TOP500, в котором первые позиции принадлежат гибридным и гетерогенным вычислительным системам, таким как Summit, Fugaku, TaihuLight и др. В гибридной вычислительной системе, содержащей процессоры, графические ускорители и программируемые логические интегральные схемы (ПЛИС), в отличие от гетерогенной, ни одна из архитектур не является преобладающей. В то же время теоретически ожидаемый рост эффективности реализации задач в ГВС не обеспечивается при простом совмещении узлов различных архитектур на основе «федеративного» принципа, когда вычисления выполняются только узлами, архитектура которых наилучшим образом соответствует решаемой задаче. Для повышения эффективности реализации прикладных задач необходимо использовать все узлы ГВС.

Совместное использование различных узлов ГВС для решения задачи требует от программиста обширных и глубоких знаний разных моделей представления вычислений, особенностей языков и технологий программирования, специфичных для каждой архитектуры, а также очень высокой квалификации для согласования фрагментов, выполняющихся на различных узлах системы. Согласование фрагментов вычислений в узлах разных типов необходимо не только при создании, но и при каждой корректировке программы, вызванной рутинным изменением текущей конфигурации ГВС в связи с недоступностью одного или группы узлов из–за отказа оборудования и/или коммутационной системы. Для согласования вычислений в разных узлах ГВС при изменении доступного вычислительного ресурса необходима корректировка (портация) параллельной программы, требующая от программиста усилий и времени, в ряде случаев сравнимого с временем разработки новой программы. Более строго, портация (от англ.

porting, портирование или перенос) понимается как адаптация программы или её фрагмента для выполнения в другой среде и/или вычислительной архитектуре с максимальным сохранением пользовательских свойств. Портируемость или переносимость является свойством программного кода или исполняемого модуля программы. Как правило, портируемость обеспечивается с помощью общего (как язык программирования С) или промежуточного (как аппаратно–независимый байт–код Java) представления кода для несовместимых аппаратных платформ. В ряде случаев используются специальные программы, преобразующие аппаратные инструкции исполняемого модуля одной архитектуры в инструкции другой специальным транслятором перед выполнением или «на лету».

Для вычислительных задач портацию будем понимать как преобразование вычислений с созданием исполняемых модулей для узлов ГВС. Трудоемкость разработки и портации программы зависит от количества различных вычислительных архитектур в ГВС, структуры задачи и числа используемых технологий программирования, поскольку значительные различия между ними не позволяют в настоящее время использовать одну единую технологию для эффективного программирования всех узлов ГВС. Причина этого заключается в отсутствии известных теоретических методов и технологий преобразования и/или портации вычислений для различных архитектур и конфигураций ГВС при совместном использовании вычислительных узлов для решения задачи.

В этой связи необходимы новые теоретические основы организации параллельных вычислений в едином контуре, учитывающие как внутренний параллелизм и информационные зависимости задач, так и особенности вычислительных архитектур конвергентной мультипроблемно–ориентированной вычислительной системы. Теоретические основы должны обеспечивать преобразование вычислений к неопределенному заранее аппаратному ресурсу гибридной вычислительной системы с учетом многообразия существующих и потенциально возможных вычислительных архитектур. Целесообразно, чтобы аппаратный ресурс ГВС был параметром представления задачи, с изменением которого параллельные вычисления с помощью научно–обоснованных методов могли быть преобразованы (портированы) к заданной архитектуре и конфигурации гибридной вычислительной системы. Параметризация представления задачи аппаратным ресурсом позволит создавать независимые от архитектуры и конфигурации или *ресурсонезависимые* параллельные программы, рационально использующие особенности вычислительных архитектур ГВС. Принципы организации и методы преобразования ресурсонезависимых вычислений при решении прикладной задачи на узлах различных архитектур ГВС позволят разработать инструментальные средства портации и новую технологию программирования гибридных вычислительных систем.

Поэтому **актуальна разработка теоретических основ ресурсонезависимого программирования гибридных вычислительных систем** – новых научных понятий, методологических принципов, положений и выявленных закономерностей, позволяющих представить вычисления для различных архитектур в единой модели и преобразовать ее с помощью формализованных методов и алгоритмов в модель реализации для гетерогенного вычислительного ресурса, содержащего вычислительные узлы на основе процессоров, графических ускорителей и программируемых логических интегральных схем (ПЛИС). Совокупность теоретических основ, методов преобразования вычислений и инструментальных программных средств позволит создать технологию ресурсонезависимого программирования гибридных вычислительных систем, обеспечивающую сбалансированное решение задачи для заранее неопределенного

аппаратного ресурса ГВС из различного числа узлов разных архитектур, объединенных в единый вычислительный ресурс. Формализация методов и приемов создаваемой технологии ресурсонезависимого программирования позволит сократить время и автоматизировать портацию (преобразование, трансформацию или перенос) прикладных задач для всего многообразия архитектур и конфигураций ГВС при заданном уровне реальной производительности (эффективности).

Объектом исследования являются технологии программирования высокопроизводительных вычислительных систем.

Актуальная **научная проблема,** решаемая в диссертации – разработка теоретических основ, методов и инструментальных программных средств ресурсонезависимого программирования, обеспечивающих сокращение времени портации прикладных задач для гибридных вычислительных систем различных архитектур и конфигураций при заданном уровне реальной производительности.

Цель диссертационной работы – сокращение времени портации параллельных программ для гибридных вычислительных систем.

В общей сформулированной **научной проблеме** можно выделить несколько частных задач, для решения которых необходимо:

1 Провести анализ современных вычислительных архитектур и тенденций их развития, определить их ограничения при эффективной реализации прикладных задач различных классов для высокопроизводительных гетерогенных вычислительных систем.

2 Проанализировать известные методы и технологии программирования высокопроизводительных вычислительных систем, сформулировать требования и принципы программирования конвергентных архитектур гибридных вычислительных систем.

3 Разработать теоретические основы ресурсонезависимого программирования: единую форму вычислений, параметром которой является доступный аппаратный ресурс, и методы ее преобразования к различным архитектурам гибридной вычислительной системы.

4 Разработать методику портации задач к рациональной реализации на заранее неопределенном вычислительном ресурсе гибридной вычислительной системы.

5 Разработать методы и алгоритмы портации прикладных задач различных классов, содержащих связанные информационной зависимостью подзадачи с различной степенью параллелизма.

6 Разработать инструментальные программные средства технологии ресурсонезависимого программирования, реализующие разработанные принципы, методы и алгоритмы портации прикладных задач на архитектуру и конфигурацию гибридной вычислительной системы при заданном уровне реальной производительности.

7 Провести апробацию технологии ресурсонезависимого программирования при решении прикладных задач различных предметных областей и сравнить результаты с известными решениями.

Степень разработанности темы

Исследованиями, в той или иной мере связанными с тематикой поставленных в диссертации задач, занимались такие известные ученые, как Э. Дейкстра, Ч. Хоар, Р. Хокни, В. Хейдлер, Дж. Донгарра, Т.Стерлинг (за рубежом) и В.С. Бурцев, В.Г. Хорошевский, В.В. и Вл.В. Воеводин, А.И. Легалов, А.О. Лацис, В.А. Крюков, В.А. Вальковский, Б.Я. Штейнберг, А.В. Каляев, И.А. Каляев, И.И. Левин и многие другие исследователи, внесшие значительный вклад в теорию и практику программирования высокопроизводительных вычислительных систем.

В работе получены следующие **основные новые научные результаты**:

1. Модель параллельных вычислений для гибридных вычислительных систем на основе информационного графа задачи, **отличающаяся** от известных единой для различных вычислительных архитектур формой описания вычислений в виде кадровой структуры.
2. **Новое** представление кадровой структуры с использованием вычислительного ресурса как параметра параллельно–конвейерной реализации задачи на заранее неопределенном ресурсе гибридной вычислительной системы.
3. **Новые** принципы преобразования параметризованной аппаратным ресурсом кадровой структуры к архитектуре и конфигурации гибридной вычислительной системы с помощью редукции производительности.
4. Теоремы о применении методов редукции производительности для преобразования кадровой структуры к архитектуре и конфигурации гибридной вычислительной системы.
5. Метод преобразования кадровой структуры и сокращения аппаратных затрат с помощью редукции производительности, **отличающийся** использованием доступного аппаратного ресурса гибридной вычислительной системы как параметра портации.
6. **Новый** метод преобразования и сокращения аппаратных затрат кадровой структуры на основе синтеза последовательности микро–кадров, выполняющих вычисления структурно с сокращенной степенью параллелизма на ограниченном доступном ресурсе (меньшем ресурса аппаратной реализации базового подграфа).
7. Метод преобразования задач, содержащих связанные информационной зависимостью подзадачи с разной степенью параллелизма, **отличающийся** от известных согласованной редукцией производительности различных по вычислительной трудоемкости кадровых структур с синтезом сбалансированного по интервалу обработки данных решения.
8. Теоретические основы технологии ресурсонезависимого программирования гибридных вычислительных систем, объединяющие совокупность предложенных научных понятий и положений, выявленных закономерностей и принципов, разработанных методов и алгоритмов портации в достаточную для синтеза рационального решения задачи методику.
9. **Новые** алгоритмы функционирования инструментальных программных средств технологии ресурсонезависимого программирования гибридных вычислительных систем: алгоритм преобразования параметризованной ресурсом кадровой структуры прикладной задачи к целевой архитектуре ГВС программой «Прокруст» и алгоритм редукции производительности выделенного фрагмента задачи в заданное коэффициентом редукции число раз при нехватке аппаратного ресурса программой «Щелкунчик».

Новизна полученных результатов подтверждается отсутствием аналогов, опубликованных в открытых источниках.

Положения, выдвигаемые для защиты:

1. Современные технологии программирования высокопроизводительных вычислительных систем не обеспечивают единое представление параллельных вычислений для процессоров, графических ускорителей и ПЛИС из–за различий моделей и архитектурной специфики, что исключает автоматическую портацию прикладных программ на различные архитектуры и конфигурации гибридных вычислительных систем.

2. Единое представление параллельных вычислений для различных вычислительных архитектур может быть построено на основе информационного графа задачи, отображаемого в максимально параллельную кадровую структуру, которая в зависимости от доступного аппаратного ресурса ГВС может быть преобразована в структуру с меньшей степенью параллелизма.

3. Преобразование параметризованной аппаратным ресурсом кадровой структуры к заданной архитектуре и конфигурации гибридной вычислительной системы реализуемо формальными редукционными преобразованиями сокращения производительности и аппаратных затрат, число вариантов которых существенно меньше числа вариантов, анализируемых распараллеливающим компилятором для систем с распределенной памятью.

4. Совокупность разработанных формальных методов редукции производительности и аппаратных затрат, алгоритмов и инструментальных программных средств преобразования параметризованной аппаратным ресурсом кадровой структуры, составляющих основу технологии ресурсонезависимого программирования, сокращает время портации параллельно–конвейерных программ на заданную архитектуру и конфигурацию гибридной вычислительной системы, в том числе для задач, содержащих информационно–зависимые подзадачи с различной степенью параллелизма.

Результаты, выдвигаемые для защиты:

1. Теоремы о применении редукции производительности для преобразования кадровой структуры к архитектуре и конфигурации гибридной вычислительной системы.

2. Метод преобразования параметризуемой аппаратным ресурсом кадровой структуры к архитектуре и конфигурации гибридной вычислительной системы на основе редукции производительности, сокращающий ее аппаратные затраты.

3. Метод преобразования кадровой структуры к новой форме организации вычислений – микро–кадрам, выполняющим вычисления с меньшей степенью параллелизма на ограниченном подмножестве функциональных устройств и информационных каналов.

4. Метод преобразования задач с информационно–зависимыми подзадачами с разной степенью параллелизма, на основе согласованной редукции производительности различных по вычислительной трудоемкости кадровых структур с синтезом сбалансированного по интервалу обработки данных решения.

Практическая значимость работы. Все разработанные методы преобразования кадровой структуры формализованы и адаптированы для автоматического применения. Разработанные теоретические основы, методы преобразования, алгоритмы и программные средства портации позволили создать технологию ресурсонезависимого программирования гибридных вычислительных систем, апробированную при решении ряда прикладных задач, содержащих связанные информационной зависимостью подзадачи с разной степенью параллелизма. Применение разработанных инструментальных средств технологии ресурсонезависимого программирования для портации прикладных задач математической физики, символьной и цифровой обработки сигналов с различными видами информационной зависимости позволяет в 3–6 раз сократить время портации ресурсонезависимых программ на различные архитектуры и конфигурации гибридных вычислительных систем с обеспечением реальной производительности не ниже заданного уровня. Практическая значимость работы и полученные результаты эффективности портации ресурсонезависимых программ подтверждаются актами внедрения результатов в академических, научно–исследовательских и промышленных организациях. Внедрение разработанной

технологии ресурсонезависимого программирования гибридных вычислительных систем открывает возможность создания автоматического распараллеливающего компилятора для систем с распределенной памятью, позволяющего сократить время портации программ на процедурных языках программирования в ресурсонезависимые параллельно–конвейерные решения для гибридных вычислительных систем.

Реализация результатов работы. Основные результаты диссертационных исследований были использованы более чем в 25 научно–исследовательских и опытно–конструкторских работах, среди которых можно отметить следующие:

1. ОКР «Разработка технологии создания высокопроизводительных модульно–наращиваемых многопроцессорных вычислительных систем с программируемой архитектурой на основе реконфигурируемой элементной базы» (НИИ МВС ТРТУ, г/к № 02.447.11.1007 от 06 июля 2005 г., шифр “Медведь”, № гос. рег. 0122.0510630, 2005–2006).
2. ОКР «Создание семейства высокопроизводительных многопроцессорных вычислительных систем с динамически перестраиваемой архитектурой на основе реконфигурируемой элементной базы и их математического обеспечения для решения вычислительно трудоемких задач» (НИИ МВС ЮФУ, шифр “Большая Медведица”, № ГР 01.2007 05707, 2007–2008).
3. НИР «Разработка теоретических основ построения гетерогенных многопроцессорных вычислительных систем с программируемой архитектурой» (НИИ МВС ЮФУ, шифр “Карнавал”, № ГР 01.2007 02728, 2007–2010).
4. ОКР «Разработка реконфигурируемой вычислительной системы РВС–7 и организация на ее основе производства реконфигурируемых вычислительных систем с производительностью до 10^{15} операций в секунду в одностоечном конструктиве 47U» (НИИ МВС ЮФУ, шифр 2011–2.7–527–008–001 «Плеяда», № ГР И111102120521, 2011–2013).
5. НИР «Разработка и исследование технологии создания ресурсонезависимого прикладного программного обеспечения высокопроизводительных вычислительных систем гибридного типа» (НИИ МВС ЮФУ, шифр «Русалка», № ГР 114101540025, 2014–2016).
6. НИОКР «Разработка нового поколения программного обеспечения РВС для решения ресурсоемких научно–технических задач различных предметных областей», (НИЦ СЭ и НК, шифр «Поколение», 2016–2023).

Апробация результатов работы. Основные результаты диссертационной работы были представлены на ряде международных и всероссийских тематических конференций, среди которых можно выделить следующие:

в России: серия международных научных школ «Высокопроизводительные вычислительные системы» (ВПВС), 2004, 2005, 2007, 2011 гг.; серия всероссийских научных конференций «Научный сервис в сети Интернет», г. Новороссийск, 2006–2014 гг; серия Международных научных конференций «Параллельные вычислительные технологии» (ПАВТ), 2011–2021 гг.; серия международных научно–практических конференций «Искусственный Интеллект. Интеллектуальные и многопроцессорные системы» (ИИ), 2004, 2006, 2008, 2010 гг.; международные научно–технические конференции «Многопроцессорные вычислительные и управляющие системы» (МВУС), 2007 г., 2009 г.; серия международных научных конференций «Суперкомпьютерные системы и их применение» (SSA), 2008, 2010, 2012 гг.; международные научно–технические конференции «Суперкомпьютерные технологии» (СКТ), 2010, 2012, 2014, 2016, 2018, 2020 гг.; международные конференции «Parallel Computing Technologies»

(PaCT), 2011, 2015, 2019, 2021 гг.; международные конференции «Суперкомпьютерные дни в России», 2015–2021 гг.; всероссийские мультиконференции по проблемам управления (МКПУ), 2013, 2015, 2017, 2019, 2021 гг.; национальные Суперкомпьютерные Форумы (НСКФ), 2012–2019 гг.,

за рубежом: научно–практическая конференция молодых ученых и специалистов «Технологии высокопроизводительных вычислений и компьютерного моделирования», University of Amsterdam, г. Амстердам, Нидерланды, 2012 г.; Science and Information Conference (SAI), 2013, United Kingdom; IFAC International Conference on Programmable Devices and Embedded Systems (PDES), 2013, 2016, Czech Republic; 18th International Conference on Circuits, Systems, Communications and Computers (CSCC 2014), 2014, Greece; 6th European Conference of Computer Science (ECCS '15), 2015, Italy; 5th International Conference on Informatics, Electronics & Vision (ICIEV), 2016, Bangladesh; 5th International Young Scientist Conference on Computational Science «Procedia Computer Science», 2016, Poland.

Публикации. По теме работы опубликовано более 170 работ, из которых 29 индексируются в международных базах наукометрических данных Web of Science и Scopus, 36 работ из списка ВАК РФ, получено 15 свидетельств об официальной регистрации программ для ЭВМ.

Структура и краткое содержание диссертации.

Диссертация состоит из введения, пяти глав, в которых изложены решения поставленных частных научных задач, заключения, библиографического списка и приложений. Содержание работы изложено на 348 страницах, работа содержит 87 рисунков и 17 таблиц, список использованной литературы из 248 наименований.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Во введении обоснована актуальность работы, дана ее общая характеристика, сформулированы цель и задачи исследования.

В первой главе рассмотрены основные компоненты архитектуры современных высокопроизводительных вычислительных систем: процессоры, графические ускорители, специализированные вычислители и ПЛИС, проанализированы тенденции их развития и технологии программирования. Существенное замедление роста производительности процессоров фон–неймановской архитектуры, по праву считающихся основными узлами многопроцессорных вычислительных систем, привело к заметной стагнации высокопроизводительных вычислений. Наглядными индикаторами этого стали двукратное замедление закона Мура (рис.1) и остановка роста тактовых частот, а причинами – технологическая «стена» (достижение физических пределов миниатюризации транзисторов в микроэлектронике), оказавшая влияние на все вычислительные архитектуры, «стены» памяти, энергетики и параллелизма на уровне команд. Графические (GPU) и специализированные ускорители на основе процессорной архитектуры в плане организации вычислений эффективны для узкого класса задач и во многом унаследовали известные проблемы процессоров, поэтому не позволяют обеспечить требуемые показатели производительности для большинства прикладных областей. Близкий к линейному рост производительности достигается для вычислительных систем на основе ПЛИС, которые обеспечивают 100– и 1000–кратный выигрыш в скорости решения целого ряда как слабосвязанных, так и сильносвязанных задач благодаря адаптации своей архитектуры к структуре решаемой задачи. При этом успешно решаемые процессорами задачи управления – с большим числом вложенных

условных операторов, рекурсивных вызовов и структур с обратной связью – реализуются на ПЛИС неэффективно.

