Федеральное государственное бюджетное учреждение науки
Институт динамики систем и теории управления имени В.М. Матросова
Сибирского отделения Российской академии наук

На правах рукописи

Феоктистов Александр Геннадьевич

Организация предметно-ориентированных распределенных вычислений в гетерогенной среде на основе мультиагентного управления заданиями

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

Диссертация на соискание ученой степени
доктора технических наук

Научный консультант:
академик РАН,
И.В. Бычков

Иркутск – 2021

Оглавление

Введение

Актуальность темы исследования. В настоящее время решение крупномасштабных фундаментальных и прикладных задач, обеспечивающих прорывные результаты и конкурентное преимущество в научной, производственной, экономической и других сферах человеческой деятельности, обоснованно требует применения высокопроизводительных вычислений – High Performance Computing (HPC). Анализ современных тенденций развития параллельных и распределенных вычислительных систем, базирующийся на теоретических и практических результатах исследований ведущих российских и зарубежных ученых, в том числе С.М. Абрамова, А.И. Аветисяна, А.П. Афанасьева, В.Б. Бетелина, И.В. Бычкова, А.В. Бухановского, Вл.В. Воеводина, В.П. Гергеля, Б.М. Глинского, В.П. Иванникова, В.А. Ильина, В.П. Ильина, И.А. Каляева, В.Н. Коваленко, Д.А. Корягина, В.В. Коренькова, В.Д. Корнеева, И.И. Левина, А.И. Легалова, Ю.Е. Малашенко, В.Э. Малышкина, Г.А. Опарина, Г.И. Радченко, С.И. Смагина, Л.Б. Соколинского, А.А. Сорокина, В.В. Стегайлова, О.В. Сухорослова, В.В. Топоркова, В.Г. Хорошевского, А.Н. Черных, Б.Н. Четверушкина, М.В. Якобовского, D. Abramson, D. Andersen, P. Bouvry, K. Bubendorfer, R. Buyya, H. Casanova, T.L. Casavant, R.S. Chang, E. Deelman, J. Dongarra, I. Foster, K. Kesselman, Y.C. Lee, P. Sloot, D. Talia, T. Tannenbaum, H. Topcuoglu, A. Zomaya и других известных специалистов, позволяет сделать ряд важных выводов.

Тенденция развития современных гетерогенных распределенных вычислительных сред (ГРВС), таких как грид-системы или облачные инфраструктуры, заключается в стремительном росте их масштаба и суммарной производительности, которые зачастую достигаются путем увеличения числа их элементов. В связи с этим одним из ключевых свойств современных научных приложений, таких как распределенные пакеты прикладных программ (РППП) и системы, основанные на применении научного рабочего процесса (англ., workflow), является их масштабируемость (обеспечение роста ускорения вычислений в

процессе решения задачи при увеличении числа узлов и поддержка близкой к единице эффективности использования ресурсов – отношения ускорения к числу узлов).

Как правило, разработка приложений и генерация заданий по решению задач осуществляется с помощью специализированных инструментальных средств. В то же время выполнение заданий и выделение им ресурсов в ГРВС производится традиционными системами управления распределенными вычислениями, установленными в узлах среды. В их числе Portable Batch System (PBS) [106, 206], HTCondor [111, 137] и Simple Linux® Utility for Resource Management (SLURM) [117, 190]. В качестве связующего программного обеспечения (ПО) зачастую используются различные метапланировщики (GridWay [99, 107], Condor Directed Acyclic Graph Manager (Condor DAGMan) [200], Production and Distributed Analysis (PanDA) [144], Nimrod/G [30], gLite [270] и др.), обеспечивающие взаимодействие между вычислительными подсистемами инструментальных комплексов и вышеупомянутыми системами управления в узлах среды. Успешным примером реализации подобного подхода является грид-инфраструктура Национальной нанотехнологической сети (ГридННС) [280].

Для повышения гибкости и управляемости гетерогенной среды выполнения распределенных вычислений зачастую прибегают к виртуализации. В виртуализированных средах используются различные гипервизоры, такие как Xen, Kernel-based Virtual Machine (KVM) и VMware ESXi [112], средства управления контейнерами Linux Containers (LXC) и Docker [58], а также платформы, подобные OpenStack [25].

Конечные пользователи приложений, запускающие задания по решению своих задач в ГРВС, являются, как правило, специалистами в разных предметных областях. Они имеют свои субъективные требования к выполнению приложений и эксплуатации общих ресурсов среды. В роли пользовательских критериев качества решения задачи в диссертации рассматриваются общее время и стоимость решения задач, а также ускорение вычислений. Наличие большого числа пользователей ГРВС и разнообразие спектра решаемых ими задач обуславливает необходимость

предоставления им разных уровней обслуживания в зависимости от заданных критериев качества решения задачи, учитывая в то же время предпочтения владельцев ресурсов среды. В рамках диссертации к таким предпочтениям относятся эффективность их использования, средняя загрузка, а также степень ее балансировки, оцениваемая среднеквадратическим отклонением. Эти показатели, а также ускорение вычислений рассчитываются традиционными методами [248].

Обеспечение требуемых уровней обслуживания в ГРВС невозможно без комплексного решения ряда проблем: гибкого администрирования процесса обслуживания заданий; учета специфики предметных областей задач пользователей; устранения различного рода неопределенностей, возникающих при планировании вычислений и распределении ресурсов из-за полного или частичного отсутствия информации о характеристиках и свойствах вычислительных процессов; обеспечения масштабируемости вычислений в разнородной среде и др. Решение этих проблем с помощью широко используемых сегодня вышеперечисленных систем управления неосуществимо в полной мере, так как они изначально создавались для других целей. Их информационно-управляющие модели, как правило, не совместимы с моделями, описывающими предметные области приложений.

Поэтому активно развиваются подходы к решению проблем обеспечения удовлетворительного уровня обслуживания в гетерогенной вычислительной среде, базирующиеся на интеллектуализации методов и средств управления с помощью мультиагентных технологий. Большой вклад в развитие теории и практики интеллектуальных систем управления внесли С.Н. Васильев, Т.А. Гаврилова, В.И. Городецкий, А.И. Дивеев, И.А. Каляев, Л.В. Массель, Д.А. Поспелов, К.А. Пупков, П.О. Скобелев, В.Б. Тарасов, В.Ф. Хорошевский, В.П. Хранилов, N. Jennings, A. Kusiak, M. De Weerdt, G. Weiss, M. Wooldridge и другие известные специалисты в этой области.

При создании интеллектуальных систем управления распределенными вычислениями особое внимание привлекает использование экономических механизмов управления (см. работы В.В. Топоркова, B. Abramson и R. Buyya),

обеспечивающих повышение качества выполнения вычислительных процессов (сокращение времени и стоимости решения задач, повышение эффективности использования ресурсов, балансировки их нагрузки и др.).

Успешные практические решения в области мультиагентного управления распределенными вычислениями в гетерогенных средах известны (см., например, работы В.И. Городецкого, И.А. Каляева, Э.В. Мельника, А.В. Тимофеева, J. Cao, E.H. Durfee, S. Jarvis и A. Singh), однако многие мультиагентные системы создаются с ориентацией на специальные программно-аппаратные инфраструктуры и не могут в полной мере поддерживать требуемое функционирование в средах с иными характеристиками.

Все вышесказанное позволяет сделать вывод об актуальности разработки новых моделей, алгоритмов, инструментальных средств и технологии предметно-ориентированных распределенных вычислений и управления ими в гетерогенной среде.

Цель работы состоит в разработке технологии предметно-ориентированных распределенных вычислений, позволяющей согласовать критерии качества решения задачи и предпочтения владельцев ресурсов и улучшить показатели этих критериев и предпочтений по сравнению с известными метапланировщиками, такими как GridWay и Condor DAGMan, за счет использования мультиагентных технологий управления заданиями и привлечения дополнительных знаний в процессе планирования вычислений и распределения ресурсов.

Основными задачами для достижения поставленной цели являются:

- анализ современных подходов к организации распределенных предметно-ориентированных приложений, а также известных методов и средств управления заданиями таких приложений в ГРВС;
- разработка модели предметно-ориентированной ГРВС;
- разработка моделей для определения показателей качества выполнения заданий в такой среде;
- разработка системы классификации заданий;
- разработка алгоритмов мультиагентного управления заданиями РППП;

– разработка подхода к созданию приложений для решения крупномасштабных задач в ГРВС;
– разработка технологии предметно-ориентированных распределенных вычислений и мультиагентного управления в такой среде.

Объектом исследования являются методы и средства организации вычислений и распределения ресурсов в ГРВС.

Предметом исследования выступают модели, алгоритмы и инструментальные средства разработки РППП, а также мультиагентного управления их заданиями в ГРВС.

Методы исследования. При решении поставленных задач использовались методы и средства концептуального и имитационного моделирования, организации распределенных вычислений, реализации мультиагентных технологий, а также экономические механизмы регулирования спроса и предложения вычислительных ресурсов.

Научную новизну диссертации представляют следующие результаты исследования, выносимые на защиту и расширяющие существующий базис теории и практики распределенных вычислений:

1) разработана агрегированная модель ГРВС, которая в сравнении с подобными моделями обеспечивает взаимосвязанное представление алгоритмических знаний предметных областей решаемых задач, а также знаний о программно-аппаратной инфраструктуре среды и административных политиках использования ее ресурсов;
2) предложены модели и алгоритмы определения показателей качества решения задач в ГРВС, базирующиеся на применении предложенной агрегированной модели среды и набора специализированных методов прогнозирования времени выполнения заданий;
3) создана система классификации заданий, позволяющая в отличие от известных систем подобного назначения привлечь дополнительные экспертные знания администраторов узлов ГРВС для детализации спецификаций вычислительных процессов решения пользовательских задач

относительно особенностей ресурсов среды с целью снижения неопределенности в распределении заданий по узлам;

4) предложен мультиагентный алгоритм планирования вычислений и распределения ресурсов, характерными особенностями которого являются применение экономических механизмов регулирования спроса и предложения этих ресурсов, а также возможность его адаптации к различным моделям сочетания критериев качества решения задач и предпочтений владельцев ресурсов на основе методов дискретного многокритериального выбора;
5) разработан пакетный подход к организации предметно-ориентированных вычислений, базирующийся на построении РППП, среда функционирования которых в отличие от других подобных сред может включать ресурсы HPC-кластеров, грид-систем, облачных инфраструктур и других программно-аппаратных компонентов, а также обеспечивать их интегрированное использование для решения крупномасштабных задач;
6) создана технология предметно-ориентированных распределенных вычислений в ГРВС, интегрирующая вышеупомянутые модели, алгоритмы, систему классификации заданий и пакетный подход, специализированные инструментальные средства создания и применения РППП, а также программно-аппаратные ресурсы среды в рамках единой технологической цепочки решения крупномасштабных задач.