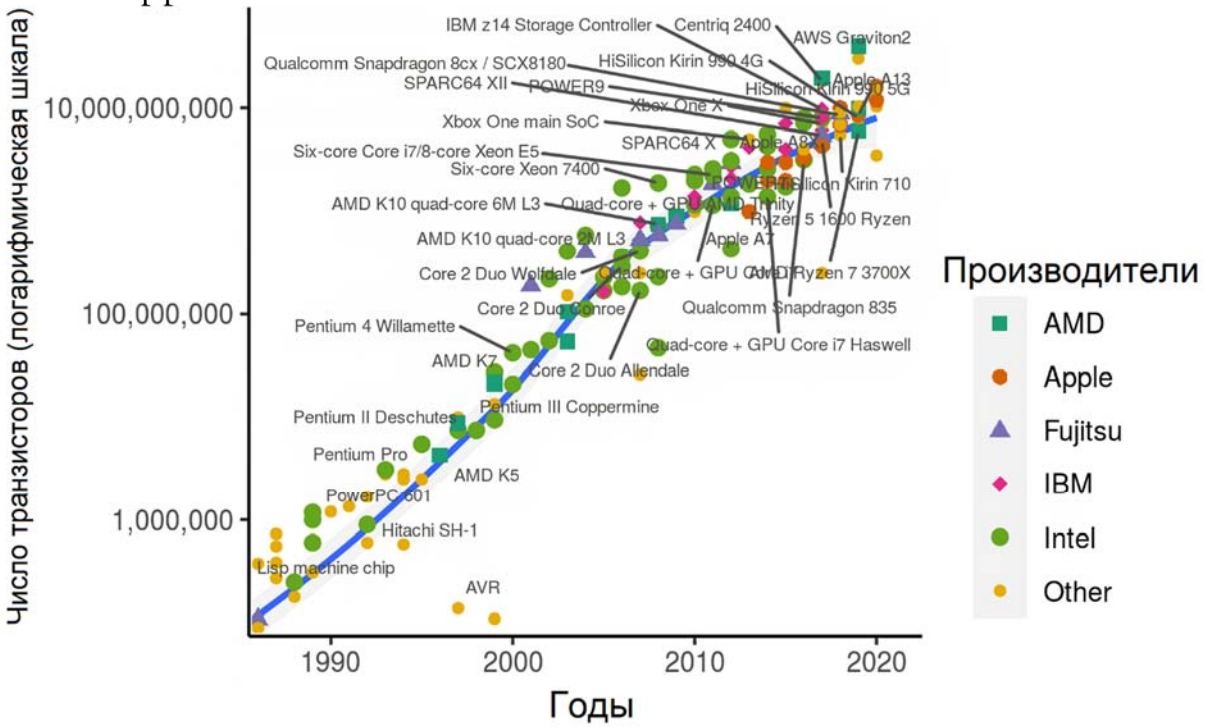


Рис. 1. Динамика роста числа транзисторов на кристалле в 1986-2020 гг.

Поскольку ни один из массовых вычислительных компонентов (процессоры, графические ускорители, ПЛИС) в полной мере не отвечает современным требованиям по эффективности реализации различных классов прикладных задач, наиболее перспективной высокопроизводительной вычислительной архитектурой по оценкам ведущих исследователей является гибридная (гетерогенная или конвергентная), содержащая ПЛИС, процессоры и графические ускорители. Однако широкое распространение гибридных вычислительных систем с узлами различных архитектур сдерживается отсутствием единой и универсальной технологии их программирования в общем вычислительном контуре.

В современных технологиях программирования процессорных узлов (MPI, OpenMP, НОРМА, DVM, PVM), графических ускорителей (CUDA, OpenACC и др.), ПЛИС (MitrionC, COLAMO) или гибридных вычислительных систем (OpenCL) архитектурная специфика присутствует уже на уровне моделей представления вычислений, поэтому задача реализуется только на заранее заданной конфигурации и фиксированной архитектуре вычислительной системы. Даже один из наиболее универсальных инструментов – технология OpenCL – требует от программиста непосредственно управлять перемещением данных, определять места хранения переменных в иерархии памяти и вручную реализовывать параллелизм в коде, что не позволяет обеспечить автоматическую реализацию фрагментов вычислений на нескольких ядрах (процессорах и/или графических ускорителях и/или ПЛИС).

Перечисленные технологии из–за существенных различий моделей представления вычислений не позволяют без участия программиста преобразовывать (портировать) вычисления прикладной задачи к другой конфигурации и/или архитектуре вычислительной системы, что исключает автоматическую портацию прикладных задач, тем самым *доказывая первое положение, выдвигаемое для защиты*. Преодолеть ограничения парадигм программирования различных архитектур при портации прикладных задач возможно с помощью нового подхода – создания единой формы

представления вычислений, которая может быть портирована на заранее неопределенное число вычислительных узлов для различных архитектур ГВС. Для этого аппаратный ресурс и архитектура вычислительных узлов должны быть изменяемыми и параметризуемыми компонентами трансляции модели представления вычислений в модель реализации. Для учета архитектуры и доступного аппаратного ресурса ГВС необходимы новые теоретические положения, принципы и методы преобразования единой формы представления вычислений, которые обеспечат совмещение и плавный переход от процедурной организации вычислений, характерной для процессоров и графических ускорителей, к свойственной ПЛИС структурно–процедурной, со сбалансированной вычислительной структурой.

Для представления вычислений часто используются графы: например, граф управления, управляющий граф, сеть Петри, граф алгоритма, информационный граф и др. Для параллельно–конвейерных архитектур обычно используются параллельные формы графа алгоритма, а для ПЛИС – информационный граф – конечный ориентированный ациклический граф, вершинами которого являются операции над данными, а дуги отражают информационную зависимость между ними. Информационный граф описывает максимально–параллельную форму задачи, в которой число входных и операционных вершин соответствует размерности всех обрабатываемых данных задачи. В отличие от графа алгоритма и других графовых моделей, информационный граф задачи (ИГЗ) (рис.2) позволяет иерархически описывать параллельные вычисления с помощью связанных информационными зависимостями подграфов из нескольких арифметических, условных или коммутационных операций. Операционные подграфы ИГЗ распределены по слоям F_{ij} и итерациям Φ_i: слои S_i содержат информационно–независимые подграфы g_{ij} , поэтому связи между ними в слое отсутствуют, а итерации описывают информационную зависимость подграфов разных слоев между собой. Характеристики аппаратной реализации (латентность выполняемых операций, длительность такта обработки, производительность и др.) в ИГЗ не учитываются, что обеспечивает инвариантность формы вычислений для рассматриваемых архитектур процессоров, графических ускорителей и ПЛИС.

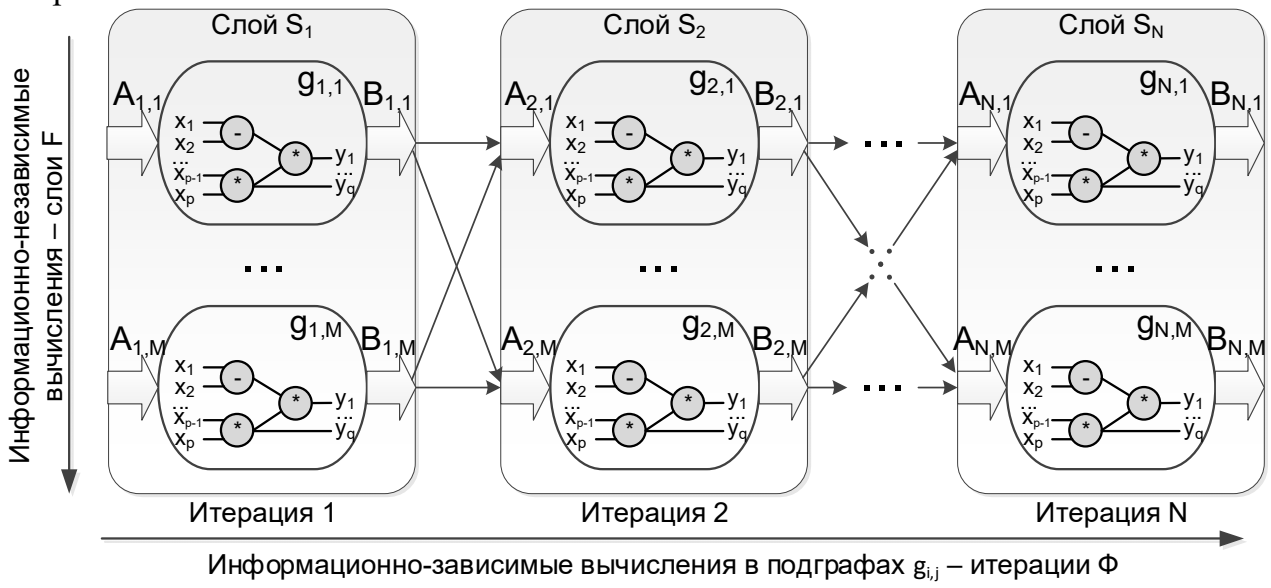


Рис. 2. Пример функционально–регулярного ИГЗ

Реализация ИГЗ в виде вычислительной структуры в силу максимального параллелизма требует множество вычислительных устройств и каналов памяти для одновременной подачи всех заданных параметрами задачи входных данных и

выполнения всех операций задачи, что обычно недостижимо для реальных вычислительных систем. Поэтому ИГЗ преобразуется в кадровую структуру – аппаратно (структурно) реализуемую на доступном ресурсе реконфигурируемой вычислительной системы (РВС) вычислительную структуру Q и правила организации входных ($A^i = x^i(t)$) и выходных ($B^j = y^j(t)$) потоков для нее (рис.3). Кадровая структура позволяет описывать реализацию вычислений для всех целевых архитектур – максимально–параллельные вычисления ИГЗ, параллельно–конвейерные вычисления, реализуемые в ПЛИС и в графических ускорителях, и последовательные (процедурные) вычисления, характерные для процессоров.

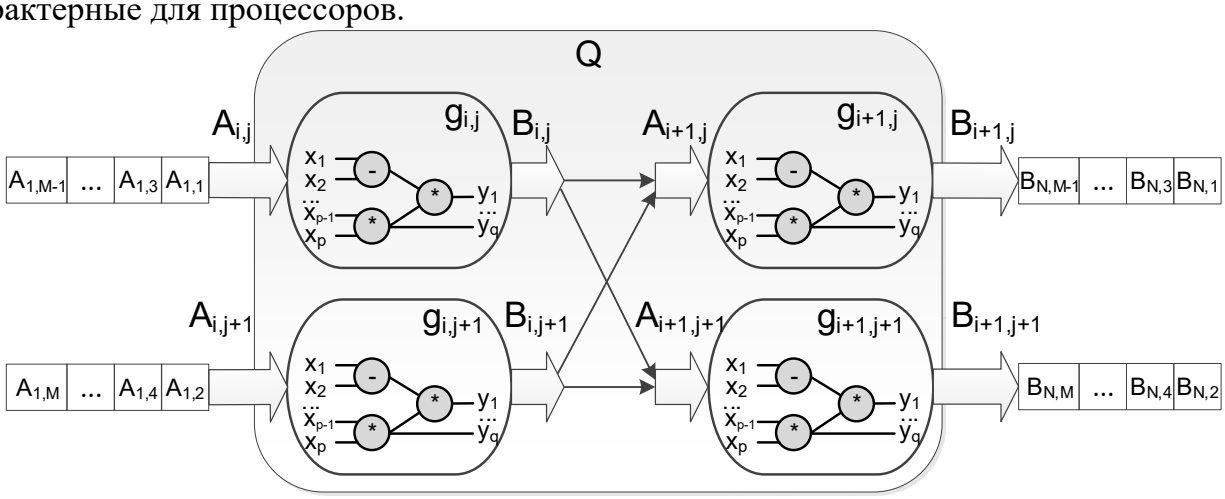


Рис. 3. Кадровая структура, реализующая вычисления ИГЗ на доступном аппаратном ресурсе

Для разграничения понятий архитектуры ГВС как вычислительной системы и вычислительной архитектуры ее компонентов (процессоров, графических ускорителей и ПЛИС) как принятого способа организации вычислений, предложен **термин** «целевая архитектура», означающий вычислительную архитектуру компонента гибридной вычислительной системы, что позволило сформулировать определение ресурсонезависимого программирования.

Определение 1. Параллельные программы, позволяющие изменять характеристики аппаратной реализации каждого фрагмента вычислений в зависимости от параметров целевой архитектуры ГВС, являются независимыми от архитектуры и конфигурации системы или *ресурсонезависимыми*. Ресурсонезависимость параллельных программ подразумевает инвариантность параллельных вычислений относительно доступного вычислительного ресурса.

Для ресурсонезависимого программирования гибридных вычислительных систем как единой конвергентной архитектуры предложены принципы:

1. Вычисления должны быть представлены в единой для различных целевых архитектур форме, обеспечивающей возможность реализации каждого фрагмента задачи на любой из целевых архитектур гибридной вычислительной системы.

2. Представленные в единой форме одновременные вычисления преобразуются к целевым архитектурам ГВС с учетом параметра – архитектуры и конфигурации аппаратного ресурса.

3. Методы преобразования вычислений к целевым архитектурам ГВС должны обеспечивать возможность изменения характеристик аппаратной реализации: способа организации вычислений, частоты, числа устройств и др. для каждого фрагмента задачи.

4. Выбор наиболее рационального варианта реализации прикладной задачи или ее фрагмента для каждой из целевых архитектур ГВС должен осуществляться с учетом

характеристик решения – латентности, интервала обработки данных, разрядности, числа устройств и т.д.

5. Методы преобразования вычислений должны учитывать информационную структуру прикладной задачи, в том числе для подзадач с разной степенью параллелизма, чтобы синтезируемое для параметрически заданного аппаратного ресурса решение было согласованно между вычислительными узлами разных целевых архитектур.

6. Методы преобразования вычислений должны обеспечивать сбалансированное рациональное решение как при увеличении, так и при сокращении доступного аппаратного ресурса гибридной вычислительной системы.

Сформулированные принципы открывают возможности разработки новых научных понятий и положений ресурсонезависимого программирования для описания единой модели вычислений, а также формализованных методов и алгоритмов ее портации – преобразования в модель реализации для гетерогенного вычислительного ресурса, содержащего вычислительные узлы на основе процессоров, графических ускорителей и ПЛИС.

Вторая глава описывает единую форму представления вычислений и правила их преобразования к целевым архитектурам гибридной вычислительной системы. В качестве единой формы представления параллельных вычислений для различных целевых архитектур предложен информационный граф задачи, позволяющий представлять информационные зависимости на уровне операций и подграфов задачи. Информационный граф задачи с точностью до топологических перестановок инвариантен к различным способам описания вычислений, что позволяет минимизировать число представлений параллельных вычислений, различающихся только порядком выполнения операций. Информационный граф не зависит от архитектуры и доступного вычислительного ресурса ГВС, поэтому позволяет представлять вычисления для любой ее целевой архитектуры, что соответствует первому принципу ресурсонезависимого программирования.

Для перехода от структурно–топологического представления вычислений и информационных зависимостей вершинами и дугами информационного графа к устройствам и коммутационной структуре целевых архитектур предложено преобразовать его в максимально параллельную кадровую структуру, которая учитывает характеристики реализации фрагмента вычислений (число слоев, итераций и устройств, латентность, интервал обработки данных, разрядность выполняемых операций и др.). Максимально параллельная кадровая структура (МПКС) как неразрывная совокупность вычислительной структуры, аппаратно–реализуемой на доступном ресурсе, и правил организации входных и выходных потоков данных синтезируется из функционально–регулярной формы информационного графа $G = \Phi_i(F_{ij}(g_{jk}^i))$ (см. рис.2), где F_{ij} – функция упорядочивания информационно–независимых подграфов в вычислительном слое $S_i, i = 1 \dots N$ задачи T, Φ_i – функция отображения информационно–зависимых вычислительных слоев $S_1, S_2, \dots, S_N$ для минимальных неделимых фрагментов вычислений, т.е. изоморфных подграфов g_{jk}^i.

Для учета характеристик аппаратной реализации на целевых архитектурах в представлении кадровой структуры предложена параметризация МПКС доступным аппаратным ресурсом вычислительной системы A_{RCS}: все компоненты задачи T с ИГЗ G могут быть представлены как композиция параллельной и последовательной составляющих, реализация которых определяется аппаратным ресурсом A_{RCS} вычислительной системы:

$$T(A_{RCS}) = \Phi_{\text{пар}} \overset{A_{RCS}}{\circlearrowleft} \Phi_{\text{пос}}(F_{\text{пар}} \overset{A_{RCS}}{\circlearrowleft} F_{\text{пос}}(g_{\text{стр}} \overset{A_{RCS}}{\circlearrowleft} g_{\text{проц}})) \quad (1),$$

где $\overset{A_{RCS}}{\circlearrowleft}$ – композиция параллельной и последовательной (процедурной) составляющих, зависящая от аппаратного ресурса A_{RCS} вычислительной системы;

$\Phi_{\text{пар}}$ – параллельная, т.е. структурная или пространственная, реализация информационно–зависимых подграфов g_{jk}^{i} кадровой структурой с конвейером, каждая ступень которого представляет реализацию подграфа g_{ijk}, в предельном случае – структурная реализация всех итераций максимально–параллельной кадровой структуры;

$\Phi_{\text{пос}}$ – последовательная (или процедурная), реализация информационно–зависимых подграфов g_{jk}^{i} кадровой структурой с конвейерными ступенями с обратной связью, в предельном случае – реализация всех информационно–зависимых подграфов g_{jk}^{i} (обработка всех итераций) одной ступенью;

$F_{\text{пар}}$ – параллельная, т.е. структурная или пространственная, реализация информационно–независимых подграфов g_{jk}^{i} в кадровой структуре, в предельном случае – структурная реализация всех подграфов слоя;

$F_{\text{пос}}$ – последовательная (или выполненная во времени), реализация информационно–независимых подграфов g_{jk}^{i} упорядочиванием данных, подаваемых на вход кадровой структуры, в предельном случае – обработка входных данных всех подграфов слоя одной реализацией g_{jk}^{i} с кратным увеличением потока обрабатываемых данных;

$g_{\text{стр}}$ – структурная реализация базового подграфа g_{ijk} как минимальной кадровой структуры (МинКС);

$g_{\text{проц}}$ – процедурная реализация базового подграфа g_{ijk} как минимальной кадровой структуры.