Практическая значимость. Применение вышеперечисленных результатов диссертационных исследований позволяет обеспечить управление заданиями в предметно-ориентированной ГРВС на основе согласования заданных критериев качества решения задач и предпочтений владельцев ресурсов.

Основные результаты диссертационных исследований были использованы в рамках следующих научно-технических работ:

– гранта № 075-15-2020-787 Министерства науки и высшего образования РФ на выполнение крупного научного проекта по приоритетным

направлениям научно-технологического развития (проект «Фундаментальные основы, методы и технологии цифрового мониторинга и прогнозирования экологической обстановки Байкальской природной территории»);

– проектов Российского фонда фундаментальных исследований № 01-07-90220-в «Разработка инструментальной среды для создания и поддержки функционирования распределенных пакетов знаний в сети Интернет», № 10-07-00146-а «Средства создания и поддержки проблемно-ориентированных распределенных систем, основанных на знаниях», № 15-29-07955-офи_м «Разработка методов, алгоритмов и инструментальных средств планирования выполнения масштабируемых приложений в разнородной кластерной Grid»; № 16-07-00931-а «Методология и инструментальные средства разработки и применения проблемно-ориентированных мультиагентных систем управления масштабируемыми вычислениями в разнородной распределенной вычислительной среде» и № 19-07-00097-а «Фундаментальные проблемы непрерывной интеграции функционального наполнения распределенных пакетов прикладных программ на основе инженерии знаний»;
– регионального проекта РФФИ и Правительства Иркутской области № 20-47-380002 р_а «Математическое и информационное моделирование инфраструктурных объектов Байкальской природной территории»;
– проектов фундаментальных исследований Президиума РАН № 21.6 «Разработка фундаментальных основ создания научной распределенной информационно-вычислительной среды на основе технологий GRID» (2004 г.), № 13.3 «Концептуальные основы и программные средства разработки проблемно-ориентированных распределенных вычислительных сред» (2009-2011 гг.) и № 14.1 «Модели, методы и инструментальные средства для испытания и оценки надежности функционирования проблемно-ориентированных распределенных вычислительных сред» (2012-2014 гг.), проект «Разработка новых подходов к созданию и

исследованию моделей сложных информационно-вычислительных и динамических систем с приложениями» программы 1.33П (2016 и 2017 гг.), проект «Фундаментальные проблемы решения сложных практических задач с помощью суперкомпьютеров» программы № 27 (2018 г.), проект «Методы, алгоритмы и инструментальные средства децентрализованного группового решения задач в вычислительных и управляющих системах» программы № 30 (2018 г.);

- НИР № 111-09-103 «Создание и внедрение в учебный процесс вычислительного кластера международного факультета» ФГБОУ ВО ИГУ» (2008-2010 гг.) и № 111-15-701 «Разработка и внедрение комплексов автоматизированного анализа и систематизации научных данных» (2017-2019 гг.);
- госзаданий и базовых тем исследований ИДСТУ СО РАН.

Под руководством автора разработаны следующие программные комплексы и вычислительные среды, функционирующие на базе интегрированных ресурсов Центра коллективного пользования (ЦКП) «Иркутский суперкомпьютерный центр СО РАН» (ИСКЦ), ИДСТУ СО РАН, Института систем энергетики им. Л.А. Мелентьева СО РАН (ИСЭМ СО РАН), Иркутского научного центра (ИНЦ) СО РАН и Международного института экономики и лингвистики (МИЭЛ) ФГБОУ ВО «Иркутский государственный университет» (ИГУ):

- инструментальный комплекс Distributed Computing Environment Toolkit (DISCENT) для организации распределенных вычислительных сред;
- инструментальные комплексы DIStributed COmputing system of Modular Programming (DISCOMP) и Object Relations in LANguage of DescriptiOns Tools (Orlando Tools), применяющиеся в качестве сред параллельного программирования ЦКП ИСКЦ;
- инструментальный комплекс SIRIUS III для автоматизации имитационного моделирования систем массового обслуживания;
- экспериментальная Грид ИДСТУ СО РАН;

– учебная информационно-вычислительная среда МИЭЛ ИГУ;
– предметно-ориентированная среда для исследования живучести энергетических инфраструктур.

Результаты диссертации применены в ИСЭМ СО РАН, МИЭЛ ИГУ, Обществе с ограниченной ответственностью (ООО) «Терминал комплекс» и ООО «Катанна». Акты о внедрении данных результатов и справки об использовании программ для ЭВМ, разработанных в рамках диссертационного исследования, приведены в Приложении А.

Достоверность и обоснованность полученных в диссертации результатов подтверждается корректным применением классических методов исследования, анализом адекватности разработанных моделей и алгоритмов на основе имитационного и полунатурного моделирования, а также в процессе решения практических задач.

Соответствие диссертации паспорту специальности. Тема и основные результаты диссертации соответствуют следующим областям исследований паспорта специальности 05.13.11 – «Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей»:

– модели, методы, алгоритмы, языки и программные инструменты для организации взаимодействия программ и программных систем;
– модели и методы создания программ и программных систем для параллельной и распределенной обработки данных, языки и инструментальные средства параллельного программирования.

Апробация. Основные результаты диссертационного исследования докладывались автором на следующих научных мероприятиях: 4th ACM International Conference on Future Networks and Distributed Systems (Санкт-Петербург, 2020 г.); 6th Latin America High Performance Computing Conference (Турриальба, Коста-Рика, 2019 г.); 1st and 2nd International Workshops on Information, Computation, and Control Systems for Distributed Environments (Иркутск, 2019 и 2020 гг.); V Международной конференции «Информационные и нанотехнологии» (Самара, 2019 г.); 5th Latin American Conference on High

Performance Computing (Букараманга, Колумбия, 2018 г.); Международных конференций «Суперкомпьютерные дни в России» (Москва, 2018 и 2019 гг.); научной конференции «Фундаментальные проблемы организации распределенных облачных вычислений при решении крупномасштабных научных задач» (Дивноморское, 2018 г.); IV и V Всероссийских научно-технических конференциях «Суперкомпьютерные технологии» (Дивноморское, 2016 и 2018 гг.); Международной конференции «Информационные технологии в науке, образовании и управлении» (Гурзуф, 2017 г.); научно-технической конференции «Управление в распределенных и сетевых системах» в рамках X и XI Всероссийских Мультиконференций по проблемам управления (Дивноморское, 2017 и 2019 гг.); XVIII Всероссийской конференции молодых ученых по математическому моделированию и информационным технологиям (Иркутск, 2017 г.); Международной конференции «Вычислительная и прикладная математика» в рамках «Марчуковских чтений» (Новосибирск, 2017 г.); XI Международной научной конференции «Параллельные вычислительные технологии» (Казань, 2017 г., Калининград, 2019 г.); 41th International Convention on «Information and communication technology, electronics and microelectronics» (Риека, Хорватия, 2016 г.), 12th International Symposium on Intelligent Systems (Москва, 2016 г.); III Российско-монгольской конференции молодых ученых по математическому моделированию, вычислительно-информационным технологиям и управлению (Иркутск, 2015 г.) и XV Байкальской всероссийской конференции «Информационные и математические технологии в науке и управлении» (Иркутск, 2010 г.), а также семинарах ИДСТУ СО РАН.

Публикации. Результаты научных исследований автора отражены в 76 научных работах [63–87, 324–327, 332–334, 337–340, 345–350, 352–370, 372–375, 377–379, 381, 382, 387–392]. Основные публикации представлены в российских журналах [79, 324, 334, 345, 347, 349, 350, 357, 358, 360, 362, 366–368, 372–375, 381, 382, 390, 392], рекомендованных Высшей аттестационной комиссией для опубликования научных результатов диссертации, а также проиндексированы в

международных базах цитирования Web of Science и Scopus [63–67, 69–77, 80–87]. Получено 19 свидетельств о регистрации программ для электронных вычислительных машин (ЭВМ) [323, 328–331, 335, 336, 341–344, 351, 371, 376, 380, 383–386].

Личный вклад автора. Все выносимые на защиту научные положения получены соискателем лично. Из совместных работ в диссертацию включены только те результаты, которые принадлежат непосредственно автору.

Структура работы. Диссертация состоит из введения, шести глав, заключения, библиографии из 402 наименований, списка принятых сокращений и 11 приложений. Общий объем основного текста работы – 262 страницы, включая 28 таблиц и 80 рисунков.

Глава 1. Агрегированная модель предметно-ориентированной гетерогенной распределенной вычислительной среды

Выбор оптимальной конфигурации вычислительной системы при проведении крупномасштабных паучных экспериментов определяется алгоритмами построения плана решения задачи и назначения подходящих ресурсов для его выполнения. Успешная работа этих алгоритмов во многом зависит от степени полноты информации о специфики решаемой задачи, а также вычислительных характеристиках и правилах использования ресурсов системы. Таким образом, возникает необходимость разработки модели ГРВС для представления всесторонних знаний об этой среде в некотором унифицированном и стандартизированном формате, который может быть применен различными программными комплексами, функционирующими в рамках данной среды.

В первой главе предложена агрегированная модель (АМ) предметно-ориентированной ГРВС, которая, в отличие от известных, позволяет осуществить взаимосвязанное представление алгоритмических знаний предметных областей решаемых задач, а также знаний о программно-аппаратной инфраструктуре среды и административных политиках, определенных для ее ресурсов. АМ обеспечивает фундаментальную основу для разработки моделей и алгоритмов управления вычислениями в ГРВС, включая планирование вычислений и распределения ресурсов. Ее построение выполнено с использованием методов и средств искусственного интеллекта, инженерии знаний и концептуального программирования.