В отличие от ранее известных и принятых в теории структурно–процедурной организации вычислений форм описания кадровых структур, параметризуемое аппаратным ресурсом ресурсонезависимое представление (1), позволяет:

– явно параметризовать кадровую структуру значением доступного аппаратного ресурса, при подстановке которого она может меняться в диапазоне от минимальной до максимально параллельной;

– описывать задачи, минимальная кадровая структура $g_{\text{стр}}$ которых превышает доступный аппаратный ресурс A_{RCS};

– получать результат преобразования (1) при подстановке значения аппаратного ресурса, без промежуточных возвратов к МинКС, как в технологии индуктивных программ.

Подстановка значений архитектуры и конфигурации A_{RCS} в ресурсонезависимое представление задачи (1) определяет параллелизм всех компонентов кадровой структуры и занимаемый ею аппаратный ресурс ГВС, что *доказывает второе положение, выдвигаемое для защиты*.

Портация к целевым архитектурам ГВС выполняется переходом к информационно–эквивалентной структуре с меньшим параллелизмом и, соответственно, занимаемым аппаратным ресурсом. Параллелизм кадровой структуры можно представить в виде пространства возможных реализаций, измерениями которого являются параметры быстродействия (рис.4): слои L (информационно–независимые подграфы), итерации It (информационно–зависимые подграфы), команды C (устройства), разрядность ρ, интервал обработки данных I. Время решения T, соотнесенное с

аппаратным ресурсом кадровой структуры, характеризует ее эффективность. При портации максимально параллельной кадровой структуры в ней могут быть сокращены как количество слоев и итераций, так и параметры аппаратной реализации базового подграфа g_{jk}^{i} – разрядность обрабатываемых данных, число одновременно выполняемых операций (или работающих устройств) и интервал обработки данных.

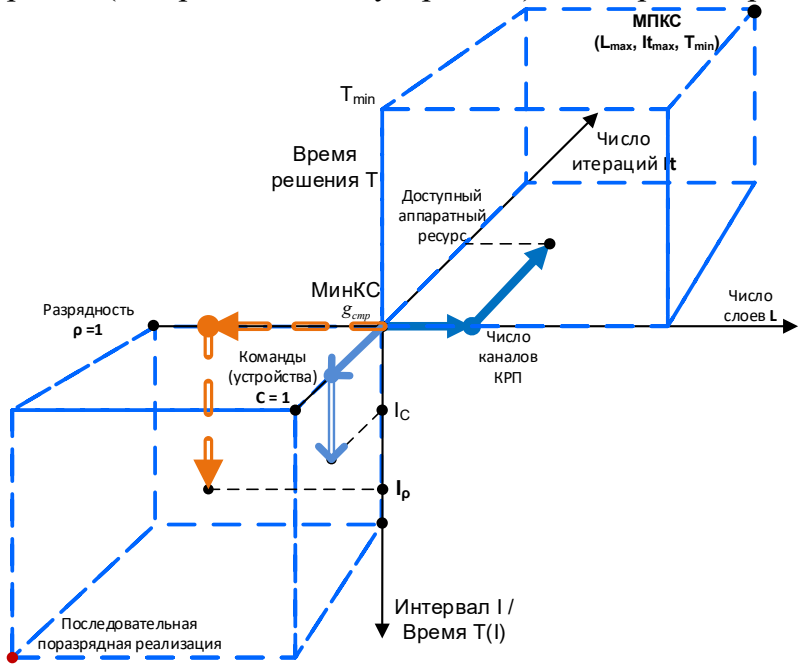


Рис. 4. Пространство возможных реализаций кадровых структур

Максимально параллельная кадровая структура, соответствующая информационному графу, на рис.4 представлена точкой МПКС с теоретически минимальным временем решения. В традиционных методах портации прикладные программисты переходили от МПКС к структурно реализуемой на доступном аппаратном ресурсе кадровой структуре: МинКС $g_{\text{стр}}$ или результату ее масштабирования по числу слоев (каналов КРП) и итераций (ресурсу), показанному стрелками в верхнем октанте на рис.4.

В отличие от известных методов структурно–процедурной организации вычислений предложенное представление (см. рис. 4) позволяет непрерывно описать все многообразие возможных кадровых структур с различной производительностью, а портацию представить как движение от максимально–параллельной кадровой структуры к структуре, реализуемой на параметрически заданном аппаратном ресурсе целевых архитектур. Более того, в отличие от известных методов индукции и редукции, предложенное представление позволяет описывать кадровые структуры, аппаратные затраты на реализацию которых превышают доступный ресурс, и так же непрерывно сокращать их степень параллелизма не только по слоям и итерациями, но и по разрядности и числу устройств в нижнем октанте пространства вплоть до точки минимального аппаратного ресурса – последовательной поразрядной реализации вычислений. Но даже эта точка не является теоретическим пределом портации, поскольку дальнейшее сокращение производительности кадровой структуры возможно увеличением интервала обработки данных при движении по оси *I*, что в ряде случаев позволяет согласовать вычисления с разными интервалами.

Движение в пространстве возможных реализаций может происходить и в сторону увеличения степени параллелизма кадровой структуры в любом из октантов пространства, что позволяет представлять индукцию вычислений при масштабировании вычислительного ресурса. Изменение степени параллелизма кадровой структуры в пространстве возможных реализаций представимо пространственной ломаной линией (рис.5), каждое звено которой отражает меняющуюся характеристику и/или аппаратные затраты на ее реализацию. Критерием остановки движения является достижение области доступного аппаратного ресурса вычислительной системы. Область доступного

аппаратного ресурса для каждой задачи задается абсолютными значениями характеристик параллелизма кадровой структуры и содержит все реализуемые на целевых архитектурах заданной ГВС кадровые структуры, различающиеся производительностью и временем решения задачи. Границами портации являются области минимального и максимального доступного аппаратного ресурса.

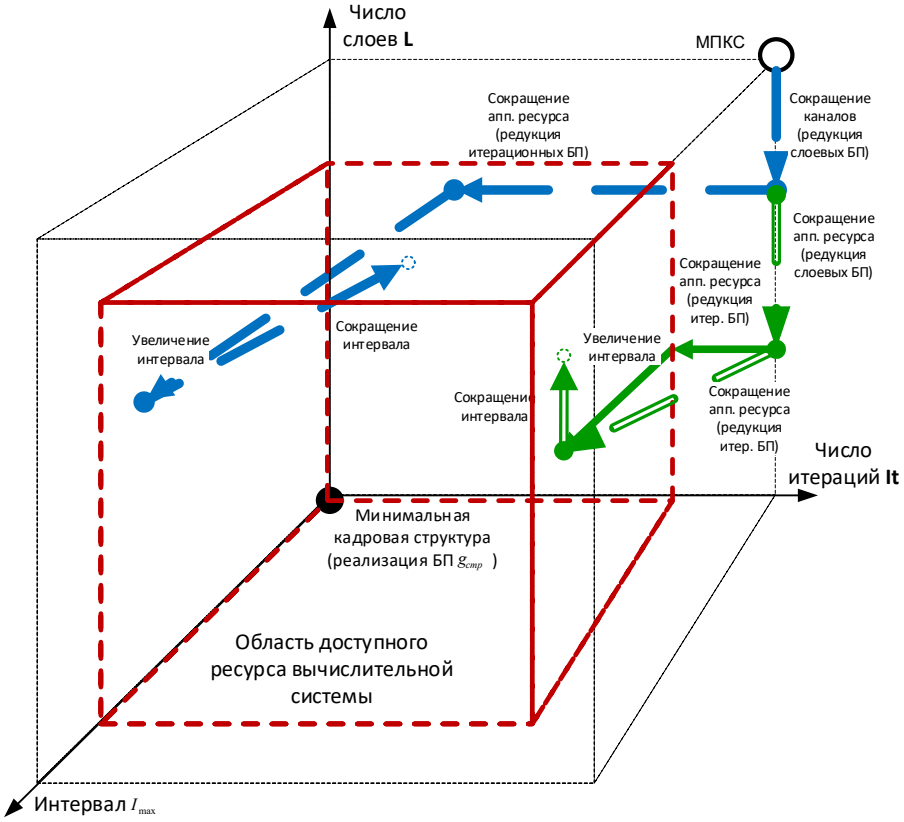


Рис. 5. Изменение параметров быстродействия кадровой структуры в пространстве возможных реализаций

Область максимального доступного ресурса ограничивает параллелизм кадровой структуры по числу каналов и аппаратному ресурсу сверху, а область минимального доступного ресурса позволяет завершить ее преобразование на уровне минимальной логической единицы гибридной вычислительной системы (например, одного процессора или кристалла ПЛИС), ниже которой сокращение аппаратных затрат непродуктивно.

Минимальный и максимальный доступные ресурсы позволяют определить критерий завершения портации кадровой структуры для доступного аппаратного ресурса заданной конфигурации целевых архитектур ГВС. После достижения области доступного ресурса из всех возможных вариантов необходимо выбрать рациональную по времени решения задачи кадровую структуру. В связи с этим портация прикладной задачи понимается как движение к *локальной* (достижение кадровой структурой области параметрически заданного доступного ресурса), а потом *глобальной* (поиск рациональной по времени решения задачи или заданному уровню производительности кадровой структуры в достигнутой области) целям. Глобальная цель не является оптимумом кадровой структуры, а представляет рациональный вариант организации параллельных вычислений для целевых архитектур гибридной вычислительной системы.

В соответствии с поставленной целью работы – минимизацией времени портации, для сокращения числа преобразований, выполняемых при достижении локальной и глобальной целей, сформулированы правила преобразования кадровой структуры:

– исходной точкой для преобразований является максимально параллельная кадровая структура;

– движение от максимально параллельной кадровой структуры начинается в направлении области доступного аппаратного ресурса сначала по оси «Число слоев», затем по оси «Число итераций» вплоть до минимальной кадровой структуры;

– критерием завершения движения является достижение кадровой структурой области доступного аппаратного ресурса, расположенной между максимальным и минимальным доступными ресурсами;

– если минимальная кадровая структура не входит в область доступного аппаратного ресурса, то движение продолжается в нижнем октанте в направлении последовательной поразрядной реализации по осям «Команды (устройства)» и «Разрядность»;

– при движении в пространстве возможных реализаций сначала достигается локальная цель, потом – глобальная.

Требуемый для реализации максимально параллельной кадровой структуры аппаратный ресурс, как правило, превышает доступный ресурс целевой архитектуры сразу по нескольким компонентам, поэтому для приближения к области доступного ресурса необходимо выбрать наиболее предпочтительное из возможных направлений движения. Для определения направления движения введено понятие *критического* ресурса как компонента реализации текущей кадровой структуры, доля которого в ее общих аппаратных затратах в наибольшей мере превышает доступный ресурс целевых архитектур. В пространстве реализаций критический ресурс соответствует наиболее удаленному от области доступного аппаратного ресурса компоненту целевой архитектуры ГВС. Достижение области доступного аппаратного ресурса как локальной цели портации выполняется минимизацией критического ресурса, т.е. сокращением (редукцией) параллелизма кадровой структуры с помощью числа слоев, итераций, команд/устройств и разрядности таким образом, чтобы связанные с ними аппаратные затраты кадровой структуры приближались к области доступного аппаратного ресурса. На каждом участке движения в пространстве возможных реализаций при минимизации критического ресурса кадровая структура приближается к области доступного аппаратного ресурса.

После сокращения критического ресурса на текущем интервале движения происходит его перерасчет, по результатам которого возможно изменение направления дальнейшего движения. Вершины ломаной линии, отражающей движение кадровой структуры в пространстве возможных реализаций, в которых меняются направление движения и/или критический ресурс, будем называть точками выбора альтернатив редукции. Характер изменения критического ресурса в точке выбора альтернатив определяется текущими значениями параметров быстродействия кадровой структуры, доступного аппаратного ресурса гибридной вычислительной системы, выбранного преобразования и коэффициента сокращения аппаратных затрат. Число точек выбора альтернатив определяет количество звеньев ломаной линии при движении кадровой структуры в пространстве возможных реализаций для достижения локальной и глобальной целей и количество преобразований кадровой структуры, а, следовательно, и время ее портации. Максимальное число точек выбора альтернатив зависит от суммарного числа компонент всех целевых архитектур ГВС, что определяет минимальное необходимое число преобразований кадровой структуры для достижения локальной цели портации – области доступного ресурса в пространстве возможных реализаций. Число компонент целевых архитектур сравнительно невелико: для ПЛИС оно определяется такими компонентами как число ячеек LUT, число ячеек MLUT, число регистров и триггеров Flip–Flop, число блоков внутренней памяти BRAM, число блоков аппаратной обработки DSP и число внешних связей кристалла ПЛИС; для графических ускорителей – числом АЛУ, объемом памяти и числом каналов; для процессоров – числом АЛУ и каналов. Таким образом, суммарное количество компонент для всех целевых архитектур

невелико и не превышает 20, что существенно меньше числа вариантов, анализируемого распараллеливающим компилятором для систем с распределенной памятью, который оценивается по теореме Кэли как n^{n-2}, где n – число вычислительных узлов системы. Для минимизации числа точек выбора альтернатив предложены критерии выбора предпочтительных направлений и параметров движения в пространстве возможных реализаций кадровых структур.

Сокращение степени параллелизма кадровой структуры при структурно–процедурной реализации задач на реконфигурируемых вычислительных системах выполнялось прикладным программистом с помощью известных *редукционных* преобразований: сокращения числа слоевых (информационно–независимых) и итерационных (информационно–зависимых) подграфов, числа устройств в аппаратной реализации, разрядности выполняемых операций, частоты и интервала обработки данных. Эти преобразования могут использоваться в качестве механизма достижения области доступного аппаратного ресурса при портации прикладных задач на целевые архитектуры ГВС.

В третьей главе описываются методы преобразования параметризуемой ресурсом кадровой структуры к целевым архитектурам гибридной вычислительной системы. Методы редукции производительности и аппаратных затрат применялись прикладными программистами при портации задач с заданными параметрами на заранее определенный вычислительный ресурс. Для произвольной и не фиксированной на этапе создания программы конфигурации целевых архитектур ГВС необходимо разработать и формализовать теоретические основы применения редукции производительности и аппаратных затрат в качестве механизма достижения области доступного аппаратного ресурса при портации.

Производительность максимально параллельной кадровой структуры при решении прикладной задачи P для данных различной разрядности на произвольной архитектуре и конфигурации вычислительной системы можно определить как отношение общего числа вычислительных операций NC_P к времени решения задачи t_P:

$$Perf_P = \frac{NC_P}{t_P}. \quad (2)$$

Общее число вычислительных операций над двоичными разрядами NC_P максимально параллельной кадровой структуры с функционально–регулярным ИГЗ можно оценить как произведение числа слоевых N_F и итерационных N_Φ подграфов, числа устройств Op в базовом подграфе g_{ijk} и разрядности обрабатываемых данных ρ:

$$NC_P = N_F \cdot N_\Phi \cdot Op \cdot \rho. \quad (3)$$

Время решения задачи P для максимально параллельной кадровой структуры будет пропорционально латентности этой структуры lat и длительности такта обработки τ:

$$t_P = lat \cdot \tau. \quad (4)$$

Тогда производительность $Perf_P$ максимально параллельной кадровой структуры задачи P согласно (2)

$$Perf_P = \frac{N_\Phi \cdot N_F \cdot Op \cdot \rho \cdot Freq}{lat}, \quad (5)$$

где $Freq = \tau^{-1}$ – частота обработки данных.

Редукция производительности сокращает $Perf_P$ в целое число раз пропорционально значению целочисленного коэффициента редукции R

$$Perf_P(R) = \left\lceil \frac{N_\Phi \cdot N_F \cdot Op \cdot \rho \cdot Freq}{lat \cdot R} \right\rceil. \quad (6)$$

Редукция производительности заключается в сокращении числителя и знаменателя дроби (6) на общие множители параметров быстродействия кадровой структуры

$N_Ф, N_F, Op, \rho$ и коэффициента R и выполняется следующими *редукционными* преобразованиями, обеспечивающими сбалансированное сокращение степени параллелизма и информационную эквивалентность результатов вычислений:

– редукцией по числу подграфов R^N, сокращающей число реализуемых в кадровой структуре информационно–независимых (N_F) и информационно–зависимых ($N_Ф$) подграфов ИГЗ;

– редукцией по числу устройств R^{Op}, сокращающей число одновременно работающих устройств Op кадровой структуры совмещением операций в одном устройстве;

– редукцией по разрядности R^ρ, сокращающей число одновременно обрабатываемых разрядов за счет преобразования параллельной обработки разрядов каждого операнда в параллельно–последовательную;

– редукцией по интервалу R^I, увеличивающей интервал (скважность) обработки/подачи данных без изменения аппаратных затрат;

– редукцией по частоте R^{Freq}, сокращающей частоту работы кадровой структуры.

Редукция R^N по числу информационно-независимых подграфов N_F соответствует обратному преобразованию к широко используемому в HPC распараллеливанию по данным. Редукция R^N по числу информационно-зависимых подграфов $N_Ф$ сокращает число ступеней конвейерной структуры, добавляя обратную связь. Редукции по числу устройств R^{Op}, разрядности R^ρ, интервалу R^I и частоте R^{Freq} применялись как частные эвристики при решении задач на РВС, для целевых архитектур графических ускорителей и процессоров они не использовались.

Для редукций по числу устройств R^{Op} и разрядности R^ρ предложена новая форма организации вычислений – последовательность микро–кадров (м–кадров), каждый из которых выполняет вычисления с меньшей степенью параллелизма на ограниченном доступном аппаратном ресурсе целевых архитектур (рис. 6). Основное отличие м–кадра MCS_2^{Op} (рис. 6–в) от кадровой структуры CS (рис. 6-а) – совмещение нескольких операций в одном вычислительном устройстве, приводящее к увеличению интервала обработки данных, но обеспечивающее единый и сбалансированный темп обработки данных без использования дополнительной памяти для хранения промежуточных результатов обработки входного потока данных при структурно–процедурной реализации CS двумя сокращенными кадровыми структурами $CS/2$ (рис. 6-б).