В данной главе рассматриваются ключевые особенности вычислительной среды, на которую ориентированы диссертационные исследования. Проводится сравнительный анализ известных моделей управления заданиями в распределенных вычислительных средах. Предлагается подход к построению АМ предметно-ориентированной ГРВС. Описываются основные категории знаний, представленные в АМ такой среды, а именно алгоритмические знания предметных областей решаемых задач, знания о программно-аппаратной инфраструктуре

среды, а также экспертные знания администраторов среды. В конце главы приводятся краткие выводы о полученных в ней научных результатах.

1.1. Предметно-ориентированная гетерогенная распределенная вычислительная среда

Диссертационные исследования ориентированы на ГРВС, характеризующуюся рядом ключевых особенностей. На рисунке 1.1 представлена структурная схема, отражающая взаимодействие основных субъектов и объектов вычислительной среды.

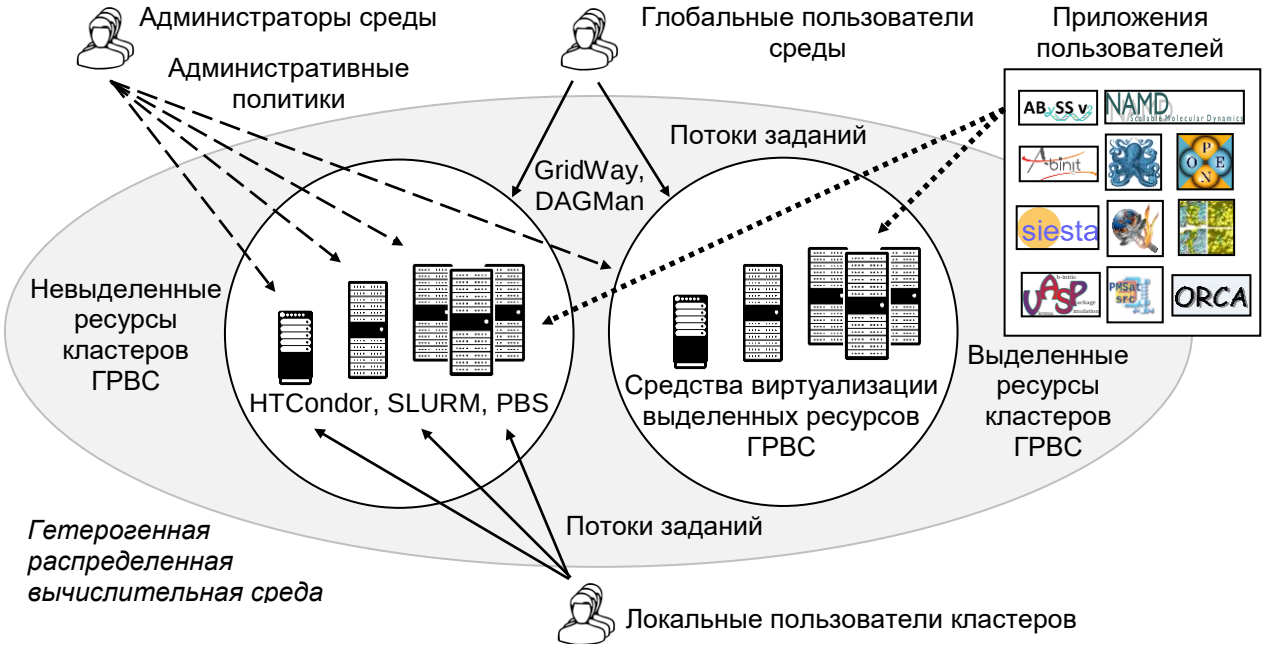


Рисунок 1.1 – Схема взаимодействия основных субъектов и объектов среды

Среда организуется на базе ресурсов ЦКП. В связи с этим все ее ресурсы являются общими для пользователей центра и распределяются на основе административных политик, определенных как на уровне вычислительной среды в целом, так и на уровне отдельных ее узлов. Порядок прохождения заданий пользователей в среде жестко регламентирован.

Административные политики в узлах среды могут различаться между собой. Они устанавливают правила (дисциплины) и квоты использования ресурсов

пользователями, критерии его эффективности, такие как загрузка ресурсов и ее балансировка, справедливость распределения ресурсов, а также различные реальные или виртуальные экономические показатели выполнения вычислительных работ.

В зависимости от масштабности решаемых задач, в инфраструктуру ГРВС могут быть включены персональные компьютеры (ПК), серверы, кластеры, грид-системы и облачные платформы. В качестве основных компонентов среды выступают вычислительные кластеры, в том числе гибридные кластеры с разнородными узлами. Кластеры организуются на базе как выделенных, так и невыделенных узлов, и, следовательно, существенно различаются по степени надежности своих ресурсов [70], определяемой временем наработки до отказа, стационарными или интервальными коэффициентами надежности. Вследствие усложнения инфраструктуры современных кластеров, включая рост числа их вычислительных элементов, вероятность сбоев программно-аппаратного обеспечения в процессе решения задач существенно повышается. Сбои происходят в случайные моменты времени.

Узлы разных кластеров отличаются своими вычислительными характеристиками (используемым интерконнектом, производительностью процессоров, числом ядер, объемами оперативной и дисковой памяти, числом уровней кэш-памяти и т.д.).

В рамках ЦКП решают задачи как локальные пользователи отдельных кластеров среды, так и глобальные – те, кому нужны ее интегрированные ресурсы. В обоих случаях они разделяют общие ресурсы среды при решении своих задач.

Для решения задачи пользователь должен сформировать задание системе, представляющее собой спецификацию вычислительного процесса. Такая спецификация содержит информацию о требуемых ресурсах, исполняемых прикладных программах, входных/выходных данных, а также другие необходимые сведения. Пользователь также определяет критерии качества решения своей задачи: время, стоимость, надежность, безопасность и другие характеристики.

Среда поддерживает обработку заданий разных классов: последовательных и

параллельных, независимых и взаимосвязанных, локальных и удаленных, многовариантных (сериализуемых), а также различающихся по времени, требуемому для выполнения задания, размерам необходимой оперативной и дисковой памяти, соотношению вещественных и целочисленных операций и другим вычислительным характеристикам. На кластерах установлены системы управления прохождением заданий (СУПЗ). Они осуществляют обработку заданий, поступающих в их общие очереди и затем направляемых оттуда на назначенные ресурсы среды. Предполагается, что свободных ресурсов среды, как правило, недостаточно для одновременного обслуживания всех заданий, находящихся в очередях.

СУПЗ обеспечивают интерфейс пользователя с системой, регистрацию и подключение вычислительных ресурсов и пользователей, администрирование прав доступа к вычислительным ресурсам и квот на их использование, защиту программных, аппаратных и информационных ресурсов, соблюдение прав их владельцев, организацию работы с файлами, поддержку многозадачного режима работы, обслуживание очередей заданий, планирование загрузки ресурсов, поддержку параллельных и распределенных вычислений.

Интеграция кластерных ресурсов в рамках среды может базироваться на следующих технологиях:

- использование СУПЗ, например, HTCondor, PBS Torque или SLURM, поддерживающих проведение вычислений на нескольких кластерах;
- распределение заданий с помощью метапланировщиков GridWay или Condor DAGMan;
- применение пакета Globus Toolkit (GT) [96] в качестве связующего ПО, взаимодействующего с установленными на кластерах СУПЗ;
- виртуализация выделенной части ресурсов ГРВС с помощью платформы OpenStack, поддерживающей работу как с широким спектром гипервизоров для управления виртуальными машинами (ВМ), так и со средствами работы с контейнерами.

В первых двух случаях интеграция осуществляется на основе модели грид-вычислений. В последнем случае виртуализация ресурсов производится с использованием модели облачных вычислений. Виртуализированные ресурсы представляют собой приватное облако. СУПЗ включаются в состав виртуальных сущностей (ВМ или контейнеров).

В рамках среды функционируют приложения, характеризующиеся разной степенью масштабируемости вычислений, чувствительности к неоднородности ресурсов, потребности в виртуализации ресурсов среды, а также необходимости интеграции модели своей предметной области с информацией о программно-аппаратной инфраструктуре среды и административных политиках, определенных для ее ресурсов. Приложения, чувствительные к неоднородности ресурсов, выполняются, как правило, в однородных узлах какого-либо кластера или в виртуальной среде. Потребность в ней возникает также и у приложений, использующих ПО, отличное от того, которое установлено в узлах среды. Пользователи приложений, которые могут выполняться на разнородных ресурсах, заинтересованы в максимальном использовании вычислительных мощностей ГРВС.

Модель предметной области решаемой задачи (или класса задач) представляет собой формализацию описания исследуемой системы, ее объектов, их взаимосвязей, а также происходящих в ней событий и процессов. Сложная система, как правило, характеризуется большим числом объектов, между которыми существуют зависимости различной природы, разнообразием выполняемых функций, нетривиально организованным управлением, наличием внешних стохастических воздействий из внешнего окружения и другими особенностями.

Процессы решения задач предметной области, связанные с исследованием системы, так же являются достаточно сложными и неоднозначными. Поэтому для достижения требуемого качества их выполнения в ГРВС зачастую необходима дополнительная информация о возможностях ресурсов среды и допустимых правилах их использования, позволяющая оптимально отобразить вычислительные процессы на структуру используемых ресурсов.

В общем случае в среде имеется программно-аппаратная вычислительная избыточность (приложение может быть размещено и запущено в разных узлах среды, а одни и те же вычисления могут быть произведены с помощью различных приложений). Приложения в дискретные моменты времени порождают потоки заданий, конкурирующие за общие ресурсы среды. Эти потоки могут в дальнейшем объединяться и перераспределяться, создавая новые потоки. На рисунке 1.2 схематично изображены потоки заданий, поступающие с машин пользователей на управляющие узлы и далее на кластеры. Линии разных типов и цветов соответствуют потокам разной мощности.

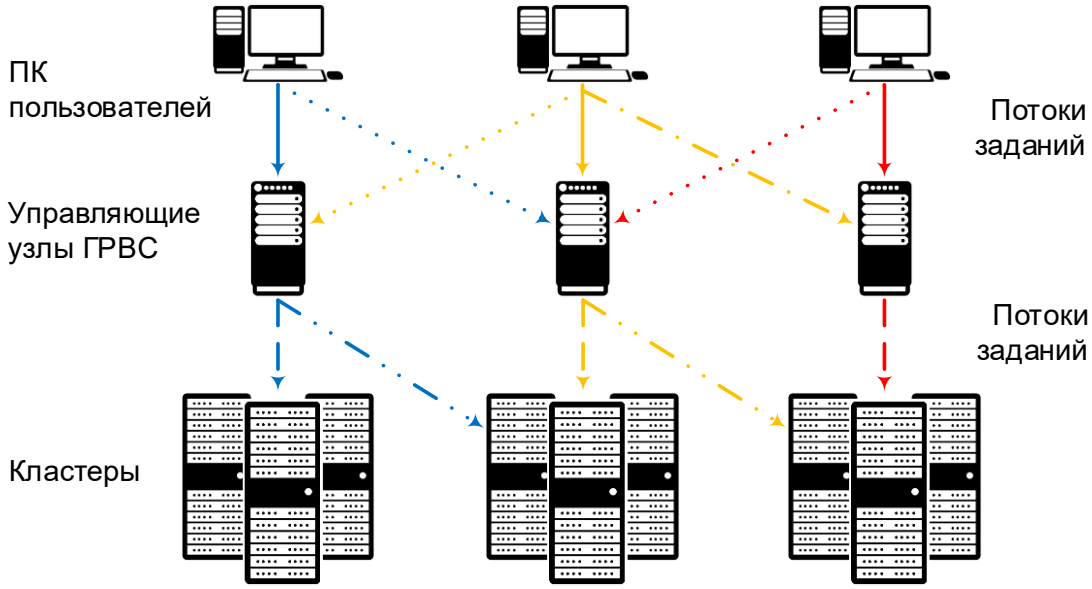


Рисунок 1.2 – Потоки заданий

Сложность управления потоком заданий в ГРВС, интегрирующей виртуализированные и традиционные кластерные ресурсы, возникает из-за существующих различий в моделях облачных и грид-вычислений, а также конфликтов между предпочтениями владельцев ресурсов и критериями качества выполнения заданий, которые указываются пользователями этих ресурсов.

Таким образом, предметно-ориентированная ГРВС представляет собой совокупность программно-аппаратных средств, обеспечивающих ее конечным пользователям возможность решать определенные классы задач из их предметных

областей исследования, используя требуемые для этого вычислительные ресурсы среды. ГРВС обеспечивает гибкие языковые возможности и программные средства для построения моделей предметных областей, а также их использования при формулировке задач и управления вычислениями в процессе их решения.

1.2. Анализ подходов к построению моделей управления заданиями в распределенных вычислительных средах

Предметная область любой сферы деятельности может быть представлена в виде совокупности знаний, отражающих ее организационно-функциональную структуру. Формализованная спецификация таких знаний является моделью предметной области. Она, как правило, включает информацию об основных объектах предметной области и связей между ними, атрибутах объектов и их характеристиках, правилах и процессах функционирования объектов, событиях, фактах и явлениях, свойственных данной области, а также методах решения задач, возникающих в ней.

В диссертационной работе рассматриваются большие задачи, для решения которых специалисту в предметной области не хватает ресурсов доступных ему компьютеров: производительности процессоров, объемов оперативной и дисковой памяти, пропускной способности телекоммуникационной среды и других необходимых возможностей. Число элементарных операций, выполняемых вычислительной системой над каждой переменной в соответствии с алгоритмом решения ресурсоемкой задачи, как правило, существенно превышает быстродействие этих компьютеров.

В связи с этим возникает необходимость представления сценариев отображения методов решения задач на архитектуру вычислительной среды в модели предметной области. Здесь одной из важных проблем организации распределенных вычислений является концептуальное моделирование предметной области. Эта задача характерна как для областей исследований, связанных с инженерией знаний, создания баз данных и информационных систем, так и для предметно-ориентированных технологий программирования [273].

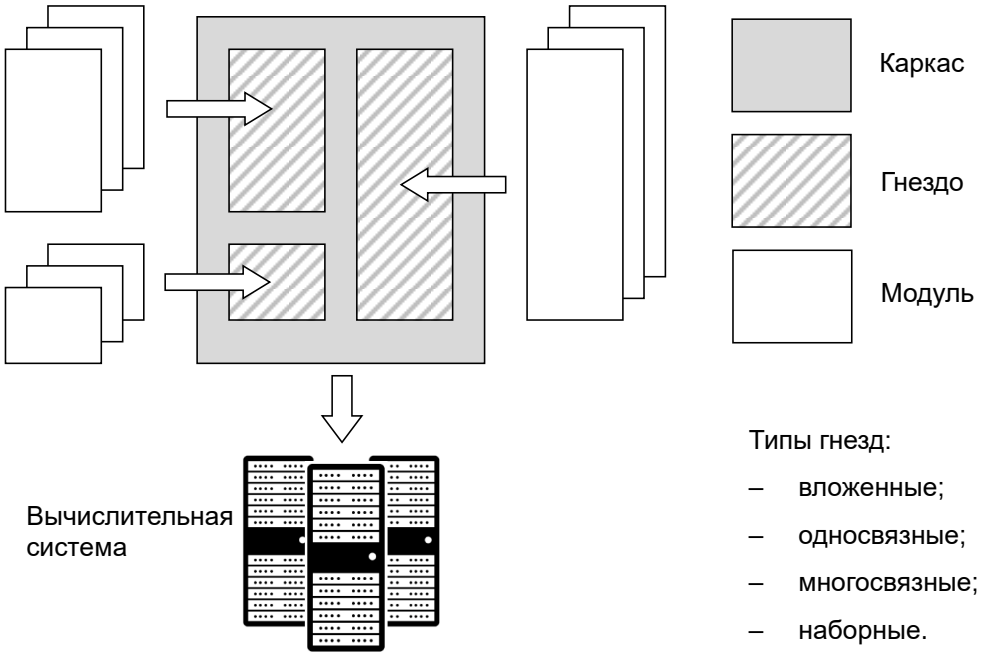
В рамках пакетного подхода [260] к разработке предметно-ориентированных программных систем (начало 70-х – конец 80-х гг. прошлого века), называемых пакетами прикладных программ (ППП), в качестве средств для концептуального моделирования предметной области использовались вычислительные модели различных видов [245, 292, 319], сети Петри [297], типизированные формальные теории [227], фреймы и функциональные семантические сети [401] и другие формы представления пакетных знаний [265]. Сочетание в пакете разнообразных сложных моделей, алгоритмов и методик их исследования достигается за счет модульной организации функционального наполнения пакета [261]. При этом знания о программно-аппаратной части вычислительной системы зачастую отделялись от предметных знаний и сосредотачивались, как правило, в его системной части [253].

В модели предметной области ППП вычислительный процесс представляется в виде схемы или плана решения задачи [318]. Схема решения задачи формируется на основе каркасного или цепочечного подходов. Каркас (управляющая программа) содержит гнезда для размещения сменных вычислительных модулей, которые могут варьироваться в различных комбинациях в зависимости от контекстных условий решаемых задач [252] (рисунок 1.3 а). В рамках цепочечного подхода осуществляется композиция сравнительно небольшого числа модулей в схему решения задачи, описывающую порядок их выполнения (рисунок 1.3 б).

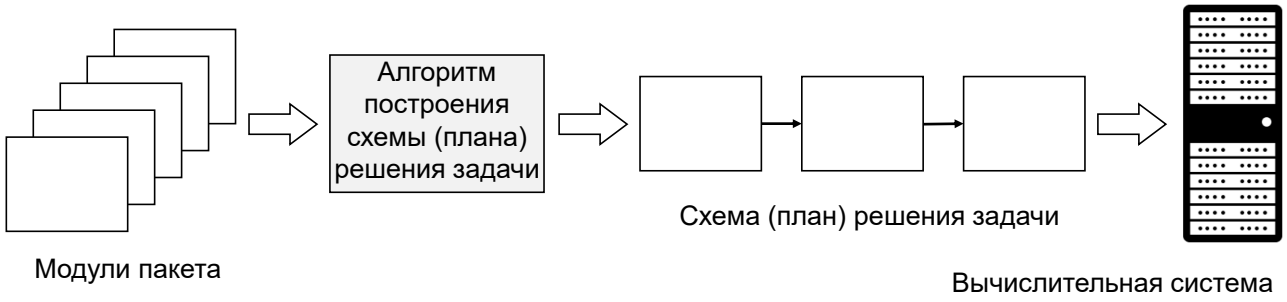
Выбор того или иного подхода к построению схем решения задач определяется спецификой этих задач. Как правило, каркасный подход используется для построения схем решения задач с неизменной структурой. В том случае, когда выбор схемы решения задачи зависит от ее постановки и требует сложного перебора модулей в процессе их композиции, целесообразно использовать цепочечный подход.

Близким направлением концептуальному моделированию является онтологическое моделирование, которое нашло широкое применение в рамках технологий программирования для семантической паутины [21]. В рамках онтологического моделирования основное внимание уделяется определению концептов, используемых в предметной области, и установлению семантических

взаимосвязей между ними.



а)



б)

Рисунок 1.3 – Подходы к построению схемы решения задачи: каркасный (а) и цепочечный (б)

В то же время результатом концептуального моделирования, как правило, является структура предметной области, включающая множества ее объектов (сущностей), семантические связи между их классами и ограничения на эти связи, а также алгоритмические элементы, отражающие процессы функционирования объектов. Таким образом, концептуальное и онтологическое моделирование, обладая сходными чертами, могут также дополнять друг друга [273].

Для поддержки онтологического моделирования используются такие языки

описания онтологий, как Ontology Web Language (OWL) [34] и Resource Description Framework (RDF) [44].

Семантическая паутина получила специализированное развитие в рамках организации грид-вычислений. Она ориентирована на обмен и объединение семантических данных из разных источников через сервисы с сохранением их смысла [18]. Семантические веб-службы основаны на применении стандартов для обмена семантическими данными. Семантическая Грид, подобно семантической паутине, решает схожие задачи применительно как к информационным, так и к вычислительным ресурсам в процессе их использования для решения ресурсоемких задач в ГРВС [46, 242]. В качестве языка для описания ресурсов Грид и заданий по обработке данных и выполнению вычислений задействуется широкий спектр расширений языка eXtensible Markup Language (XML) [104].