а) структурная реализация

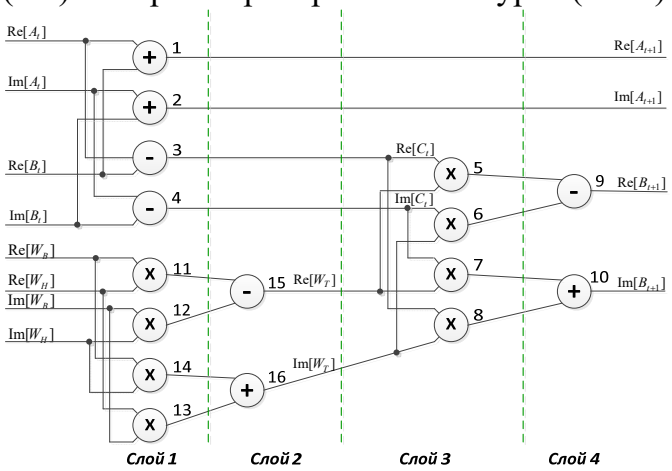
в) реализация в виде м–кадра

б) структурно–процедурная реализация

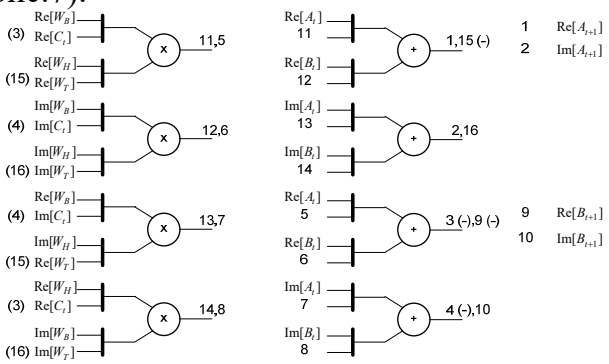
Рис. 6. Способы реализации кадровой структуры CS

При совмещении нескольких операций в одном вычислительном устройстве существует большое количество возможных вариантов формирования м-кадров, зависящих как от прикладной области, так и от конкретной решаемой задачи. В диссертации рассмотрено 5 различных *обобщенных* вариантов формирования м-кадров,

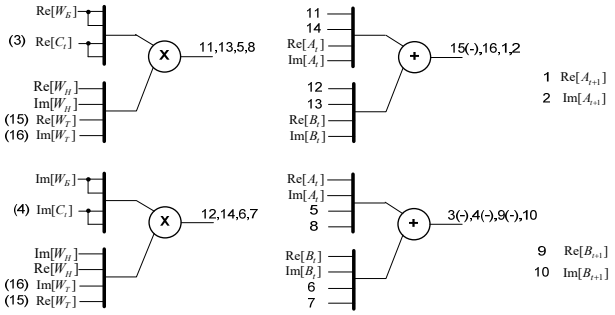
которые, по мнению автора, являются базовыми и применимыми для различных задач. С учетом специфики других предметных областей возможны новые способы синтеза м–кадров на основе предложенной в диссертации методики. Обобщенные варианты формирования м–кадров проиллюстрированы на примере реализации базовой операции (БО) быстрого преобразования Фурье (БПФ) (рис.7).



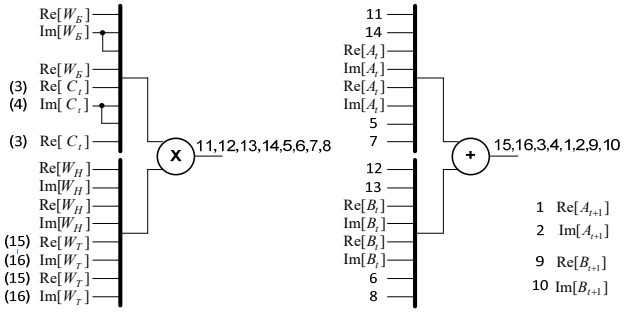
а) ИГЗ БО БПФ



б) с коэффициентом $R^{Op} = 2$ (м–кадр μ_2, μ_3, μ_4)



в) с коэффициентом $R^{Op} = 4$ (м–кадр μ_2)



г) с коэффициентом $R^{Op} = 8$ (м– кадр μ_1)

Рис. 7. Варианты м–кадров при редукции БО БПФ по числу устройств R^{Op}

На рис. 7 представлены:

1. М–кадр μ_1 (минимальный), в котором все операции каждого вида реализуются не более, чем на одном устройстве (для БО БПФ, содержащей две группы операций, м– кадр μ_1 состоит из двух устройств – умножителя и сумматора (рис. 7–г)).

2. М–кадр μ_2 (кратный), в котором число устройств для всех групп операций сокращается в одинаковое целое число раз, равное коэффициенту редукции, что соответствует «вынесению общего множителя» для операций базового подграфа кадровой структуры. Синтез м–кадра μ_2 определяется исходной кадровой структурой и не всегда возможен (например, если число операций базового подграфа не содержит общих делителей). Для БО БПФ м–кадр μ_2 возможен, причем сразу в трех вариантах: 8 устройств (4 умножителя и 4 сумматора – рис. 7–б), 4 устройства (2 умножителя и 2 сумматора – рис. 7–б) и 2 устройства (1 умножитель и 1 сумматор – рис. 7–в).

3. М–кадр μ_3, который синтезируется из максимального по числу операций яруса базового подграфа кадровой структуры, для соблюдения информационной эквивалентности дополненного устройствами, реализующими остальные операции кадровой структуры. Для рассматриваемого примера м–кадр μ_3 будет содержать 8 устройств из первого яруса (4 умножителя и 4 сумматора – рис. 7–б).

4. М–кадр μ_4, который содержит все устройства из наиболее разнородного по типу операций яруса, при необходимости дополненные устройствами для соблюдения

информационной эквивалентности. Для рассматриваемого примера м–кадр μ_4 содержит 8 устройств из первого яруса (4 умножителя и 4 сумматора – рис. 7–б) и совпадает с μ_3.

5. М–кадр μ_5 – это минимальный м–кадр μ_1 с одним дополнительным устройством, реализующим наиболее часто встречающуюся в базовом подграфе операцию. Для примера БО БПФ м–кадр μ_5 будет содержать 3 устройства. Здесь возможны два варианта: 2 умножителя и 1 сумматор или 1 умножитель и 2 сумматора.

Приведены теоретические оценки целесообразности формирования м–кадров и применения редукции по числу устройств R^{Op} в зависимости от значения коэффициента редукции, приведены абсолютные значения коэффициента редукции для ПЛИС Xilinx семейства Virtex UltraScale. На основе полученных оценок определен диапазон эффективности м–кадров для целевой архитектуры ПЛИС на примере семейства Xilinx Virtex UltraScale.

Для сокращения числа преобразований кадровой структуры при достижении локальной цели в пространстве возможных реализаций и минимизации времени портации редукционные преобразования целесообразно выполнять в комплексе, а не по отдельности. Для эффективного применения редукций сформулированы и доказаны теоремы о редукционных преобразованиях при портации кадровой структуры в целевые архитектуры ГВС.

Теорема 3.1 (о последовательном применении редукционных преобразований)

Последовательно примененные редукции R_m^T и R_n^T одного типа T с натуральными коэффициентами $m > 1$ и $n > 1$ эквивалентны редукции $R_{n \cdot m}^T$ того же типа с коэффициентом, равным их произведению $(m \cdot n) > 1$:

$$R_n^T \circ R_m^T = R_{n \cdot m}^T.$$

Следствие 3.1.1 из теоремы 3.1. Представление параметров кадровой структуры N_Φ, N_F, Op, ρ и коэффициента R в виде произведения простых сомножителей облегчает выбор наиболее рационального значения коэффициента для редукции каждого типа и сокращает число выполняемых редукционных преобразований и время портации задачи.

Следствие 3.1.2 из теоремы 3.1. Для коэффициента редукции производительности $R > 2$, являющегося простым числом, для сокращения производительности не менее, чем в R раз, целесообразна редукция в $(R + 1)$ раз.

Следствие 3.1.3 из теоремы 3.1. Редукция одного типа при кратном увеличении коэффициента редукции может быть последовательно применена к текущей кадровой структуре, без возврата к исходной максимально параллельной кадровой структуре. Если полученная в результате редукции кадровая структура нуждается в дополнительном кратном (не менее, чем вдвое) сокращении производительности и аппаратных затрат критического ресурса и редукция данного типа возможна, ее последовательное применение, без возврата к исходной кадровой структуре, сократит число преобразований для получения результата портации.

Следствие 3.1.4 из теоремы 3.1. Последовательное применение редукции того же типа не позволяет увеличить коэффициент редукции на 1. При последовательно примененных редукциях суммарный коэффициент равен произведению, а не алгебраической сумме коэффициентов редукций, поэтому при появлении накладных расходов на коммутацию и синхронизацию (например, при редукции по числу устройств и по разрядности) последовательное применение редукции не позволит увеличить коэффициент R до значения $(R + 1)$.

Теорема 3.2 (о неаддитивности последовательных редукционных преобразований)

*Для редуцированной с коэффициентом **n** кадровой структуры с помощью последовательного применения дополнительной редукции любого типа **T** с любым*

натуральным коэффициентом $k > 1$ *в общем случае невозможно получить структуру с коэффициентом редукции* $(n + x)$ *для любого наперед заданного* x, $\forall x \geq 1$, *некратного* $\boldsymbol{n}$*:*

$$R_n^{T1} \circ R_k^{T2} \neq R_{n+x}^{T}.$$

Следствие 3.2.1 из теоремы 3.2. Точный расчет коэффициента редукции сокращает число дополнительных редукционных преобразований и возвратов к исходной кадровой структуре и минимизирует время портации.

Теорема 3.3 (о коммутативности редукционных преобразований)

Суперпозиция редукций различных типов (T1 с коэффициентом $\boldsymbol{n}$ *и T2 с коэффициентом* $\boldsymbol{m}$*) коммутативна, т.е. изменение порядка выполнения редукций различных типов T1 и T2 в последовательных редукционных преобразованиях не изменяет редуцированную кадровую структуру.*

$$R_n^{T1} \circ R_m^{T2} = R_m^{T2} \circ R_n^{T1}.$$

Теорема 3.3 определяет инвариантность результата редукции от порядка применения редукционных преобразований разных типов к кадровой структуре.

Доказанные теоремы и полученные следствия являются теоретической основой для формализации приведенных ранее правил преобразования для метода портации кадровой структуры к целевым архитектурам ГВС. Метод преобразования кадровой структуры, параметризуемой аппаратным ресурсом, к архитектуре и конфигурации гибридной вычислительной системы состоит в применении редукции производительности для достижения локальной цели портации. Для сокращения времени портации необходимо минимизировать число выполняемых преобразований кадровой структуры. В наиболее общем случае для достижения области доступного ресурса может потребоваться редукционное преобразование каждого параметра быстродействия кадровой структуры. Поэтому количество параметров быстродействия кадровой структуры можно считать оценкой среднего числа преобразований метода портации, необходимых для достижения локальной цели. Для достижения глобальной цели портации может потребоваться одно из дополнительных оптимизационных преобразований: построение конвейера в конвейере, макроконвейера или автоподстановка. Общее число преобразований метода очень невелико и для рассматриваемых параметров быстродействия кадровой структуры N_{Φ}, N_F, Op, ρ для достижения глобальной цели в благоприятном случае потребуется порядка 5 преобразований, а в неблагоприятном, при возвратах к исходной кадровой структуре после каждой редукции, удвоится до 10. Полученная оценка количества редукционных преобразований, выполняемых при портации кадровой структуры в целевые архитектуры, существенно меньше числа вариантов, анализируемого распараллеливающим компилятором для систем с распределенной памятью, что *доказывает третье положение, выдвигаемое для защиты*.

В соответствии с поставленной научной проблемой одновременно с сокращением времени портации прикладной задачи необходимо обеспечить заданный уровень реальной производительности кадровой структуры. Для этого портация должна выполняться таким образом, чтобы обеспечить достижение как локальной цели (реализации на доступном ресурсе вычислительной системы), так и глобальной (заданного уровня производительности) с минимизацией числа выполняемых преобразований кадровой структуры. Поэтому для рационального применения методов преобразования кадровой структуры при заранее неопределенном вычислительном ресурсе гибридной вычислительной системы необходима методика портации. Методика портации определяет способы изменения параметров быстродействия кадровой структуры для всех рассматриваемых целевых архитектур ГВС таким образом, чтобы

каждое преобразование приближало ее к локальной или глобальной цели. Поэтому после каждого преобразования выполняется оценка аппаратных затрат и производительности кадровой структуры и проверка их соответствия заданным показателям, по результатам которой определяются значения следующего преобразования параметров быстродействия. Поскольку аппаратные затраты и производительность кадровой структуры в общем случае нелинейно зависят от ее параметров быстродействия для различных целевых архитектур ГВС, при изменении параметров быстродействия возможно не только их сокращение при редукции, но и увеличение ранее сокращенных (индукция) для достижения заданного уровня реальной производительности. Если преобразованная кадровая структура достигла области доступного аппаратного ресурса, но ее производительность ниже заданного уровня, следует найти более рациональную кадровую структуру в достигнутой области с реальной производительностью, не ниже заданной.

В методике преобразования кадровой структуры CS с аппаратными затратами $A_{CS} = (a_1^{CS}, a_2^{CS}, \dots, a_n^{CS})$ и параметрами быстродействия $P_{CS} = (p_1, p_2, \dots, p_n)$ к аппаратному ресурсу гибридной вычислительной системы $A_{HCS} = (a_1^{HCS}, a_2^{HCS}, \dots, a_n^{HCS})$ считается, что аппаратные затраты A_{CS} кадровой структуры нелинейно зависят от параметров быстродействия P_{CS}: $A_{CS} = \Psi(P_{CS})$.

Цель портации – определить параметры быстродействия P'_{CS}, которые будут являться решением системы

$$\begin{cases} A'_{CS} = \Psi(P'_{CS}) \le A_{HCS} - \text{локальная цель портации} \\ Perf(P'_{CS}) \ge Perf_з - \text{глобальная цель портации} \end{cases},$$

где $Perf(P'_{CS})$ – производительность кадровой структуры, а $Perf_з$ – заданный уровень реальной производительности.

Ниже приведена методика портации задач к рациональной реализации на заранее неопределенном вычислительном ресурсе гибридной вычислительной системы.

1. **Определение параметров редукции**

1.1. Определить критический ресурс и коэффициент сокращения аппаратных затрат $K_{cr} = max\left(\frac{a_i^{CS}}{a_i^{HCS}}\right) > 1$.

1.2. Определить коэффициент редукции $R = \lceil K_{cr} \rceil$.

1.3. Упорядочить параметры быстродействия p_i в кортеж $< P_{CS} >$ по степени влияния на критический ресурс.

1.4. Выбрать параметр быстродействия с максимальным влиянием на критический ресурс: $p_i : \psi(p_i) = max\ K_{cr}$.

2. **Определение эффективного шага редукции** R^*.

2.1. Для выбранного параметра быстродействия p_i определить один из возможных вариантов эффективного шага редукции R^*:

1) $R^* = R$;

2) $R^* = f(R, p_i), R^* < R$;

3) $R^* = \|p_i\|, R^* > R$.

3. **Редукция (сокращение параметров быстродействия) кадровой структуры**

3.1. Сократить параметр p_i с эффективным шагом R^*: $p'_i = \Theta(p_i, R^*)$ и сохранить его в кортеж: $< P'_{CS} > = < P_{CS} > \leftarrow p'_i$

4. **Оценка текущей кадровой структуры и выбор альтернатив**

4.1. Оценить аппаратные затраты и производительность кадровой структуры $A'_{CS} = \Psi(P'_{CS}),\ Perf(P'_{CS})$.

4.2. Оценить достижение целей портации из условий

$A'_{CS} = \Psi(P'_{CS}) \le A_{HCS}$ и $Perf(P'_{CS}) \ge Perf_з$.

4.3. **Если** оба условия выполняются, **то**

– перейти к п.6 для сохранения текущих параметров кадровой структуры

иначе: {Анализ альтернатив дальнейшей редукции:}

Если $A'_{CS} > A_{HCS}$, **то**:

Если $R^* = R$, **то**

– увеличить коэффициент редукции $R := R + \Delta$, где $\Delta = \left\lceil \frac{A'_{CS}}{A_{HCS}} - 1 \right\rceil$;

– восстановить исходное значение параметра быстродействия $< P'_{CS} > = < P_{CS} > \leftarrow p_i$;

– перейти к п.1.3 для редукции с увеличенным коэффициентом R.

Иначе

– перейти к п.1 для сокращения критического ресурса следующим параметром быстродействия p_{i+1}.

Если $A'_{CS} \le A_{HCS}$ **и** $Perf(P'_{CS}) < Perf_з$, **то**

– перейти к п.5 для уменьшения редукции ранее сокращенных параметров $p_1, \dots, p_{i-1}$.

5. **Индукция (увеличение ранее сокращенных параметров быстродействия) кадровой структуры**

5.1. Сформировать кортеж ранее сокращенных параметров $< p_{i-1}, \dots p_1 >$.

5.2. Выбрать очередной элемент кортежа p_k, определить для него коэффициент масштабирования по аппаратному ресурсу $K_M = min\left(\frac{A'_{CS}}{A_{HCS}}\right)$ и коэффициент индукции $Ind = \lfloor K_M \rfloor$.

5.3. Для выбранного параметра быстродействия p_k определить один из возможных вариантов эффективного шага индукции:

1) $Ind^* = Ind$;

2) $Ind^* = \varphi(Ind, p_k) < Ind$;

3) $Ind^* = \|p_k\|$.

5.4. Увеличить параметр p_k с выбранным шагом Ind^*: $p'_k = \Theta^{-1}(p_k, Ind^*)$ и сохранить его в кортеж: $< P'_{CS} > = < P'_{CS} > \leftarrow p'_k$

5.5. Перейти к п.4.

6. Сохранить кадровую структуру

6.1. Добавить текущие параметры кадровой структуры в список, упорядоченный по минимуму производительности.

6.2. **Если** $Perf(P'_{CS}) < Perf_з$ **и** <были другие критические ресурсы>, **то**

– выбрать следующий критический ресурс и перейти к п.1.1.

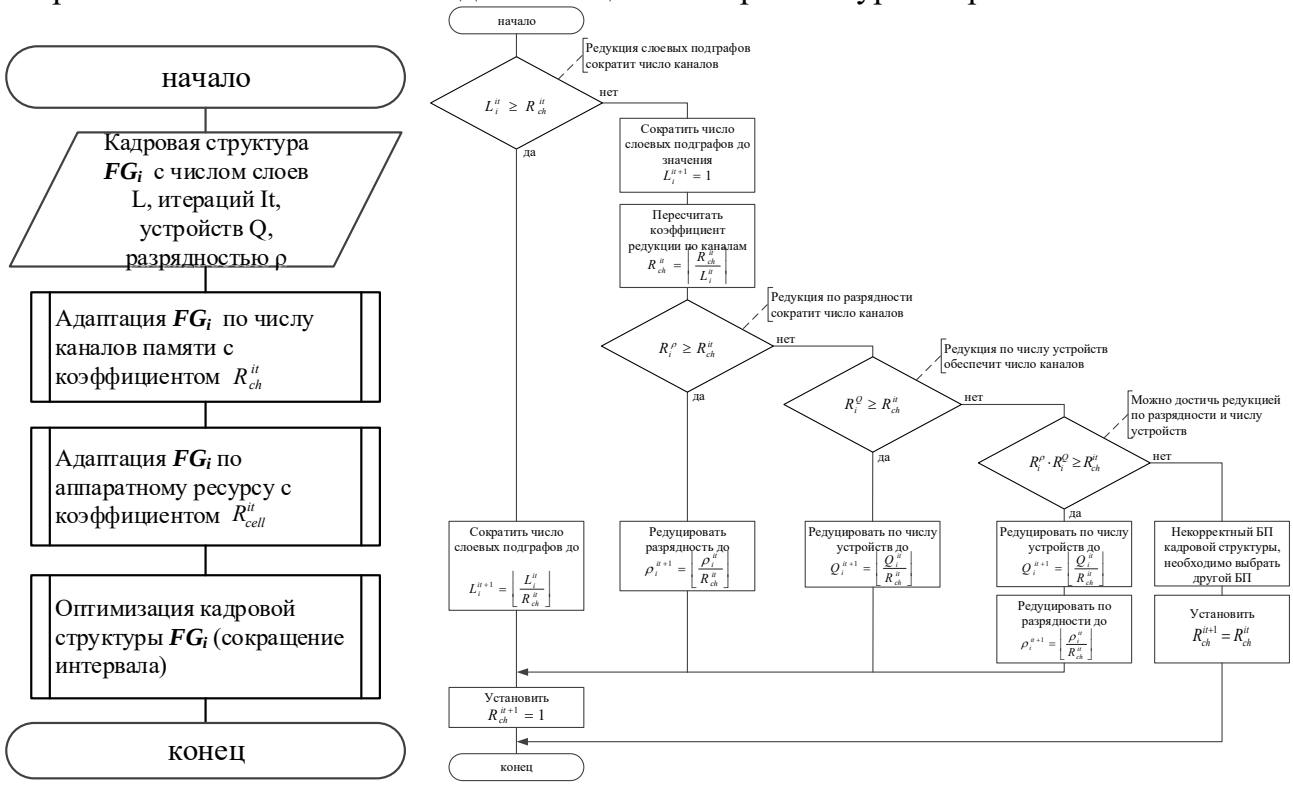
7. Выбор рационального варианта редукции кадровой структуры

7.1. Выдать первый элемент списка кадровых структур с наибольшей производительностью.

После расчета коэффициента редукции $R > 1$ в п. 1.2 он представляется в виде разложения на простые множители $R = \prod_{i=1}^{k} r_i$.Если коэффициент редукции R – простое число, то, с учетом следствия 3.1.2, разложение выполняется для значения $(R + 1)$. При определении эффективного шага редукции R^* в п.2.1 параметры быстродействия кадровой структуры $P_{CS} = (p_1, p_2, \dots, p_n)$, согласно теоремам 3.1 и 3.3, также принимают вид разложения на простые множители для удобства поиска корней диофантова неравенства $R_{p_1} \cdot R_{p_2} \cdot \ldots \cdot R_{p_n} \ge R$, где R_{p_i} – коэффициент сокращения параметра быстродействия. В ряде случаев эффективный шаг редукции приводит к предельному

сокращению одного или даже нескольких параметров быстродействия до значения $R^* = \|p_i\|, R^* > R$ (например, в случае перехода к процедурной обработке или при векторизации информационного графа).

Предложенная обобщенная методика преобразования кадровой структуры абстрагируется от параметров конкретной архитектуры вычислительной системы, поэтому может применяться для всех рассматриваемых архитектур и их комбинаций. Общее число анализируемых вариантов зависит от числа параметров быстродействия кадровой структуры, которые влияют на аппаратные затраты в целевой архитектуре. Для целевых архитектур процессоров и графических ускорителей число параметров минимально, т.к. они позволяют изменять только число каналов Ch и количество вычислительных узлов CPU/GPU. Набольшее количество варьируемых параметров у целевой архитектуры ПЛИС $A_{FPGA} = (Ch^{FPGA}, LUT, BRAM, HBM, DSP, FF)$, где Ch_{FPGA} – количество каналов, LUT – число ячеек **L**ook–**U**p **T**able в кристалле ПЛИС, $BRAM$ – количество блоков внутренней памяти **B**lock **RAM**, HBM – число блоков высокоскоростной памяти **H**igh **B**andwidth **M**emory, DSP – количество блоков цифровой обработки сигналов **D**igital **S**ignal **P**rocessor, FF – число регистров **F**lip–**F**lop. Поэтому в диссертации разработан ряд алгоритмов преобразования кадровой структуры для наиболее сложной среди рассматриваемых целевой архитектуры ПЛИС, обладающей наибольшим количеством параметров быстродействия $P_{CS} = \{L_i^{it}, It_i^{it}, Ch_i^{it}, Op_i^{it}, \rho_i^{it}, g_i\}$: L_i^{it} – число информационно–независимых подграфов в слое F_{ij}, It_i^{it} – число итераций в функции Φ_i, Ch_i^{it}– число внешних каналов кадровой структуры, Op_i^{it} – число устройств кадровой структуры FG^{it}, ρ_i^{it} – разрядность обрабатываемых данных в аппаратной реализации. Общее число подграфов $N_i^{it} = L_i^{it} \cdot It_i^{it}$ представляется произведением числа итерационных It_i^{it} и слоевых L_i^{it} подграфов. Обобщенный алгоритм преобразования кадровой структуры представлен на рис. 8, а алгоритм редукции по числу каналов, который может использоваться для всех целевых архитектур – на рис. 9.



а) обобщенный алгоритм б) редукция по числу каналов памяти

Рис. 8. Схема обобщенного алгоритма преобразования кадровой структуры

Редукция по числу каналов памяти сокращением количества слоевых подграфов одновременно сокращает и аппаратные затраты, поэтому для задач ряда предметных областей, например, символьной обработки, локальная цель портации $A'_{CS} = \Psi(P'_{CS}) \le A_{HCS}$ может быть достигнута уже после сокращения числа каналов. Если редукция по числу слоевых подграфов не обеспечила сокращения числа каналов до приемлемого уровня, т.е. $L_i^{it} < R_{ch}^{it}$, то число слоевых подграфов сокращается до минимума $L_i^{it+1} = 1$, коэффициент редукции по каналам представляется в виде $R_{ch}^{it} = L_i^{it} \cdot R_{\text{доп}}^{it}$, а дальнейшее сокращение числа каналов для коэффициента $R_{\text{доп}}^{it}$ выполняется с помощью редукций по разрядности и числу устройств. Достижение $R_{\text{доп}}^{it}$ редукцией по разрядности более предпочтительно, т.к. она пропорционально коэффициенту редукции линейно сокращает число каналов памяти и увеличивает интервал обработки данных, что позволяет достаточно точно оценить параметры результирующей кадровой структуры. Для целевых архитектур ПЛИС и графических ускорителей разработан алгоритм преобразования кадровой структуры к структурной (рис.9), а для процессоров – к процедурной (рис.10) формам организации вычислений.

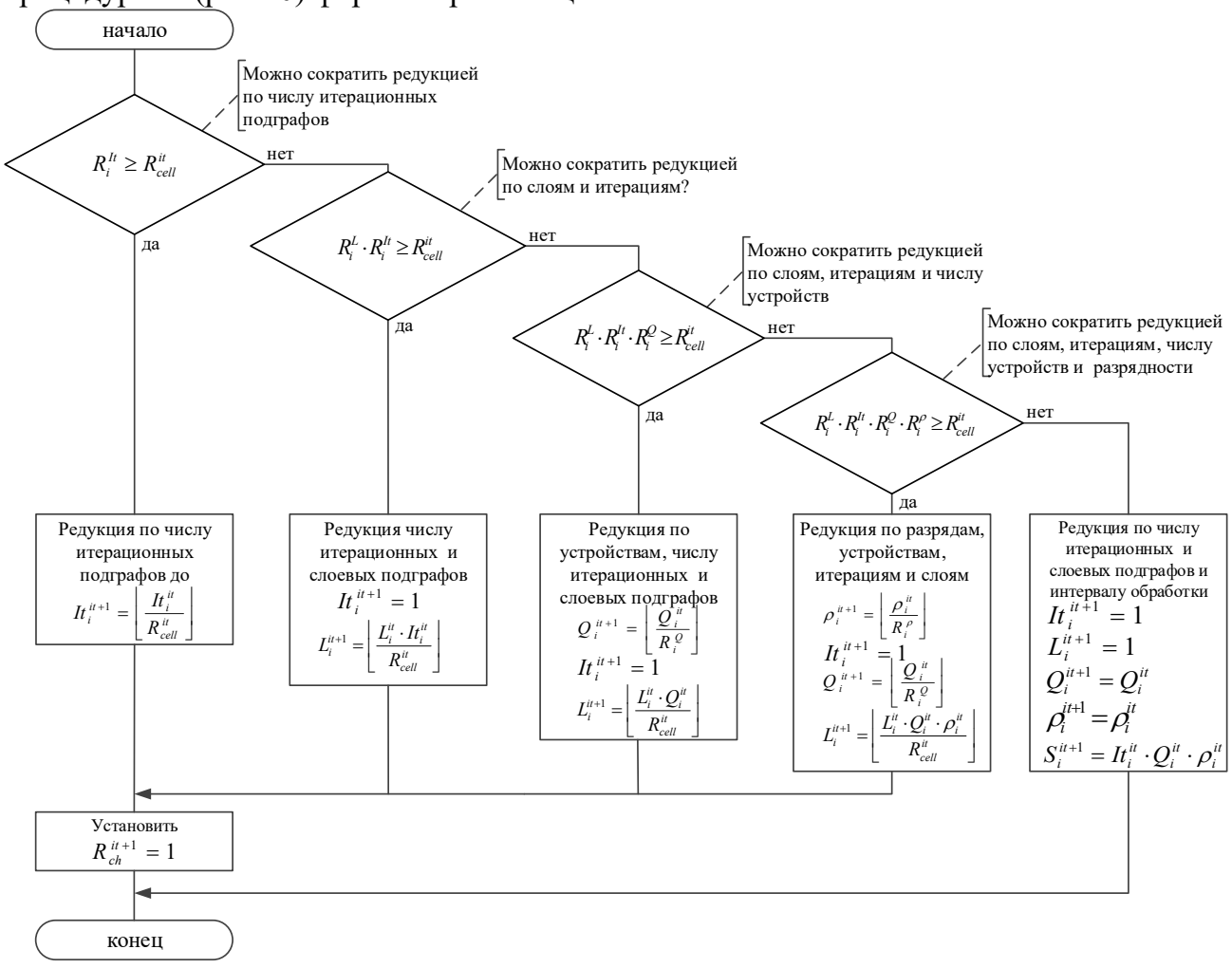


Рис. 9. Схема алгоритма преобразования кадровой структуры к структурной реализации

Первый этап алгоритма (см. рис. 9) – это сокращение итерационных подграфов до мультиконвейерной или конвейерной реализации $g_{\text{стр}}$ с возможностью дальнейшего построения «конвейера в конвейере», мультиконвейера или макроконвейера. Для процедурной реализации $g_{\text{проц}}$ (рис.10) целью преобразования кадровой структуры является оптимизация к макроконвейеру. Отличием этого алгоритма от предыдущих

является отсутствие редукции по числу устройств среди применяемых преобразований, поскольку $g_{\text{проц}}$ изначально подразумевает процедурную обработку

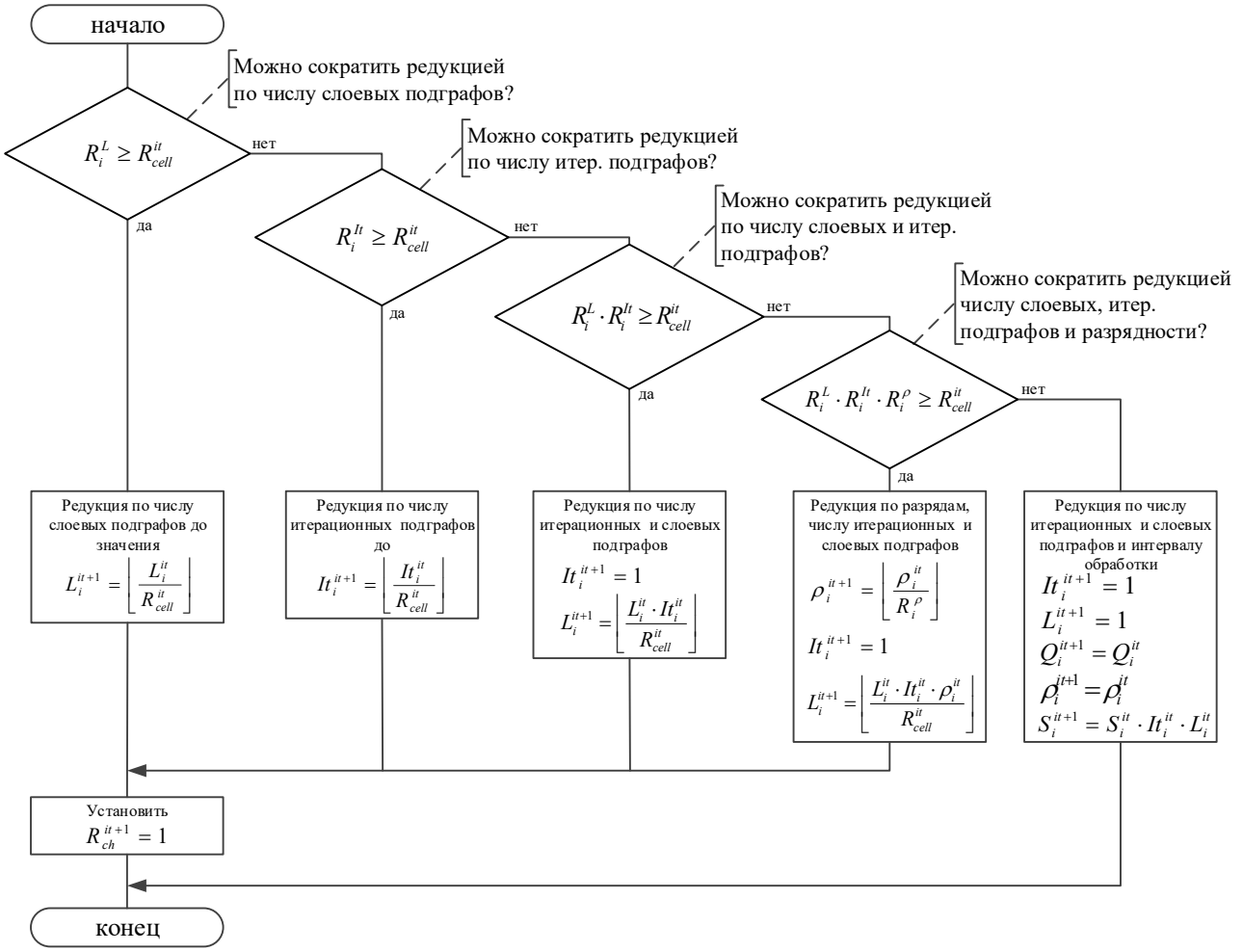


Рис. 10. Схема алгоритма преобразование кадровой структуры к процедурной реализации

В четвертой главе рассмотрены методы и алгоритмы преобразования прикладных задач, содержащих информационно–зависимые подзадачи с разной степенью параллелизма. Традиционная для структурно–процедурной организации вычислений реализация и портация таких задач состояла в процедурной смене кадровых структур, соответствующих подзадачам, с обменом данными через промежуточную память (см. рис. 6–б), что увеличивало как аппаратные затраты, так и время решения. Рациональная реализация информационно–зависимых кадровых структур, которая является глобальной целью методов портации, представляет единую согласованную конвейерную вычислительную структуру, работающую без разрывов потока данных памятью, своего рода «аналог» м–кадра (см. рис. 6–в). Особенность портации информационно–зависимых подзадач с разной степенью параллелизма к рациональному варианту состоит в том, что коэффициент редукции един для всех подзадач, но в каждой кадровой структуре для его достижения могут изменяться различные параметры быстродействия и использоваться разные редукционные преобразования. Поэтому степени параллелизма и интервалы обработки данных редуцированных структур могут различаться, что требует согласования и дополнительных аппаратных затрат для синхронизации потоков данных. Для портации задач с информационно–зависимыми подзадачами необходимы принципы согласованного преобразования кадровых структур с разной степенью параллелизма.

Информационно–зависимые подзадачи предложено представлять в виде кадровых *подструктур* – функциональных подграфов, каждый из которых, как и кадровая структура, описывается своим «базовым подграфом» ${g^s}_{ijk}$, числом слоев ${L^s}_i$ и итераций ${It^s}_i$, числом устройств ${Op^s}_i$, разрядностью ${\rho^s}_i$ и интервалом обработки данных ${I^s}_i$ ($s = 1..\boldsymbol{S}$, где $\boldsymbol{S}$ – число подзадач). Согласование параметров быстродействия и интервала обработки данных при редукции задачи с информационно–зависимыми кадровыми подструктурами формально представляется в виде решения системы из $\boldsymbol{S}$ диофантовых линейных неравенств, связывающих их параметры быстродействия с коэффициентом редукции:

$$\begin{cases} R_1^L \cdot R_1^{It} \cdot R_1^{Op} \cdot R_1^{\rho} \cdot R_1^I \geq R, \\ R_2^L \cdot R_2^{It} \cdot R_2^{Op} \cdot R_2^{\rho} \cdot R_2^I \geq R, \\ \quad\quad\quad\quad ... \\ R_s^L \cdot R_s^{It} \cdot R_s^{Op} \cdot R_s^{\rho} \cdot R_s^I \geq R. \end{cases} \quad (7),$$

где $R_1^L \ldots R_s^L$ – коэффициент редукции подзадач $1 \ldots s$ по числу слоевых подграфов,
$R_1^{It} \ldots R_s^{It}$ – коэффициент редукции подзадач $1 \ldots s$ по числу итерационных подграфов,
$R_1^{Op} \ldots R_s^{Op}$ – коэффициент редукции подзадач $1 \ldots s$ по числу устройств,
$R_1^{\rho} \ldots R_s^{\rho}$ – коэффициент редукции подзадач $1 \ldots s$ по разрядности,
$R_1^I \ldots R_s^I$ – коэффициент редукции подзадач $1 \ldots s$ по интервалу.

Каждое неравенство системы (7) задает локальную цель преобразования одной подзадачи – пропорциональное сокращение ее производительности и аппаратных затрат. Для максимально эффективной реализации информационно–зависимых кадровых структур необходимо найти такое решение системы, в котором степени параллелизма во всех кадровых структурах будут согласованы для обработки данных сбалансированной конвейерной структурой с минимально возможным интервалом. Поиск точного решения системы является трудноразрешимым уже при сравнительно небольшом числе подзадач, а с ростом количества неравенств точное решение (7) в целых числах не всегда существует, т.к. зависит от абсолютных значений параметров быстродействия кадровых подструктур и коэффициента редукции. Для синтеза сбалансированной конвейерной вычислительной структуры можно найти *рациональное* решение с единым, но не обязательно минимальным, интервалом обработки данных для всех согласованных по степени параллелизма кадровых структур.