Развитие параллельных и распределенных вычислительных систем привело к появлению СУПЗ, а также связующего ПО, предназначенного для организации взаимодействия прикладного ПО с СУПЗ. Задание, определяющее процесс решения задачи и требования к ресурсам вычислительной системы, становится одним из основных понятий в области управления вычислениями.

В связи с этим активно стали развиваться модели распределенных вычислений. Данному направлению исследований посвящена обширная библиография (см., например, [30, 35, 36, 45, 57, 128, 139, 155, 177, 211, 221, 249, 270, 272, 277, 307, 316]. В этих работах основное внимание зачастую уделялось вопросам построения моделей распределения ресурсов и диспетчеризации заданий.

В последнее десятилетие актуализировались также исследования по управлению сервис-ориентированными и виртуализированными ресурсами ГРВС [29, 93, 124, 151, 158, 183, 237, 239, 246, 310, 402]. Известно успешное применение научных сервисов в вычислительной химии, компьютерной алгебре, статистическом анализе временных рядов (статистическом прогнозировании), биоинформатике, геоинформатике и во многих других областях. К настоящему времени разработан широкий спектр инструментов для построения сервис-ориентированных распределенных вычислительных сред. К их числу относятся,

например, средства общего назначения Amazon Elastic Compute Cloud [12], Google App Engine [98] и Microsoft Windows Azure [149], специализированные системы Globus [7] и Uniform Interface to Computing Resources (UNICORE) [193], язык программирования Opa [204]. Однако проблема учета специфики задач, решаемых в сервис-ориентированных средах, по-прежнему остается актуальной.

В ряде научных работ, связанных с исследованием процессов управления распределенными вычислениями, используются средства и методы явного описания специфики предметной области решаемых задач и схем их решения (см., например, [291, 303, 308]).

Схема решения задачи тесно коррелирует с понятием рабочего процесса (workflow), широко используемым в зарубежной литературе [17].

Выделяются два типа рабочих процессов: абстрактный и конкретный. Вычисления, описываемые абстрактным рабочим процессом, абстрагированы от конкретных ресурсов. Они могут выполняться с помощью различных программных приложений, реализующих одинаковые функции, и на любых ресурсах, подходящих по своим характеристикам требованиям, которые необходимы для решения задачи с помощью данного рабочего процесса.

Напротив, вычисления в рамках конкретного рабочего процесса должны быть проведены на предопределенных ресурсах. На практике, учитывая динамичный и изменчивый характер распределенных вычислительных сред, пользователям предпочтительно использовать абстрактные рабочие процессы, в которых конкретизация используемых приложений и ресурсов может быть осуществлена непосредственно перед или во время выполнения рабочего процесса в соответствии с текущим состоянием ресурсов среды.

Абстрактные рабочие процессы являются частным случаем схем решения задач, рассматриваемых в диссертационном исследовании. Как правило, абстрактный рабочий процесс состоит из набора взаимосвязанных элементарных структур, описывающих порядок обработки данных программными приложениями. Отдельные виды таких структур рассматриваются в [23, 110, 192]. Ниже приведено их обобщенное рассмотрение, в котором учтены дополнительные

специфические структуры, реализующие многовариантные расчеты с варьированием по алгоритмам и данным, а также избыточные вычисления.

Простейшая элементарная структура – это процесс (рисунок 1.4 а), в рамках которого приложение выполняет обработку поступающих ему на вход данных с целью получения некоторого результата их обработки (выходных данных).

Несколько таких процессов могут быть объединены последовательно для создания независимой последовательной цепочки вычислений, реализующей по сути конвейерную обработку данных, когда результаты вычислений предыдущего процесса поступают на вход следующему процессу. Данная структура продемонстрирована на рисунке 1.4 б.

На рисунках 1.4 в-д проиллюстрированы основные структуры управления данными: распределения, агрегирования и перераспределения. При распределении данных решаются, как правило, две задачи: создание наборов данных для разных алгоритмов решения задачи и генерация множества вариантов данных, обрабатываемых в дальнейшем разными экземплярами одной программы.

Несмотря на то, что распределение данных может занимать много времени, оно ведет к увеличению параллелизма на последующих этапах вычислений. Агрегирование данных осуществляется на основе результатов нескольких отдельных процессов по их готовности и потенциально может требовать больших временных затрат вычислительных ресурсов. Такое агрегирование может обуславливать сужение параллелизма вычислений. Данные, агрегированные на предыдущих этапах вычислений, могут быть перераспределены между различными процессами на следующих этапах, что способствует расширению параллелизма вычислительного процесса.

Структура на рисунке 1.4 е представляет процесс передачи данных «многие – многим», когда каждое из приложений на одном уровне передает данные каждому приложению на следующем уровне. Данная структура неявно реализует режим «fork/join», когда приложения на следующем уровне могут быть запущены только в том случае, если выполнены все приложения на предыдущем уровне. Избыточные вычисления, когда одни и те же данные могут быть вычислены

разными приложениями, реализуются структурой, изображенной на рисунке 1.4 ж.

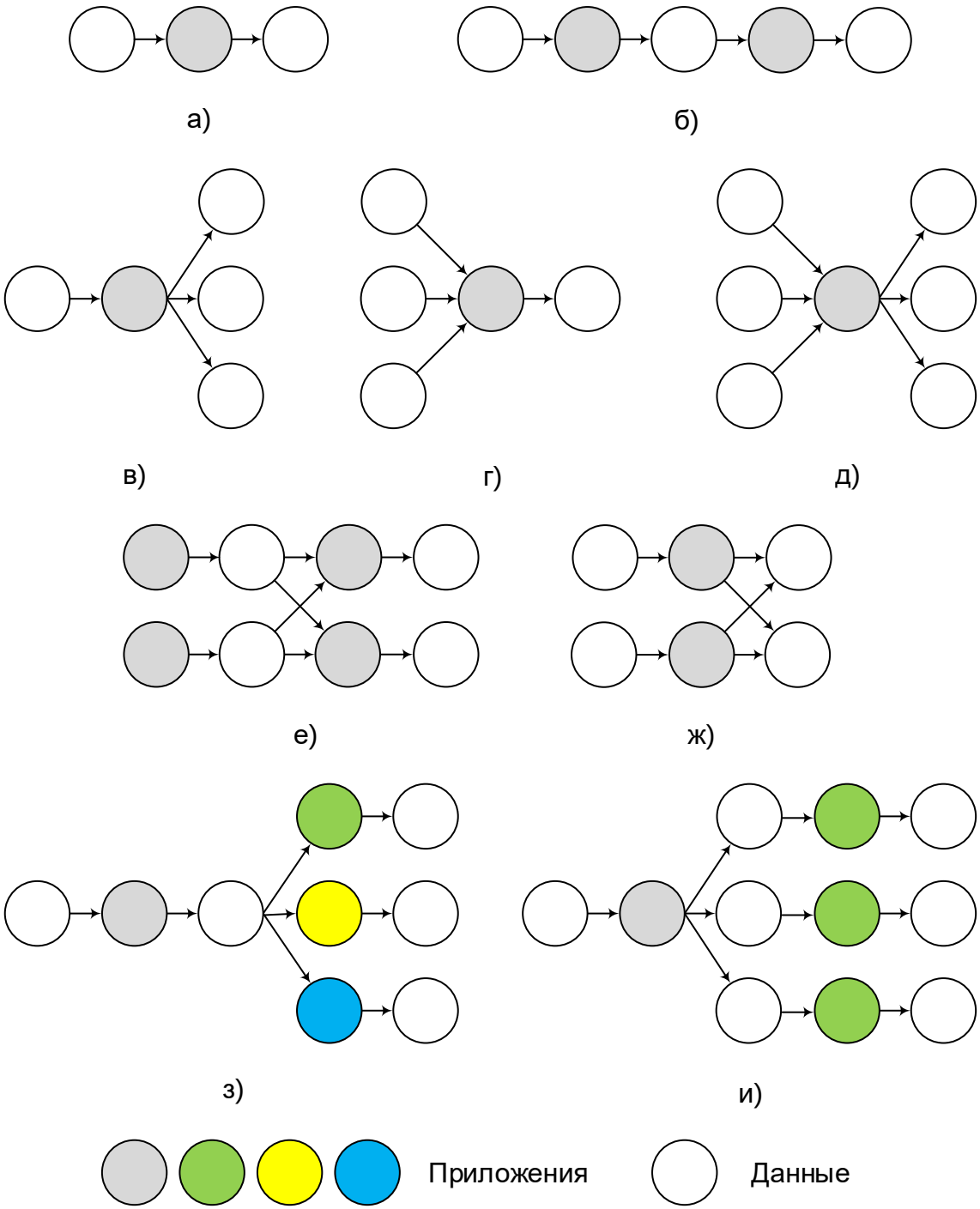


Рисунок 1.4 – Элементарные структуры рабочего процесса: процесс (а); последовательная цепочка вычислений (б); распределение (в), агрегирование (г) и перераспределение данных (д); передача данных «от каждого – каждому» (е); избыточные вычисления (ж); многовариантные расчеты с варьированием по алгоритмам (з) и данным (и)

На рисунках 1.4 з и 1.4 и представлены структуры, отражающие особенности проведения многовариантных расчетов с варьированием по алгоритмам и данным соответственно.