Снизить сложность поиска рационального решения можно сокращением размерности решаемой системы. Для этого предложено одновременно согласовывать параметры двух неравенств системы, описывающих информационно–взаимозависимые (связанные непосредственной информационной зависимостью) кадровые структуры. Найденное для них решение в дальнейшем считается единой «подзадачей» с общими рассчитанными параметрами R_i^L, R_i^{It}, R_i^{Op}, R_i^{ρ}, R_i^I, с которой согласовываются параметры следующей информационно–взаимозависимой подзадачи и т.д. Согласование параметров кадровой подструктуры очередного неравенства с объединенной «подзадачей» будем называть операцией *присоединения* к промежуточному согласованному решению, в результате которого количество неравенств с неопределенными параметрами в (7) сокращается на 1. Это можно сформулировать как первый принцип согласованного преобразования информационно–зависимых кадровых структур: **на каждом этапе преобразования информационно–зависимых подзадач согласование параметров параллелизма и интервала обработки данных выполняется для двух информационно–взаимозависимых кадровых структур.** Присоединение

информационно–взаимозависимых кадровых структур к согласованному решению последовательно повторяется для всех подзадач в (7) и обеспечивает гарантированную сходимость преобразования информационно–зависимых подзадач.

Будем считать, что на каждом этапе решения системы рассматриваются два неравенства, соответствующие информационно–взаимозависимым кадровым структурам. Выбор первой пары кадровых структур для согласования очень важен, т.к. полученная в результате присоединения «подзадача» во многом определяет параметры всех последующих присоединяемых кадровых структур. Критерием выбора первой пары неравенств должен быть ее вклад в общую трудоемкость, критический ресурс и время решения задачи. Как правило, наибольший вклад во время решения задачи вносит подзадача с наибольшей вычислительной трудоемкостью и она же характеризуется наибольшим превышением аппаратных затрат по критическому ресурсу. Тогда основным критерием для первого неравенства системы (7) следует выбрать трудоемкость подзадачи, а второе неравенство определять на основе информационной взаимозависимости с учетом вычислительной трудоемкости и аппаратных затрат. Это позволяет сформулировать второй принцип согласованного преобразования информационно–зависимых кадровых структур: **для начального согласования параметров параллелизма и интервала обработки данных выбирается подзадача с наибольшей вычислительной трудоемкостью и максимальным превышением аппаратных затрат по критическому ресурсу.**

В соответствии с рассмотренными в главах 2 и 3 диссертации методами, для портации каждой информационно–зависимой кадровой структуры должна быть задана «под–область» доступного аппаратного ресурса из общего ресурса целевых архитектур гибридной вычислительной системы. Наиболее рациональным способом распределения доступного аппаратного ресурса между подзадачами является его выделение пропорционально вычислительной трудоемкости, когда наибольшая доля выделяется самой трудоемкой подзадаче, а менее вычислительно–трудоемкие, вклад которых в общее время решения незначителен, можно реализовывать на оставшемся ресурсе гибридной вычислительной системы, в том числе – активно используя процессорные вычислительные узлы. Поэтому третий принцип согласованного выбора параметров параллелизма и интервала обработки данных в информационно–зависимых кадровых структурах можно сформулировать следующим образом: **аппаратный ресурс гибридной вычислительной системы распределяется между информационно–зависимыми кадровыми структурами пропорционально их вкладу в общую вычислительную трудоемкость задачи**. Выделение аппаратных ресурсов должно быть гибким (динамическим), позволяя использовать не задействованные для преобразования одной кадровой структуры ресурсы при портации следующей.

На основе сформулированных принципов согласованного преобразования разработан метод и обобщенный алгоритм (рис.11) преобразования задач с информационно–зависимыми подзадачами с разной степенью параллелизма. На первом этапе метода для выбора первой пары неравенств оценивается вклад каждой подзадачи в общую вычислительную трудоемкость по числу устройств в максимально–параллельной кадровой структуре. Кадровую структуру с наибольшим числом устройств и/или занимаемым в максимально –параллельной форме аппаратным ресурсом предложено называть «флагманской» («флагманом» или «ф»), а структура с существенно меньшим (как минимум на 1 десятичный порядок) числом устройств или ресурсом считается «вспомогательной» и обозначается «катером» («к»). С помощью введенных обозначений описаны встречающиеся при реализации прикладных задач информационные

взаимозависимости подзадач, которые условно обозначены как: «флагман»–«катер» (ф–к), «флагман» и несколько «катеров» (ф–{к}), «флагман»–«катер»–«софлагман» (ф–к–соф) и «взаимодействующие равные» ({р}). «Софлагманом» обозначается сопоставимый с «флагманом» по выполняемому объему вычислений и занимаемому аппаратному ресурсу фрагмент вычислений, а «взаимодействующие равные» представляют собой информационную зависимость сопоставимых по объему вычислений подзадач с неизморфными базовыми подграфами. Приведенный перечень описывает наиболее распространенные информационные взаимозависимости подзадач, но не является исчерпывающим и может быть расширен.

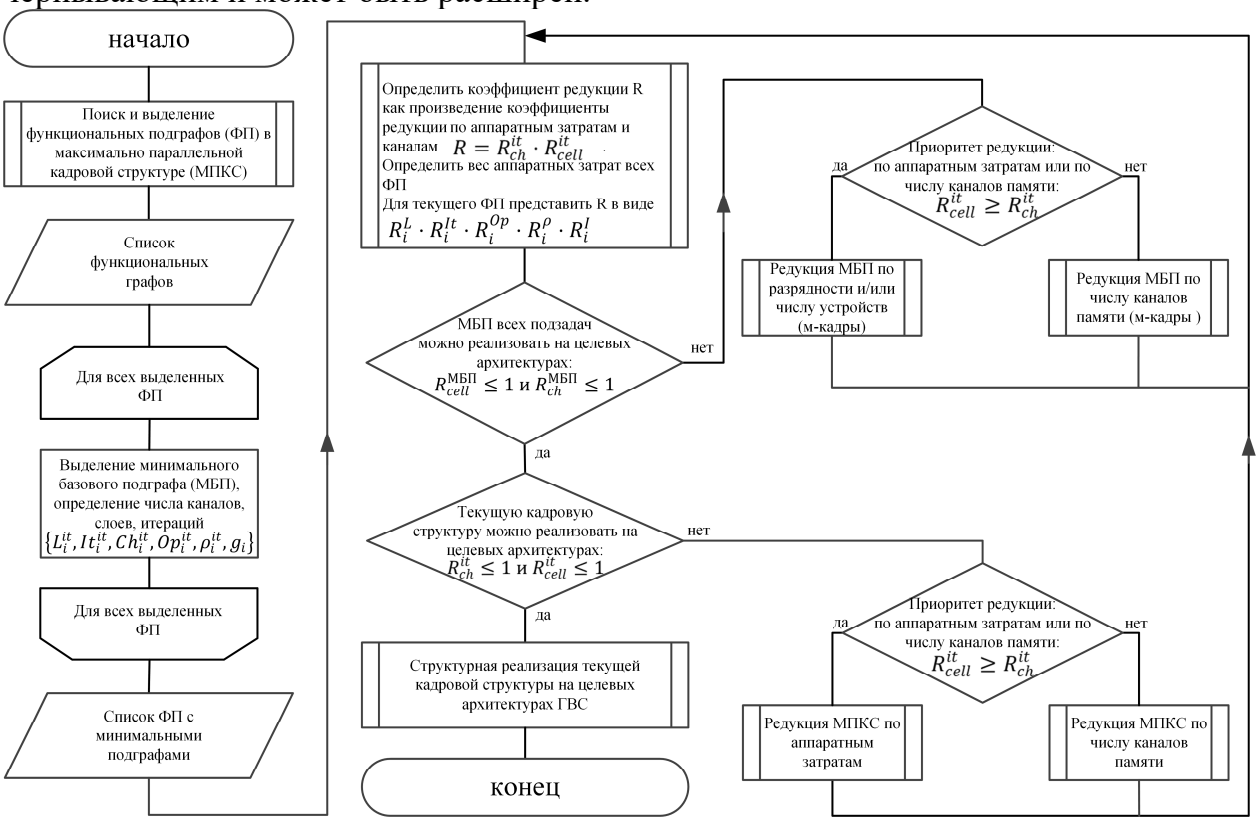


Рис. 11. Обобщенный алгоритм согласованного преобразования кадровых структур

Для информационной зависимости «флагман»–«катер» параметры кадровой подструктуры «флагмана» являются определяющими при поиске рационального решения системы диофантовых неравенств. Поэтому при портации задач с информационной зависимостью «флагман»–«катер» вначале осуществляется поиск решения для «флагмана» с помощью разработанных в третьей главе методики и алгоритмов (см.рис.8). Ряд найденных для «флагмана» параметров становятся определяющими для «катера»: они фиксируются и не меняются при портации, что сокращает число анализируемых при выполнении методики вариантов для «катера» и, конечно, сокращает общее время портации. Для задач с информационной зависимость «флагман»–«катер» при портации в целевую архитектуру ПЛИС рассмотрены возможные варианты согласованного преобразования кадровых подструктур «ф–к», разработан алгоритм портации и предложены структурные схемы блоков согласования.

Рассмотрен важный частный случай зависимости «ф»–«к»: информационная зависимость «флагмана» и множества взаимозависимых «катеров» – «ф–{к}». Главным отличием от «ф–к» является необходимость использования нескольких процессорных узлов для решения задачи. Определены критерии выбора «катеров» для операции присоединения при последовательном решении системы (7) и проанализированы возможные варианты решения. Для сокращения количества используемых процессорных

узлов сформулирована и доказана теорема об условиях бесконфликтного объединения нескольких процедурных реализаций «катеров» на одном процессорном узле ГВС:

Теорема 4.1. *Информационно–зависимые «катера» можно бесконфликтно процедурно реализовать на одном процессорном узле гибридной вычислительной системы, если выполняется условие:*

$$r_i \geq \sum_{j=i+1}^{w} Op_{k_j} \quad (8),$$

где r_i – длительность паузы до обработки следующего данного,
Op_{k_i} – число тактов обработки данных катером k_i,
w – количество объединяемых в единую процедуру «катеров», причем $w \leq \boldsymbol{S}$.

Использование универсальных процессоров для выполнения вспомогательных вычислений сбалансированно загружает вычислительный ресурс ГВС и освобождает аппаратный ресурс других целевых архитектур для возможной более эффективной реализации вычислительно–трудоемких фрагментов, таких как «флагман» и «софлагман».

Рассмотрены особые случаи информационной зависимости подзадач – «ф»–«к»–«соф» и «{р}» и специфичные для них варианты упорядочения системы (7), позволяющие упростить поиск согласованного и сбалансированного решения. Сформулированы и доказаны лемма и теорема о влиянии согласованности интервалов обработки данных в информационно–зависимых кадровых структурах на общее время решения задачи.

Лемма 4.1. *Несогласованность интервалов обработки данных в информационно–зависимых кадровых структурах «флагмана» и «софлагмана» увеличивает время решения задачи.*

Теорема 4.2. *Для рационального решения задач с информационной зависимостью «флагман»–«катер»–«софлагман» при портации на целевые архитектуры гибридной вычислительной системы необходимо обеспечить согласованный интервал обработки данных в кадровых структурах «флагмана» и «софлагмана».*

Отличием информационной зависимости «{р}» от всех предыдущих типов задач, где параметры «ф» во многом определяют параметры кадровых структур остальных подзадач, является возможность выбора любой кадровой подструктуры в качестве основы для операции присоединения, поэтому требуется анализ всех возможных вариантов выбора первого неравенства системы (7). Количество рассматриваемых при выборе рационального решения возможных вариантов можно оценить как число сочетаний $C_2^S = \frac{S \cdot (S-1)}{2}$. Количество вариантов анализа сравнительно невелико: для 5 подзадач необходимо оценить 10 различных вариантов согласования, для 10 подзадач – 45 вариантов. Автоматизация анализа вариантов с помощью программных средств позволит сократить время получения итогового решения.

В пятой главе представлены разработанные инструментальные программные средства технологии ресурсонезависимого программирования (ТРНП) и результаты портации прикладных задач с различными видами информационной зависимости на архитектуры и конфигурации реконфигурируемых и гибридных вычислительных систем.

Методы и алгоритмы преобразования кадровой структуры задач к доступному аппаратному ресурсу целевых архитектур гибридной вычислительной системы, разработанные в диссертации, реализованы в компонентах программного комплекса (рис.12): программе портации кадровой структуры на гибридные вычислительные системы различных архитектур и конфигураций («Прокруст») и программе преобразования фрагментов кадровой структуры на языке COLAMO («Щелкунчик»).

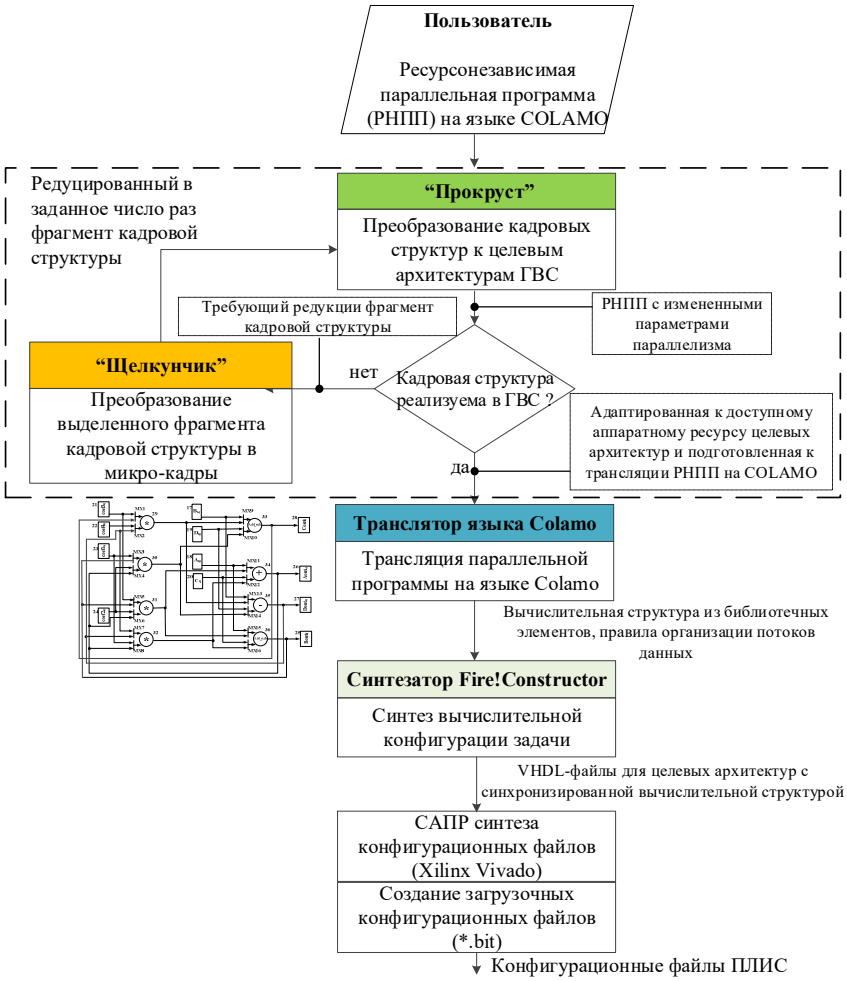


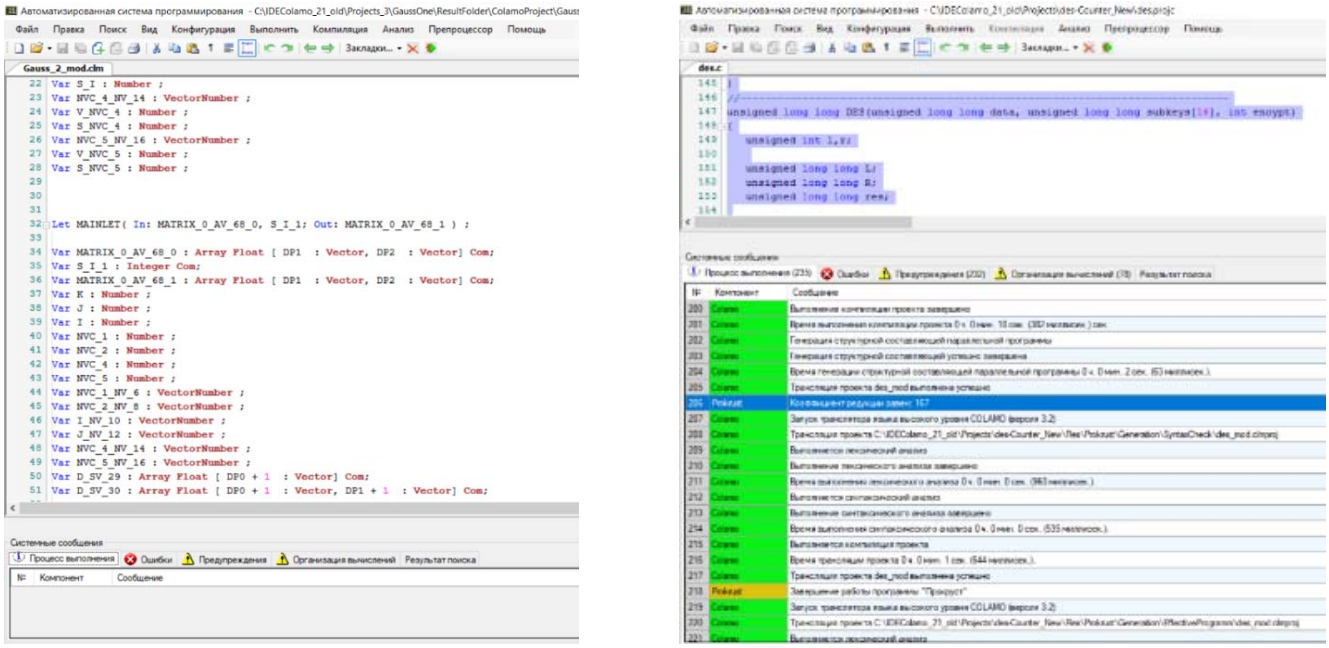
Рис. 12. Схема функционирования и взаимодействия программных средств технологии ресурсонезависимого программирования

Основным программным компонентом для преобразования ресурсонезависимой параллельной программы (РНПП) на языке COLAMO в параметризуемую доступным вычислительным ресурсом кадровую структуру является «Прокруст», который определяет характеристики каждой подзадачи: число слоев и итераций, число устройств, разрядность, интервал обработки данных и информационные зависимости между ними.