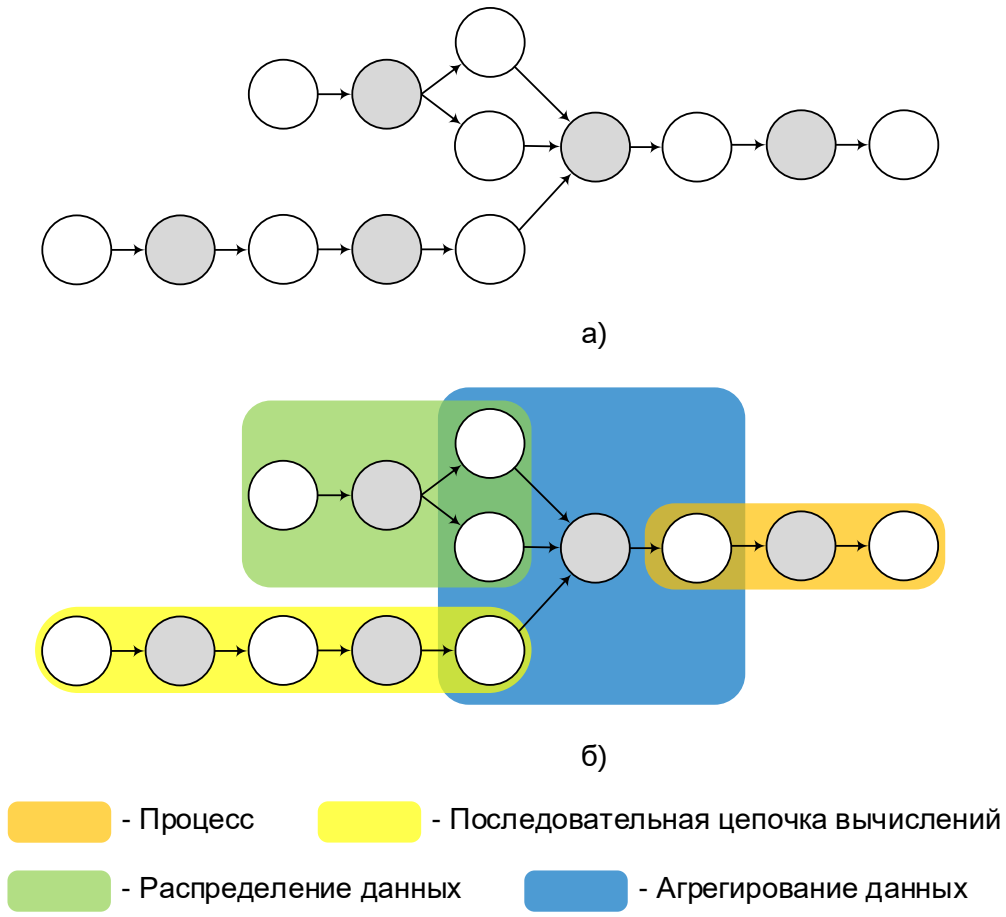


Рисунок 1.5 – Произвольная структура рабочего процесса в целом (а) и выделенные элементарные структуры обработки данных (б)

Пример рабочего процесса с произвольной структурой приведен на рисунке 1.5 а. В такой рабочий процесс могут быть включены различные структуры, рассмотренные выше (рисунок 1.5 б). Поддержка возможности дополнять и модифицировать функциональные возможности рабочих процессов, а также комбинировать их с другими рабочими процессами базируется на использовании так называемых micro-workflow [145]. В рамках пакетной проблематики такая возможность реализуется путем разработки распределенных модульных систем

программирования [252, 293].

Для организации и управления рабочими процессами применяются специализированные системы – Workflow Management Systems (WMS) [95]. Известен широкий спектр систем управления workflow [197, 222]. В их числе Askalon [59], Condor DAGMan, GridAnt [132], Grid Flow [33], Karajan [133], Kepler [9], PanDA, Pegasus [48], Taverna [160], Triana [202], UNICORE и другие системы.

Однако анализ функциональных возможностей вышеперечисленных WMS показывает, что проблемы формирования и обработки этими системами масштабируемых потоков заданий для ЦКП не решены в полной мере [201]. Ряд WMS (например, Grid Flow, Triana и UNICORE) используют собственное ПО или средства пакета GT (например, Ascalon, GridAnt и Pegasus), что неизбежно влечет дополнительные накладные расходы на управление вычислениями при организации их взаимодействия с СУПЗ, применяемыми в ЦКП. Для работы систем Condor DAGMan и Pegasus требуется установка системы HTCondor, редко используемая в суперкомпьютерных ЦКП. Системы Karajan, Kepler и Taverna имеют централизованную архитектуру, не поддерживающую динамическую декомпозицию потоков заданий между ресурсами, что не позволяет в полной мере использовать ее преимущества в случае изменения состояния используемых ресурсов.

Возможности ряда WMS (например, Condor DAGMan, Grid Ant, Karajan и UNICORE) для описания предметной области решаемых задач сильно ограничены, а средства представления предметных знаний других систем (например, Taverna) ориентированы в основном на достаточно узкие классы задач. Ограниченное число WMS поддерживает автоматический синтез рабочих процессов. Отсутствие стандартизованной спецификации предметной области и рабочих процессов не позволяет комплексировать приложения, разработанные для WMS, по данным и управлению.

Системный анализ предметной области является наиболее трудной частью процесса создания информационно-вычислительных систем и обоснованно требует привлечения высокоуровневых средств его автоматизации. В

инструментальных средствах, базирующихся на технологии Computer-Aided Software Engineering (CASE), используют два принципиально разных подхода к проектированию для проектирования и поддержки ПО: структурный и объектно-ориентированный.

Одним из подходов к структурному анализу предметной области и описанию происходящих в ней процессов является применение технологии Structured Analysis and Design Technique (SADT), разработанная Д. Россом в конце 1960-х – начале 1970-х гг. Технология SADT [178] включает совокупность правил и процедур, предназначенных для построения модели функционирования объекта некоторой предметной области. Однако эта технология успешно работает только применительно к хорошо специфицированным и стандартизованным бизнес-процессам. Наличие различного рода неопределенностей существенно затрудняет ее использование.

Другой подход базируется на диаграммах потоков данных – Data Flow Diagrams (DFD) в рамках структурного анализа предметной области с целью отображения схемы преобразования информации процессами исследуемой системы и связей между этими процессами [266]. Диаграммы потоков данных хорошо подходят для моделирования информационных систем.

Распространенным подходом к моделированию данных предметной области является также применение модели «сущность-связь» – Entity-Relationship Model (ERM) [276], предложенной П. Ченом в 1976 г. Данная модель, как правило, задействуется в структурном анализе и проектировании для выделения некоторого подмножества объектов модели предметной области.

В рамках объектно-ориентированной парадигмы анализа и проектирования ПО также широко используется Unified Modeling Language (UML) [243] – язык для визуального определения, проектирования и документирования сложных систем для различных сфер человеческой деятельности путем построения UML-моделей. Так как UML не является языком программирования, необходима генерация программного кода по UML-модели с помощью специализированных инструментальных средств. В связи с этим возникает ряд проблем:

инструментальные средства зачастую могут быть привязаны к узкому классу задач; требуется синхронизация исходного кода с его визуальным представлением; язык является достаточно громоздким, что порой затрудняет понимание UML-моделей.

CASE-технология объединяет совокупность методов проектирования ПО, а также набор инструментальных средств, позволяющих в наглядной форме моделировать предметную область, анализировать эту модель на всех стадиях разработки и сопровождения ПО и разрабатывать приложения в соответствии с потребностями их пользователей. Выбор конкретных инструментальных средств и их объединение в единую технологическую цепочку для построения модели ГРВС, учитывающей особенности предметных областей решаемых задач, требует наличия высокой квалификации и широких навыков практического применения этих инструментальных средств не только у разработчика такой модели, но и специалистов в предметных областях, которые будут использовать данную модель.

Таким образом, требуются дополнительные методы и средства концептуального моделирования предметной области ГРВС, которые обеспечат максимальную простоту и удобство ее описания различными категориями пользователей среды. Еще одной специфической особенностью таких методов и средств является то, что модель ГРВС должна обеспечивать представление разнородных знаний как о предметных областях решаемых задач, так и характеристиках самой среды.

1.3. Основные понятия и структура агрегированной модели

Исходя из специфики предметно-ориентированной ГРВС, рассмотренной в разделе 1.1, в структуру АМ, разработанной в диссертации, включено три компонента знаний [80]:

- алгоритмические знания предметных областей решаемых задач;
- знания о программно-аппаратной инфраструктуре среды;
- экспертные знания администраторов среды.

Компоненты знаний и категории субъектов, формирующих эти знания, представлены на рисунке 1.6.

Рисунок 1.6 – Структура компонентов знания АМ

Алгоритмические знания в свою очередь включают прикладные и системные знания. Структура алгоритмических знаний предложена Г.А. Опариным для пакетов прикладных программ, создаваемых в рамках САТУРН-технологии [293], и в дальнейшем адаптирована автором диссертации для распределенных вычислительных сред [337, 346, 355].

Вычислительные знания содержат информацию о прикладных и системных модулях. Под модулем понимается выделенная по тем или иным мотивам часть ПО [253]. Спецификация модуля может включать следующую информацию: текст модуля, язык программирования, тип и назначение входных, выходных и транзитных параметров, методы передачи параметров и обработки нестандартных ситуаций, требуемый компилятор и его опции, формат вызова и другие

необходимые сведения.

Системные модули обеспечивают планирование вычислений, формирование заданий, прогнозирование времени выполнения заданий, распределение ресурсов, запуск, мониторинг и завершение вычислительных процессов, динамическую декомпозицию задач, препроцессорную и постпроцессорную обработку данных, а также другие операции по управлению вычислениями.

Схемные знания традиционно включают множество объектов для описания модульной структуры модели и алгоритмов для исследования предметной области. Эти знания свободны от деталей, касающихся программной реализации моделей и алгоритмов. Они лишь содержат ссылки на объекты вычислительного слоя знаний. Основными схемными объектами предметной области являются ее параметры и операции, как наиболее простые, адекватные и выразительные средства для описания структуры вычислительных хнаний.

Параметры представляют собой значимые показатели или свойства предметной области исследуемой системы, которые можно вычислить. Результатом вычисления параметра является его значение, относящееся к одному из допустимых типов данных АМ.

Над множеством параметров определены операции – отображения, ставящие в соответствие одному подмножеству параметров другое подмножество параметров. Операции моделируют процессы, происходящие в исследуемой системе. Они реализуются модулями. Таким образом, связи между схемным и вычислительным слоями знаний определяются следующим образом: операциям соответствуют модули, а параметры являются фактическими параметрами при вызове этих модулей.

В диссертационной работе схемные знания расширены системными параметрами и операциями для описания процессов, реализуемых системными модулями. Такое расширение обеспечивает принципиально новые возможности адаптации схем решения задач к текущему состоянию ГРВС путем мониторинга ресурсов среды, перепланирования вычислений, динамической декомпозиции решаемых задач, формирования новых заданий, прогнозирования их времени

выполнения и переназначения ресурсов с помощью системных модулей.

Продукционные знания позволяют пользователям в общем случае осуществлять многокритериальный выбор управляющих решений в текущей вычислительной ситуации. Используются продукции следующего вида [300]:

$$Pr\colon [C_{pre}];\ L \Rightarrow f;\ [C_{post}],$$

где

- Pr – имя продукции;
- C_{pre} – предусловие реализации ядра продукции, представленное логическим выражением (предикатом);
- $L \Rightarrow f$ – детерминированное ядро продукции;
- L – логическое выражение в левой части ядра продукции;
- f – операция, которую нужно выполнить, если L принимает значение «истина»;
- C_{post} – постусловие продукции, квадратные скобки $[\ldots]$ означают, что стоящее в них выражение может быть опущено.

Ядро продукции может быть реализовано в том случае, когда C_{pre} принимает значение «истина». Постусловие продукции определяет действия, которые нужно произвести после выполнения операции f. В случае наличия множества готовых к применению продукций их порядок формируется в соответствии с заданными им приоритетами.

Прикладные и системные алгоритмические знания закладываются в АМ соответственно разработчиками приложений в процессе их создания с помощью специализированных высокоуровневых инструментальных средств и администраторами ГРВС, участвующими в разработке приложений. На основе алгоритмических знаний в дальнейшем формулируются постановки задач (точные условия задач с описанием входной и выходной информации, наборов ограничений и критериев качества их решения) и строятся схемы их решения.

Формализованное представление алгоритмических знаний рассматривается в следующей главе. Примеры реализации входных языков для описания таких

знаний в пакетах прикладных программ приведены в пятой главе диссертации.

Знания о программно-аппаратной инфраструктуре представляют характеристики узлов, каналов связи, сетевых устройств, топологии сети и т.д. Эти характеристики включают в себя также статистическую информацию о сбоях ПО и аппаратных средств.

Знания об административных политиках включают данные о пользователях и их заданиях, правах доступа к ресурсам и квотах на их использование, а также системах управления ресурсами, их характеристиках и дисциплинах диспетчеризации вычислительных работ (рисунок 1.7).

Знания о программно-аппаратной инфраструктуре и административных политиках извлекаются с помощью различных информационно-вычислительных и управляющих систем, контрольно-измерительных приборов и систем метамониторинга. Эти знания отображаются в АМ с помощью системных параметров, операций и модулей, а также используются для моделирования обработки потоков заданий приложений в ГРВС [347, 388].

В таблице 1 представлены результаты сравнительного анализа возможностей моделей систем подержки рабочих процессов для представления различных компонентов знаний:

- вычислительных (1);
- схемных (2);
- продукционных (3);
- знаний о программно-аппаратной инфраструктуре (4);
- знаний об административных политиках (5);
- возможности включения в схему решения задачи операций, реализованных системными модулями (6).

Очевидно, что АМ позволяет более полно отразить знания, необходимые в процессе планирования вычислений и рсапределения ресурсов, по сравнению с системами управления рабочими процессами.

Таблица 1 – Результаты сравнительного анализа возможностей представления различных компонентов знаний

Модель ГРВС	Номер компонента знаний					
	1	2	3	4	5	6
Модель Askalon	+	+	–	–	–	–
Модель Condor DAGMan	+	+	–	+	+	–
Модель Grid Ant	+	+	–	–	–	–
Модель GridFlow	+	+	–	–	–	–
Модель Karajan	+	+	–	–	–	–
Модель Kepler	+	+	–	–	–	–
Модель Pegasus	+	+	–	–	–	–
Модель Taverna	+	+	–	–	–	–
Модель Triana	+	+	–	–	–	–
Модель UNICORE	+	+	–	+	+	–
Модель GridWay	+	+	–	+	+	–
Модель PanDA	+	+	–	+	+	+
Агрегированная модель	+	+	+	+	+	+

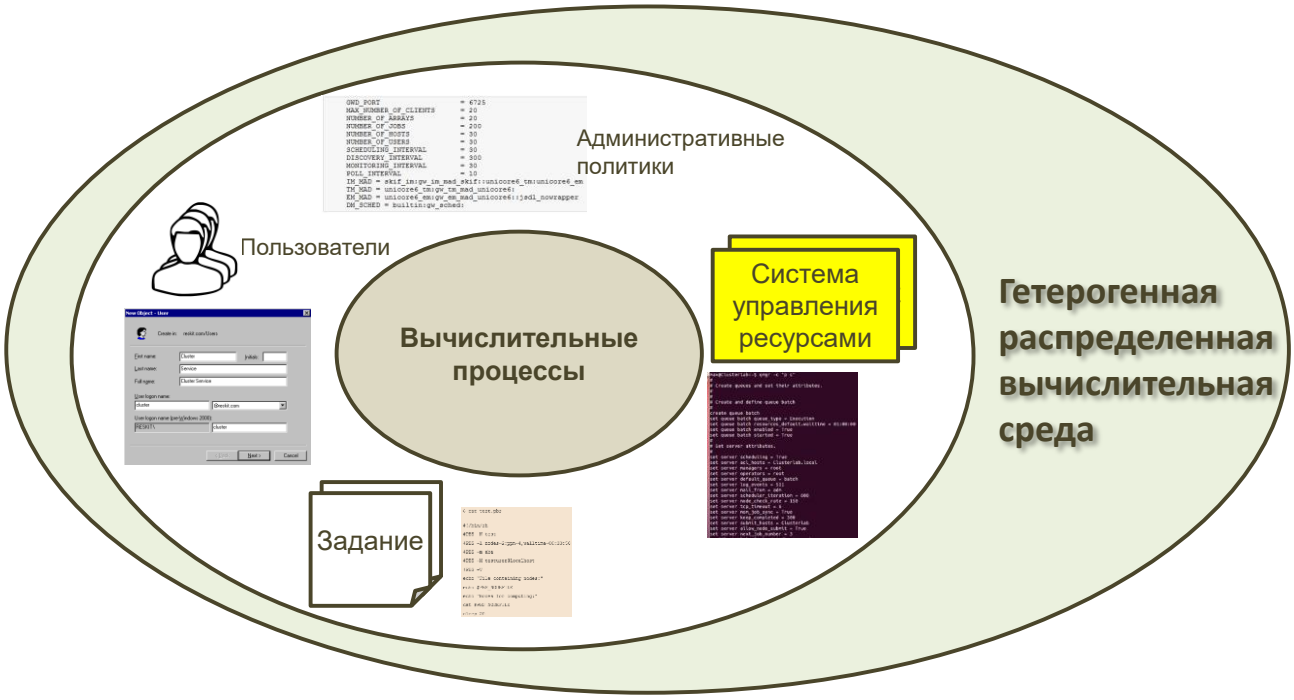


Рисунок 1.7 – Знания об административных политиках

1.4. Ядро агрегированной модели

В качестве ядра АМ предметно-ориентированной ГРВС для пакетов прикладных программ, созданных с помощью представленных в диссертационной работе инструментальных средств, выступают следующие объекты, перечисленные с указанием их основных атрибутов:

- параметры (имя, тип, область допустимых значений);
- операции (имя, тип, список входных параметров, список выходных параметров, программный модуль);
- формальные параметры (тип, назначение);
- программные модули (имя, список формальных параметров);
- логические выражения (имя, список логических параметров);
- продукции (имя, логическое выражение для предусловия выполнения ядра продукции, логическое выражение в левой части ядра продукции, имя операции, имя системной операции, реализующей постусловие, приоритет продукции);
- постановки задач (имя, список исходных параметров, список целевых параметров, список операций, список критериев качества решения задачи; список ресурсов; список критериев эффективности использования ресурсов);
- схемы решения задач (имя, список исходных параметров, список целевых параметров, список операций, список критериев качества решения задачи, список ресурсов, список критериев эффективности использования ресурсов, список ограничений);
- характеристики заданий (имя, тип, область допустимых значений, ранг, вес);
- задания (имя, имя схемы решения задачи, список характеристик задания);
- классы заданий (имя, список характеристик заданий);
- характеристики вычислительного устройства (имя, тип, область допустимых значений);
- вычислительные устройства (имя, список характеристик вычислительного устройства);

– характеристики канала связи (имя, список характеристик канала связи);
– каналы связи (имя, тип, список характеристик канала связи);
– характеристики сетевого устройства (имя, тип, область допустимых значений);
– сетевые устройства (имя, список характеристик сетевого устройства);
– пользователи (имя, логин, пароль, группа пользователей).

Ядро АМ является фундаментальной основой для построения дополнительных моделей определения критериев качества выполнения заданий, классификации заданий, планирования вычислений, распределения ресурсов и других процессов, связанных с управлением вычислениями (рисунок 1.8). Эти дополнительные модели реализуются в виде системных модулей и используются в инструментальных комплексах для разработки параллельных и распределенных приложений.