«Прокруст» также подсчитывает необходимые для реализации максимально–параллельной кадровой структуры аппаратные затраты, оценивает доступный аппаратный ресурс целевых архитектур гибридной вычислительной системы и рассчитывает коэффициент редукции производительности и аппаратных затрат по критическому ресурсу. После этого «Прокруст» анализирует и изменяет основные параметры параллелизма кадровых подструктур задачи в верхнем октанте пространства возможных реализаций – число слоев и итераций, интервал и интенсивность обработки данных в разных функциональных подграфах. При необходимости редукции минимального базового подграфа подзадачи «Прокруст» вызывает «Щелкунчик» и передает ему рассчитанный коэффициент редукции. «Щелкунчик» является вспомогательным компонентом, выполняющим преобразование параметров параллелизма кадровых подструктур задачи в нижнем октанте пространства возможных реализаций: разрядности и числа команд. «Щелкунчик» выполняет редукцию производительности переданного от «Прокруста» фрагмента кадровой структуры в заданное коэффициентом редукции число раз преобразованием в микро–кадры и возвращает ему результат. После этого «Прокруст» согласовывает интервал обработки данных во всех подзадачах, оптимизируя кадровую структуру с помощью преобразований «конвейер в конвейере», «макроконвейер» и «автоподстановка».

Взаимодействие программных средств ТРНП с пользователем осуществляется с помощью интегрированной среды разработки, общий вид основной экранной формы которой представлен на рис. 13. Запуск программ «Прокруст» и «Щелкунчик»

выполняется командами меню, расположенного в верхней части окна, а в нижней части окна расположена панель уведомлений (рис. 13–б) о состоянии процесса преобразования программы к целевым архитектурам вычислительных систем, предупреждениях и найденных ошибках.

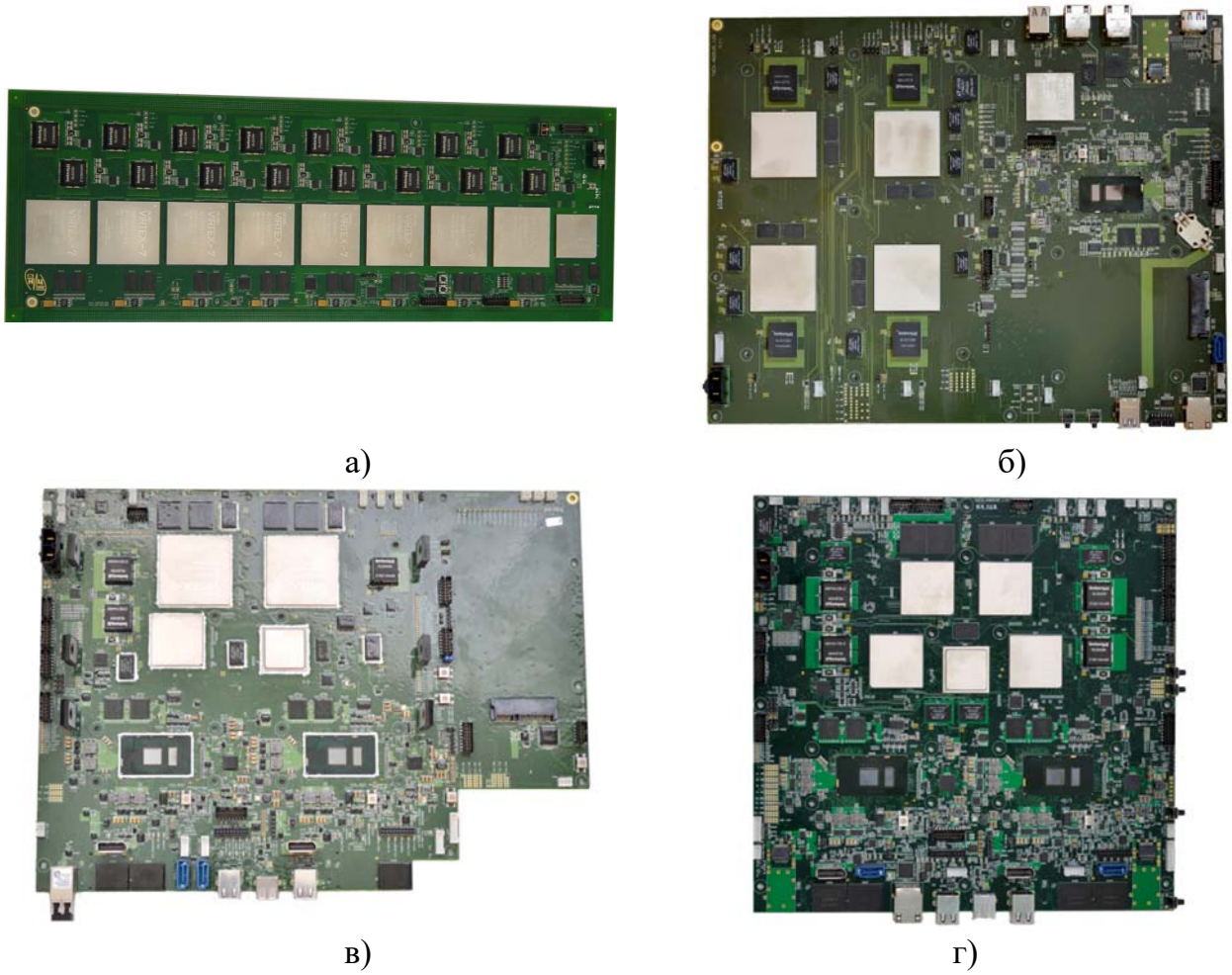


а) б)

– Рис. 13. Основная экранная форма интегрированной среды
а) с загруженной ресурсонезависимой параллельной программой;
б) панель уведомлений о состоянии портации

Переменные и их использование в циклах определяют слои, итерации и информационные зависимости подзадач и подграфов РНПП. Для изменения степени параллелизма ресурсонезависимой кадровой структуры используются целочисленные параметры параллелизма DP, влияющие на занимаемый аппаратный ресурс. Согласованное изменение параметров параллелизма DP при преобразовании к информационно–эквивалентной кадровой структуре не нарушает синтаксическую корректность ресурсонезависимой параллельной программы, т.к. параллельная составляющая с помощью формул пересчета автоматически преобразуется в потоковую. Итогом преобразований кадровых структур, выполненных программами «Прокруст» и «Щелкунчик», является программа на языке COLAMO, адаптированная к доступному аппаратному ресурсу целевых архитектур гибридной вычислительной системы. Транслятор языка программирования COLAMO и синтезатор многокристальных решений Fire!Constructor транслируют полученную программу в конфигурационные файлы целевой архитектуры ПЛИС и управляющие программы целевой процессорной архитектуры.

Экспериментальная проверка портации ресурсонезависимых параллельных программ с помощью разработанных средств проводилась на доступных автору вычислительных системах, содержащих ПЛИС фирмы Xilinx различных семейств, универсальные микропроцессоры и софт–процессоры, но разработанные методы и теоретические положения применимы ко всем рассмотренным целевым архитектурам различных гибридных вычислительных систем. Проверка автоматической портации прикладных задач компонентами комплекса, проводилась для РВС «Тайгета», «Терциус», «Терциус–3» и ГВС «Атлант–10» и «Атлант–Д», внешний вид печатных плат которых представлена на рис. 14.



а) б)

в) г)

Рис. 14. Аппаратные платформы для апробации методов портации прикладных задач
а) Тайгета; б) «Терциус»; в) «Атлант–Д»; г) «Атлант–10»

РВС «Тайгета», «Терциус» и «Терциус–3», содержащие вычислительное поле из нескольких кристаллов ПЛИС, являются одними из наиболее сложных вычислительных архитектур для портации прикладных задач существующими технологиями, такими как OpenCL, Vivado HLS и распараллеливающие компиляторы. РВС «Тайгета» производительностью 2,66 Тфлопс содержит четыре 20–слойных печатных платы с двухсторонним монтажом элементов, на каждой из которых расположены 8 ПЛИС XC7VX485T–1FFG1761 с 48,5 миллионов эквивалентных вентилей, 16 микросхем распределенной памяти SDRAM DDR2 общим объемом 2 Гбайт, интерфейсы LVDS и Ethernet и другие компоненты.

В настольном реконфигурируемом компьютере (НРК) «Терциус» и «Терциус–3» поле ПЛИС выполнено не на отдельной плате, а интегрировано с материнской платой с процессором Intel Core I5 6300U. В НРК «Терциус» производительностью 2,5 Тфлопс установлены 4 ПЛИС Xilinx Kintex UltraScale XCKU095 емкостью 100 млн. эквивалентных логических вентилей каждая, соединенные между собой в кольцо с помощью LVDS– и GTY/GTH–каналов, к каждой ПЛИС подключено по два модуля динамической памяти емкостью 1 Гбайт, поэтому общий объем памяти составляет 8 Гб. НРК «Терциус–3» производительностью 5,6 Тфлопс содержит вдвое большее количество кристаллов и другой тип ПЛИС – 8 ПЛИС Virtex XCVU095–1FFVB1760C.

ГВС «Атлант–Д» (см. рис. 14–в), содержит 2 ПЛИС Xilinx семейства Kintex–7 XC7K160T емкостью 162 тыс. логических ячеек и 2 специализированных «системы на кристалле» Xilinx семейства Zynq–7000 XC7Z030, установленных на печатной плате и объединенных в единое вычислительное поле.

ГВС «Атлант–10» (см. рис. 14–г) производительностью 7 ТФлопс содержит 4 ПЛИС Xilinx семейства Kintex–7 XC7K325T емкостью 326 тыс. логических ячеек, одну управляющую «систему на кристалле» Xilinx семейства Zynq–7000 XC7Z030 логической ёмкостью 125 тыс. логических ячеек с двухъядерным процессором ARM Cortex–A9 MPCore и SIMD–сопроцессором NEON, а также 2 процессора Intel Core i5–6300U, установленных на печатной плате и объединенных в единое вычислительное поле.

Апробация алгоритмов преобразования кадровых структур и проверка работоспособности инструментальных средств «Прокруст» и «Щелкунчик» выполнялись преобразованием ресурсонезависимых программ и портацией ряда прикладных задач в конфигурационные файлы для целевых архитектур действующих образцов ГВС. Для каждой аппаратной платформы ГВС измерялось время преобразования кадровой структуры и рассчитывалось время портации задачи как сумма времени преобразования и времени синтеза загрузочного модуля целевой архитектуры.

Время преобразования кадровой структуры разработанными средствами ТРНП сравнивалось с оценкой усредненного времени преобразования задачи прикладным программистом, которое для всех аппаратных платформ было принято равным двум 8–часовым рабочим дням или 57 600 с. Время синтеза загрузочного конфигурационного файла ПЛИС (проекта) считается равным для средств ТРНП и решений схемотехников, поскольку определяется рабочей частотой, логической емкостью и уровнем заполнения кристалла ПЛИС. Для всех портируемых прикладных задач рабочая частота составляла 250 МГц, а уровень заполнения кристалла составлял не менее 90%, поэтому время синтеза проекта определялось логической емкостью ПЛИС аппаратных платформ и составляло 3 часа для РВС «Тайгета», 7 часов для РВС «Терциус» и «Терциус–3», 1 час для ГВС «Атлант–10» и 3 часа для ГВС «Атлант–Д». Преобразование и портация задач выполнялись на персональном компьютере с процессором Intel (R) Core (TM) i7–8750H @2,2 ГГц и 16 Гб оперативной памяти, работающим под управлением операционной системы Windows 10 Pro.

Эффективность применения средств ТРНП определялась по двум основным критериям: выигрыш во времени (при преобразовании кадровой структуры и портации задачи) и достигнутый уровень реальной производительности. Выигрыш во времени преобразования и портации прикладной задачи рассчитывался как отношение времени, затраченного схемотехниками, к времени работы средств ТРНП. Оценка достигнутого уровня реальной производительности полученных решений осуществлялась аналогичным образом – как отношение количества подграфов в решениях, полученных средствами ТРНП, к числу подграфов в решении схемотехников. Для целевой архитектуры ПЛИС количество реализованных подграфов в параллельно–конвейерном решении позволяет достаточно точно оценить реальную производительность задачи.

Проверка портации проводилась для прикладных задач со всеми рассмотренными в 4 главе видами информационной зависимости: задач с единственным базовым подграфом, «флагман»–«катер», «флагман»–«множество катеров», «флагман»–«катер»–«софлагман» и «взаимодействующие равные» для РВС «Тайгета», «Терциус», «Терциус–3» и ГВС «Атлант–10» и «Атлант–Д». В качестве задач с единственным базовым подграфом портировались алгоритмы символьной обработки с информационно–независимыми подграфами и задачи линейной алгебры с прямой информационной (или итерационной) зависимостью изоморфных подграфов. В диссертации представлены результаты портации средствами ТРНП 3 симметричных блочных алгоритмов: DES, ГОСТ 28147–89, AES, 2 хеш–функций – MD5 и SHA–1, а также результаты портации РНПП решения 3–диагональных СЛАУ методом Якоби из области линейной алгебры. В

таблице 1 приведены результаты экспериментальной проверки средств ТРНП при портации наиболее характерных прикладных задач с единственным базовым подграфом на целевые архитектуры РВС и ГВС.

Таблица 1. Результаты портации задач с единственным базовым подграфом на целевые архитектуры РВС и ГВС

Задачи	**DES**	**MD5**	**SHA–1**	**Якоби**
РВС «Тайгета»				
Время преобразования (портации) РНПП, с	36,6 (10836,6)	138,8 (10938,8)	41,6 (10841,6)	2629,9 (13429,9)
Выигрыш при преобразовании (портации) РНПП	1573 (6,3)	414,8 (6,2)	1382,6 (6,3)	21,9 (5,1)
Уровень реальной производительности	**0,63**	**0,63**	**0,86**	**0,86**
РВС «Терциус»				
Время преобразования (портации) РНПП, с	38,5 (25238,5)	145,1 (25345,1)	41,1 (25241,1)	712,8 (25912,8)
Выигрыш при преобразовании (портации) РНПП	1492,9 (3,2)	396,90 (3,2)	1401,8 (3,2)	80,8 (3,2)
Уровень реальной производительности	**0,65**	**0,67**	**1,00**	**0,85**
ГВС «Атлант–10»				
Время преобразования (портации) РНПП, с	37,0 (10837,0)	143,1 (10943,1)	42,2 (10842,2)	327,5 (11127,5)
Выигрыш при преобразовании (портации) РНПП	1554,95 (5,6)	402,45(5,5)	1364,83 (5,6)	175,8 (5,5)
Уровень реальной производительности	**0,63**	**0,64**	**0,80**	**0,86**
ГВС «Атлант–Д»				
Время преобразования (портации) РНПП, с	39,3 (10839,3)	144,6 (10944,6)	41,6 (10841,6)	332,1 (11132,1)
Выигрыш при преобразовании (портации) РНПП	1463,3 (6,3)	398,3 (6,2)	1383,8 (6,3)	173,4 (6,1)
Уровень реальной производительности	**0,65**	**0,67**	**0,80**	**0,84**

Как видно из данных таблицы 1, для задач с единственным информационно–независимым базовым подграфом (DES, MD–5, SHA–1) преобразование кадровой структуры средствами ТРНП выполняется существенно (на 2–3 десятичных порядка) быстрее, чем преобразование схемотехником, что является ожидаемым эффектом автоматизации. Выигрыш во времени портации при синтезе загрузочного конфигурационного файла, в зависимости от вида информационной зависимости, находится в диапазоне от 3 до 6 раз. Снижение выигрыша во времени портации по сравнению с временем преобразования РНПП объясняется заметным вкладом времени синтеза проекта, зависящего от логической емкости ПЛИС и одинакового для средств ТРНП и схемотехников. Незначительное изменение времени преобразования кадровой структуры одной и той же задачи для разных аппаратных платформ обусловлено сравнительно небольшим количеством вариантов, рассматриваемых при выборе рационального решения, что экспериментально подтверждает третье и четвертое положения, выдвигаемые для защиты. Достигнутый средствами ТРНП уровень реальной производительности составляет не менее 0,63 по сравнению с решениями прикладных программистов для РВС, что является приемлемым результатом для средств

автоматизации и превышает уровень, обеспечиваемый рассмотренными в первой главе HLS–компиляторами. Представленные в таблице 1 результаты портации задач подтверждены актами о внедрении результатов в ФГУП «НИИ Квант», СпбГУ, НИИ МВС ЮФУ, НИЦ СЭ и НК.

Проверка портации прикладных задач с информационно–взаимозависимыми подзадачами разной вычислительной трудоемкости проводилась для всех рассмотренных аппаратных платформ РВС и ГВС. Для задачи с информационной зависимостью «флагман»–«катер» портировалась РНПП решения СЛАУ методом Гаусса, состоящая из прямого («флагман») и обратного хода («катер»). Портация задачи информационной зависимости «флагман»–{катер} выполнялась для расчета энергетических характеристик взаимодействия разных молекул, примером зависимости «флагман»–«катер»–«софлагман» служила задача расчета свертки сигналов, а в качестве зависимости «взаимодействующие равные» использовалось информационно–зависимое объединение алгоритмов DES, ГОСТ и AES. Результаты экспериментальной проверки средств ТРНП при портации прикладных задач с информационно–взаимозависимыми подзадачами разной вычислительной трудоемкости на целевые архитектуры РВС и ГВС приведены в таблице 2.

Таблица 2. Результаты портации задач с информационно–взаимозависимыми подзадачами на целевые архитектуры РВС и ГВС

Задачи	**Гаусс**	**Расчет энергий**	**Свертка**	**DES–ГОСТ–AES**
РВС «Тайгета»				
Время преобразования (портации) РНПП, с	1516,3 (12316,3)	856,4 (11656,4)	946,5 (11756,5)	481,3 (11281,3)
Выигрыш при преобразовании (портации) РНПП	37,9 (5,5)	61,7 (5,8)	60,2 (5,8)	119,6 (6,0)
Уровень реальной производительности	**0,86**	**0,60**	**0,75**	**0,67**
РВС «Терциус»				
Время преобразования (портации) РНПП, с	1789,2 (26989,2)	1923,4 (26493,4)	1293,6 (26493,6)	596,2 (25796,2)
Выигрыш при преобразовании (портации) РНПП	32,1 (3,0)	44,5 (3,1)	48,3 (3,1)	96,6 (3,2)
Уровень реальной производительности	**0,80**	**0,63**	**0,77**	**0,75**
ГВС «Атлант–10»				
Время преобразования (портации) РНПП, с	985,2 (11785,2)	732,1 (11532,1)	732,1 (11532,1)	298,3 (11098,3)
Выигрыш при преобразовании (портации) РНПП	58,4 (5,1)	78,6 (5,3)	78,6 (5,3)	193,0 (5,5)
Уровень реальной производительности	**0,68**	**0,61**	**0,67**	**0,63**
ГВС «Атлант–Д»				
Время преобразования (портации) РНПП, с	1001,6 (11801,6)	741,7 (11541,7)	987,5 (11787,5)	302,6 (11102,6)
Выигрыш при преобразовании (портации) РНПП	57,5 (5,8)	77,6 (5,9)	58,3 (5,8)	190,2 (6,1)
Уровень реальной производительности	**0,65**	**0,67**	**0,67**	**0,60**

Для задач с информационно–взаимозависимыми подзадачами разной вычислительной трудоемкости преобразование кадровой структуры средствами ТРНП также выполняется существенно (на 2–3 десятичных порядка) быстрее, чем преобразование схемотехником. Выигрыш во времени портации при синтезе загрузочного конфигурационного файла, в зависимости от вида информационной зависимости, находится в диапазоне от 3 до 6 раз. Снижение выигрыша во времени портации по сравнению с временем преобразования РНПП, как и в предыдущем рассмотренном случае, объясняется заметным вкладом времени синтеза проекта, зависящего от логической емкости ПЛИС и одинакового для средств ТРНП и схемотехников. Незначительное изменение времени преобразования кадровой структуры одной и той же задачи для разных аппаратных платформ обусловлено сравнительно небольшим количеством вариантов, рассматриваемых при выборе рационального решения, что экспериментально подтверждает третье и четвертое положения, выдвигаемые для защиты. Достигнутый уровень реальной производительности варьируется по рассматриваемым задачам и составляет не менее 0,63, что соответствует уровню создаваемых схемотехниками решений задач на РВС. Представленные в таблице 2 результаты портации задач подтверждены актами о внедрении результатов в ФГУП «НИИ Квант», ЮНЦ РАН, НИИ МВС ЮФУ, НИЦ СЭ и НК.

Время портации прикладных задач с различными видами информационной зависимости в рассматриваемые целевые архитектуры ГВС, как видно из таблиц 1–2, не превышает 30 минут. Таким образом, разработанные программные средства поддержки технологии ресурсонезависимого программирования для портации прикладных задач с единственным базовым подграфом и с информационно–взаимозависимыми подзадачами разной вычислительной трудоемкости позволяют существенно (на 2–3 десятичных порядка) сократить время преобразования кадровой структуры и в 3–6 раз уменьшить время портации ресурсонезависимых программ на различные архитектуры и конфигурации гибридных вычислительных систем, обеспечивая при этом реальную производительность не ниже 0,6 от создаваемых схемотехниками решений.

Результаты проверки работоспособности разработанных в диссертации методов и алгоритмов портации для задач символьной обработки, линейной алгебры, цифровой обработки сигналов подтверждают представленные во 2–4 главах диссертации теоретические положения ресурсонезависимого программирования, что позволяет сделать вывод об успешном решении поставленной научной задачи. Практическое использование полученных результатов и их внедрение в практику автоматической портации прикладных задач различных областей для гибридных вычислительных систем позволят сократить время создания рациональных решений с заданным уровнем реальной производительности до 30 минут.

Дальнейшее развитие разработанных инструментальных средств технологии ресурсонезависимого программирования возможно в направлении автоматизации создания информационного графа и максимально–параллельной кадровой структуры, описанной в синтаксисе COLAMO, из последовательной программы на одном из широко распространенных языков программирования. Автоматическое построение информационного графа из последовательной программы, автоматическое его преобразование к ресурсонезависимой параллельно–конвейерной форме и портация последней в целевые архитектуры ГВС открывают возможности создания автоматического распараллеливающего компилятора для систем с распределенной памятью, что является новым практическим результатом, существенным для теории и практики параллельного программирования.

В заключении работы приведены основные выводы, полученные при решении поставленной научной проблемы и ее частных задач. Сформулирован вывод о достижении цели, а также приведены рекомендации по возможным направлениям дальнейших исследований. Успешная практическая апробация разработанных теоретических основ и реализующих их программных средств на ряде прикладных задач с наиболее распространенными видами информационной зависимости подтверждает корректность разработанных методов технологии ресурсонезависимого программирования и позволяют сделать вывод об успешном решении поставленной научной проблемы. Внедрение полученных в диссертации результатов позволит существенно сократить сроки портации прикладных задач на существующих и перспективных образцах гибридных вычислительных систем, что, помимо прямого экономического эффекта, имеет важное значение для развития критической технологии «Технологии и программное обеспечение распределенных и высокопроизводительных вычислительных систем», утвержденной Указом Президента Российской Федерации от 7 июля 2011 г. №899, и повышения обороноспособности страны.

ОСНОВНЫЕ РЕЗУЛЬТАТЫ И ВЫВОДЫ

Основной научный результат диссертации заключается в решении **актуальной научной проблемы** – разработке теоретических основ, методов и инструментальных программных средств технологии ресурсонезависимого программирования, обеспечивающих сокращение времени портации прикладных задач для гибридных вычислительных систем различных архитектур и конфигураций при заданном уровне реальной производительности.

Научная новизна результатов диссертационной работы определяется тем, что в ней:

1. Проведен анализ современных вычислительных архитектур для гетерогенных вычислительных систем, определены тенденции их развития, логические и технологические ограничения, а также актуальные требования эффективности реализации прикладных задач различных классов.
2. Проведен анализ известных методов и технологий программирования высокопроизводительных вычислительных систем и сформулированы принципы программирования конвергентных архитектур гибридных вычислительных систем.
3. Сформулированы требования и разработаны теоретические основы ресурсонезависимого программирования гибридных вычислительных систем для прикладных задач различных классов.
4. Сформулированы принципы создания единой для различных архитектур модели вычислений и формы ее представления, параметром которой является доступный аппаратный ресурс.
5. Разработаны методы преобразования параметризуемой ресурсом единой формы представления вычислений для рационального решения задач на заранее неопределенном вычислительном ресурсе гибридных вычислительных систем.
6. Разработана методика преобразования задач к рациональной реализации в гибридной вычислительной системе с помощью формальных преобразований.
7. Разработаны методы и алгоритмы преобразования прикладных задач, содержащих связанные информационной зависимостью подзадачи с различной степенью параллелизма.

8. Разработанные принципы, модель, методы и алгоритмы преобразования единой формы прикладных задач объединены в технологию ресурсонезависимого программирования гибридных вычислительных систем.

9. Проведены практическая апробация разработанной технологии ресурсонезависимого программирования при решении прикладных задач различных предметных областей и сравнение результатов с известными решениями.

10. Прикладным результатом работы является использование разработанных теоретических основ, методов, алгоритмов и программных средств для сокращения времени и стоимости портации параллельных программ при выполнении более 25 научно–исследовательских и опытно–конструкторских работ и их внедрение в 5 профильных организациях и предприятиях, что подтверждается актами внедрения ФГУП «НИИ Квант», ЮНЦ РАН, Санкт–Петербургского университета, НИИ МВС ЮФУ и НИЦ СЭ и НК.

В целом результаты диссертации можно квалифицировать как новое крупное научное достижение в области совершенствования технологий программирования высокопроизводительных вычислительных систем по п. 1 («Модели, методы и алгоритмы проектирования, анализа, трансформации, верификации и тестирования программ и программных систем») в части «Модели, методы и алгоритмы проектирования и трансформации программ» и п.8 («Модели и методы создания программ и программных систем для параллельной и распределенной обработки данных, языки и инструментальные средства параллельного программирования») в части «моделей и методов создания программ для параллельной обработки данных», а также «инструментальных средств параллельного программирования» паспорта специальности 2.3.5 «Математическое и программное обеспечение вычислительных систем, комплексов и компьютерных сетей».

Разработанные в диссертации подходы могут применяться при программировании существующих и перспективных гибридных вычислительных систем различных архитектур и конфигураций, а также быть интересны специалистам в области теории и практики параллельного программирования.

Благодарности. Автор диссертации выражает благодарность своему научному консультанту доктору технических наук, профессору Левину Илье Израилевичу за консультации, постоянное внимание к работе и поддержку, а также кандидатам технических наук Гудкову Вячеславу Александровичу, Гуленку Андрею Александровичу, Сорокину Дмитрию Анатольевичу за полезные обсуждения и рекомендации.

ОСНОВНЫЕ ПУБЛИКАЦИИ ПО ТЕМЕ ДИССЕРТАЦИИ

1. Левин, И. И. К вопросу об автоматическом создании параллельных прикладных программ для реконфигурируемых вычислительных систем / И. И. Левин, А. И. Дордопуло // Вычислительные технологии. – 2020. – Т. 25, № 1. – С. 66–81. – DOI 10.25743/ICT.2020.25.1.005. (**ВАК РФ**).

2. Дордопуло, А. И. Применение методов редукции производительности для сокращения числа анализируемых вариантов параллельной программы / А. И. Дордопуло // Вестник компьютерных и информационных технологий. – 2019. – № 9(183). – С. 43–49. – DOI 10.14489/vkit.2019.09.pp.043–049. (**ВАК РФ**).

3. Архитектурно–независимая программа быстрого преобразования Фурье на языке программирования SET@L / И. И. Левин, А. И. Дордопуло, И. В. Писаренко, А. К. Мельников // Вестник Рязанского государственного радиотехнического университета. – 2019. – № 68. – С. 28–36. – DOI 10.21667/1995–4565–2019–68–2–28–36. (**ВАК РФ**).

4. Hybrid computer system programming technology with adaptation and scaling of calculations / A. A. Gulenok, A. I. Dordopulo, I. I. Levin, V. A. Gudkov // Bulletin of the South Ural State University. Series: Computational Mathematics and Software Engineering. – 2017. – Vol. 6, No. 1. – P. 73–86. – DOI 10.14529/cmse170105. (**ВАК РФ**).

5. Программирование вычислительных систем гибридного типа на языке программирования COLAMO / А. И. Дордопуло, И. И. Левин, И. А. Каляев [и др.] // Известия ЮФУ. Технические науки. – 2016. – № 11(184). – С. 39–54. – DOI 10.18522/2311–3103–2016–11–3954. (**ВАК РФ**).

6. Параллельно–конвейерная форма программы для программирования вычислительных систем гибридного типа / А. И. Дордопуло, И. И. Левин, И. А. Каляев [и др.] // Вестник Уфимского государственного авиационного технического университета. – 2016. – Т. 20, № 3(73). – С. 122–128. (**ВАК РФ**).

7. Технология программирования вычислительных систем гибридного типа / И. А. Каляев, А. И. Дордопуло, И. И. Левин [и др.] // Вычислительные технологии. – 2016. – Т. 21, № 3. – С. 33–44. (**ВАК РФ**).

8. Решение задач с существенно–переменной интенсивностью потоков данных на реконфигурируемых вычислительных системах / И. И. Левин, Д. А. Сорокин, А. К. Мельников, А. И. Дордопуло // Вестник компьютерных и информационных технологий. – 2012. – № 2(92). – С. 49–56. (**ВАК РФ**).

9. Высокопроизводительные многопроцессорные системы с реконфигурируемой архитектурой для цифровой обработки сигналов / И. А. Каляев, И. И. Левин, Е. А. Семерников, А. И. Дордопуло // Вестник Концерна ПВО Алмаз–Антей. – 2011. – № 2(6). – С. 88–104. (**ВАК РФ**).

10. Левин, И. И. Ресурсонезависимое программирование многопроцессорных систем / И. И. Левин, А. И. Дордопуло, В. А. Гудков // Известия ТРТУ. – 2006. – № 9–1(64). – С. 174–178. (**ВАК РФ**).

11. High–Level Synthesis of Scalable Solutions from C–Programs for Reconfigurable Computer Systems / A. I. Dordopulo, I. I. Levin, V. A. Gudkov, A. A. Gulenok // Lecture Notes in Computer Science. – 2021. – Vol. 12942 LNCS. – P. 88–102. – DOI 10.1007/978–3–030–86359–3_7. (**Scopus**)

12. Dordopulo, A. I. Performance Reduction for Automatic Development of Parallel Applications for Reconfigurable Computer Systems / A. I. Dordopulo, I. I. Levin // Supercomputing Frontiers and Innovations. – 2020. – Vol. 7, No. 2. – P. 4–23. – DOI 10.14529/jsfi200201. (**Scopus**)

13. Software Development Tools for FPGA–Based Reconfigurable Systems Programming / I. Levin, V. Gudkov, G. Yevstafiyev [et al.] // Communications in Computer and Information Science. – 2019. – Vol. 1129. – P. 625–640. – DOI 10.1007/978–3–030–36592–9_51. (**Scopus**)

14. High–performance reconfigurable computer systems with immersion cooling / I. Levin, A. Dordopulo, A. Fedorov, Y. Doronchenko // Communications in Computer and Information Science. – 2018. – Vol. 910. – P. 62–76. – DOI 10.1007/978–3–319–99673–8_5. (**Scopus**)

15. Развитие отечественных многокристальных реконфигурируемых вычислительных систем: от воздушного к жидкостному охлаждению / И. А. Каляев, А. И. Дордопуло, И. И. Левин, А. М. Федоров // Труды СПИИРАН. – 2017. – № 1(50). – С. 5–31. – DOI 10.15622/sp.50.1. (**Scopus**)

16. Immersion liquid cooling FPGA–based reconfigurable computer system / I. I. Levin, Y. I. Doronchenko, M. K. Raskladkin [et al.] // IFAC–PapersOnLine. – 2016. – Vol. 49, No. 25. – P. 366–371. – DOI 10.1016/j.ifacol.2016.12.070. (**Scopus**)

17. Porting of parallel applications to reconfigurable computer systems with various architectures and configurations / A. I. Dordopulo, V. B. Kovalenko, V. A. Gudkov, L. M. Slasten // 2016 5th International Conference on Informatics, Electronics and Vision, ICIEV 2016. – 2016. – P. 1122–1127. – DOI 10.1109/ICIEV.2016.7760174. (**Scopus**)

18. Parallel Application Porting by Means of Soft–Architectures / A. I. Dordopulo, V. A. Gudkov, A. A. Gulenok [et al.] // IFAC–PapersOnLine. – 2016. – Vol. 49, No. 25. – P. 372–377. – DOI 10.1016/j.ifacol.2016.12.071. (**Scopus**)

19. FPGA–based reconfigurable computer systems for digital image processing / I. I. Levin, I. A. Kalyaev, V. A. Gudkov [et al.] // International Journal of Circuits, Systems and Signal Processing. – 2015. – Vol. 9. – P. 27–32.– Режим доступа: http://www.naun.org/main/NAUN/circuitssystemssignal/2015/a082005–028.pdf (дата обращения 24.04.2023). (**Scopus**)

20. FPGA–based reconfigurable computer systems / I. A. Kalyaev, I. I. Levin, A. I. Dordopulo, L. M. Slasten // Proceedings of 2013 Science and Information Conference, SAI 2013. – 2013. – P. 148–155.– Режим доступа: https://ieeexplore.ieee.org/document/6661730 (дата обращения 24.04.2023). (**Scopus**)

Свидетельства о регистрации программ

1. **Свидетельство о государственной регистрации программ для ЭВМ** № 2022660796 Российская Федерация. Программа портации кадровой структуры на гибридные вычислительные системы различных архитектур и конфигураций : № 2022619611 : заявл. 25.05.2022 : опубл. 09.06.2022 / Левин И. И., Гудков В. А., Дордопуло А. И., Гуленок А. А. ; правообладатель Общество с ограниченной ответственностью «НИЦ супер–ЭВМ и нейрокомпьютеров» (ООО «НИЦ СЭ и НК»).

2. **Свидетельство о государственной регистрации программ для ЭВМ** № 2022660797 Российская Федерация. Программа преобразования фрагментов кадровой структуры на языке COLAMO в микро–кадры : № 2022619494 : заявл. 25.05.2022 : опубл. 09.06.2022 / Гуленок А. А., Дордопуло А. И., Левин И. И., Гудков В. А. ; правообладатель Общество с ограниченной ответственностью «НИЦ супер–ЭВМ и нейрокомпьютеров» (ООО «НИЦ СЭ и НК»).

3. **Свидетельство о государственной регистрации программ для ЭВМ** № 2022660467 Российская Федерация. Программа преобразования информационного графа задачи в ресурсонезависимую кадровую структуру на языке Colamo : № 2022619531 : заявл. 25.05.2022 : опубл. 03.06.2022 / Гудков В. А., Левин И. И., Дордопуло А. И., Гуленок А. А. ; правообладатель Общество с ограниченной ответственностью «НИЦ супер–ЭВМ и нейрокомпьютеров» (ООО «НИЦ СЭ и НК»).

4. **Свидетельство о государственной регистрации программ для ЭВМ** № 2016663176 Российская Федерация. Программный модуль программных средств синтеза масштабируемых параллельно–конвейерных решений прикладных задач для многопроцессорной вычислительной системы гибридного типа : № 2016619086 : заявл. 23.08.2016 : опубл. 29.11.2016 / И. И. Левин, А. И. Дордопуло, А. А. Гуленок, А. В. Чкан ; правообладатель федеральное государственное автономное образовательное учреждение высшего образования «Южный федеральный университет» (Южный федеральный университет).

5. **Свидетельство о государственной регистрации программ для ЭВМ** № 2016661590 Российская Федерация. Программный модуль программных средств

трансляции единого языка высокого уровня для различных типов вычислительных узлов многопроцессорной вычислительной системы гибридного типа : № 2016618869 : заявл. 17.08.2016 : опубл. 14.10.2016 / И. И. Левин, А. И. Дордопуло, В. А. Гудков, Г. А. Евстафьев ; правообладатель федеральное государственное автономное образовательное учреждение высшего образования «Южный федеральный университет» (Южный федеральный университет).

6. **Свидетельство о государственной регистрации программы для ЭВМ** № 2011612843 Российская Федерация. Интегрированная среда разработки аппаратно–программных решений : № 2011610860 : заявл. 11.02.2011 : опубл. 08.04.2011 / Левин И. И., Дордопуло А. И., Гудков В. А. ; правообладатель федеральное государственное автономное образовательное учреждение высшего образования «Южный федеральный университет» (Южный федеральный университет).

Личный вклад автора.

В работах, написанных в соавторстве, автору принадлежит:

[1, 4–7, 9, 12] – представление кадровой структуры с использованием вычислительного ресурса как параметра параллельно–конвейерной реализации задачи и метод преобразования и сокращения аппаратных затрат кадровой структуры с помощью редукции производительности;

[3, 11, 13-18] – принципы преобразования параметризованной аппаратным ресурсом кадровой структуры к архитектуре и конфигурации вычислительной системы;

[8] – принципы преобразования параметризованной аппаратным ресурсом кадровой структуры к архитектуре и конфигурации гибридной вычислительной системы с помощью редукции производительности;

[10, 19, 20] – разработка единой модели параллельных вычислений на основе информационного графа задачи.

Вклад автора в создание зарегистрированных программ для ЭВМ состоит:

[1–3] – в разработке алгоритмов программной реализации компонент средств ТРНП;

[4, 5] – в разработке методов и алгоритмов трансляции и синтеза масштабируемых параллельно–конвейерных решений прикладных задач для вычислительной системы гибридного типа;

[6] – в разработке алгоритмов и программной реализации средств трансляции и синтеза.

Подписано к печати ___.__.2023 г.
Формат 60x84$^{1/16}$. Бумага офсетная. Печать ризография.
Заказ № ___. Тираж 120 экз.

Отпечатано в типографии издателя Ступина С.А.
347900, Ростовская область, г. Таганрог, пер. Лермонтовский, 25
Тел/факс: 8 (8634) 311–288.