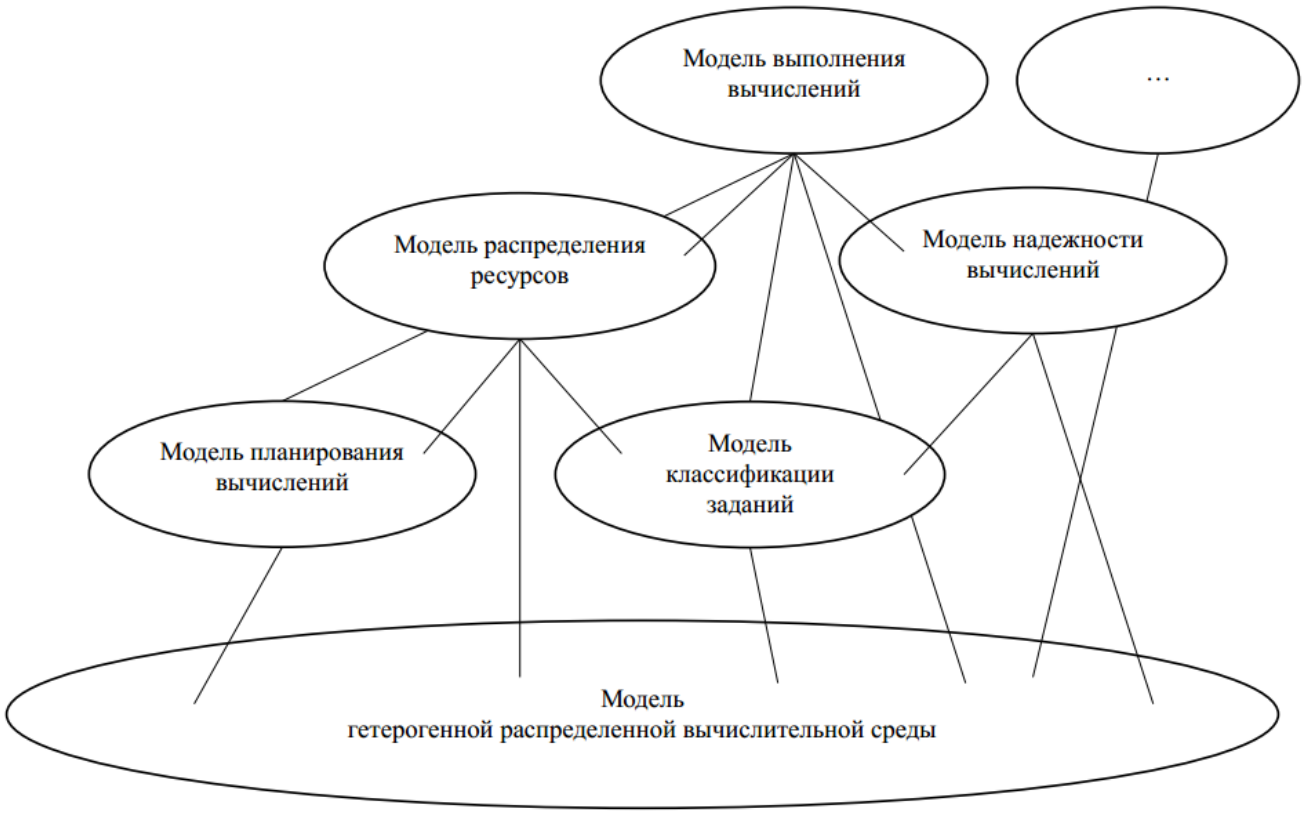


Рисунок 1.8 – Модели управления вычислениями

Для декларативного описания вычислительных моделей, формируемых на

базе ядра АМ, в инстументальных комплексах разработки РППП, рассматриваемых в пятой главе, применяются различные расширения языка XML, представленные словарями элементов, представляющих объекты модели, и их атрибутов, а также наборами правил, определяющих какие элементы и атрибуты могут входить в состав других элементов. Поддержка целостности вычислительных моделей и запросов к ним обеспечивается путем применения формата XML Schema [220].

В то же время, для передачи сложно структурированных данных в распределенной среде используется формат JSON [191], т. к. он во многих случаях облегчает и ускоряет процесс обмена данными по сравнению с XML. Валидация структур передаваемых данных осуществляется с помощью формата JSON Schema [119].

1.5. Выводы

АМ, предложенная в данной главе, является развитием интеллектных методов применения высокоуровневого ПО для интеграции распределенных информационно-вычислительных и коммуникационных ресурсов.

В первой главе получены следующие основные результаты:

- определены ключевые особенности предметно-ориентированной ГРВС, оказывающие существенное влияние на процесс управления потоками заданий в такой среде;
- выполнен анализ подходов к построению моделей обработки заданий в распределенных вычислительных средах, наиболее близкими из которых являются методы и средства описания моделей в пакетах прикладных программ и системах управления рабочими процессами;
- на основе результатов проведенного анализа предложена АМ для представления знаний о предметно-ориентированной ГРВС, являющаяся фундаментальной основой для построения дополнительных моделей управления заданиями в такой среде;

– проведен сравнительный анализ возможностей отображения рассмотренных компонентов знаний в АМ, а также в моделях известных метапланировщиков и систем управления рабочими процессами.

Результаты исследований по данной главе опубликованы в [70, 80, 337, 346, 347, 355, 388, 392].

Глава 2. Постановка задачи управления вычислениями гетерогенной распределенной вычислительной среды

Управление вычислениями традиционно включает в себя две основные составляющие этого процесса: планирование вычислений и распределение ресурсов [316]. В процессе управления необходимо учитывать предпочтения владельцев ресурсов и их пользователей, а также существующие административные политики в узлах ГРВС и самой среде в целом.

В данной главе рассматривается фрагмент АМ ГРВС (вычислительная модель), используемый для формулировки постановок задач, построения схем их решения, определения критериев качества выполнения вычислительных процессов и разработки механизмов многокритериального выбора управляющих воздействий. Формулируется постановка задачи планирования вычислений и распределения ресурсов на этой вычислительной модели. Вводятся основные критерии качества (время выполнения модулей и схем решения задач, надежность и стоимость выполнения этих схем, а также показатели эффективности использования ресурсов) вычислительных процессов, отражающих предпочтения владельцев ресурсов и их пользователей. Разрабатываются модели определения этих критериев. Предлагается реализация многокритериального выбора управленческих решений. В конце главы кратко подытоживаются полученные в ней научные результаты.

2.1. Вычислительная модель

Вычислительная модель (фрагмент АМ ГРВС) определяется структурой

$$\mathbb{M} =< Z, F, M, PR, S, J, R, A, Q, PLCS, MH, O >,$$

где $Z, F, M, S, J, R, A, Q, O$ – это соответственно множества параметров, операций, программных модулей, продукций, схем решения задач, заданий, ресурсов, агентов, ограничений на выполнение заданий и использование ресурсов, накладываемых множеством $PLCS$ административных политик, а также отношений между перечисленными объектами. Структура данных MH отражает

вычислительную историю работы модулей из M. Формализацию различных компонентов модели $\mathbb{M}$, являющейся развитием модели, предложенной в [291], можно найти в [80, 82, 334, 349, 352, 378, 391]. Их обобщение приведено ниже.

Операции из F определяют отношения между параметрами из множества Z. Параметры могут быть представлены скалярами, векторами, матрицами различных базовых типов данных или произвольными структурами данных.

Каждая операция $f_i \in F$ реализуется модулем $m_j \in M$, где $i \in \overline{1, n_f}$, $j \in \overline{1, n_m}$, n_f – число операций, n_m – число модулей. Один модуль может реализовать несколько операций. С каждой операцией f_i связано два подмножества параметров $Z_i^{in}, Z_i^{out} \subset Z$. Подмножество Z_i^{in} включает параметры, значения которых необходимо задать, чтобы получить значения параметров из подмножества Z_i^{out}. Параметры подмножеств Z_i^{in} и Z_i^{out} отражают назначение и семантику формальных параметров модуля m_j, реализующего операцию f_i. Параметры передаются между модулями в виде файлов.

Пример взаимосвязи операции и ее параметров представлен на рисунке 2.1.

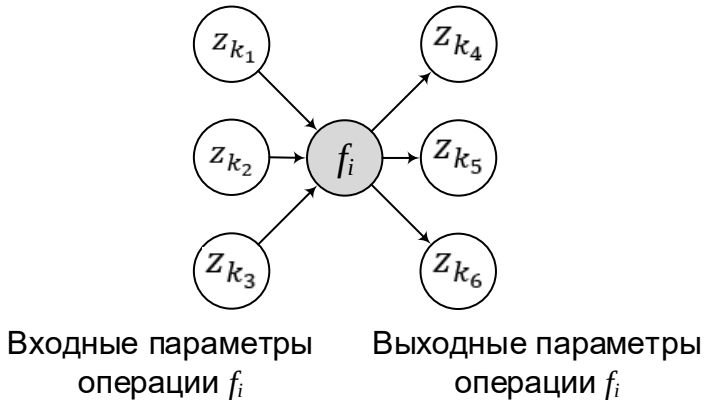
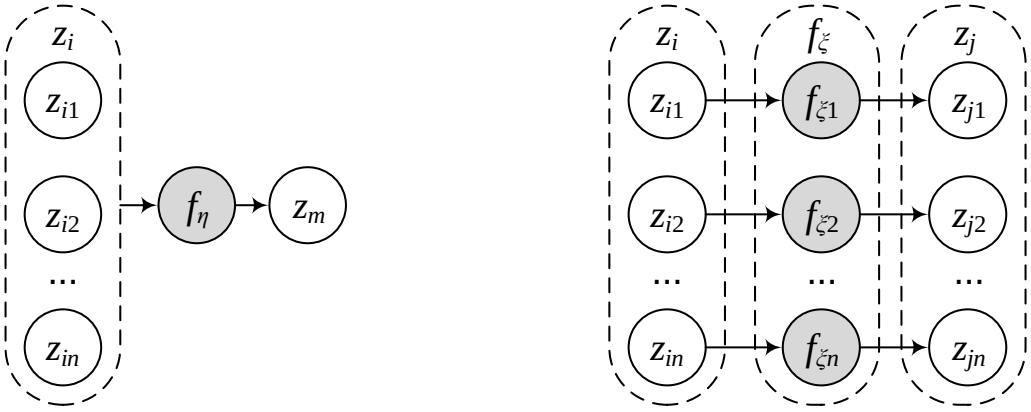


Рисунок 2.1 – Операция f_i над полем параметров $z_{k_1}, z_{k_2}, z_{k_3} \in Z_i^{in}$ и $z_{k_4}, z_{k_5}, z_{k_6} \in Z_i^{out}$

С целью поддержки в вычислительной модели параллелизма по данным в множестве параметров выделяются специальные структуры данных параметры-списки [378]. Параметр-список создается на основе одного параметра любого вида

(скаляра, вектора или матрицы) и включает множество вариантов значений этого параметра. Число элементов параметра-списка задается параметром-скаляром целого типа. Основное отличие параметра-списка от параметра-массива заключается в способе обработки элементов параметров таких типов. Параметр-массив целиком передается в вычислительный модуль, в котором далее осуществляется его обработка, в то время как параметр-список может обрабатываться поэлементно (в общем случае параллельно) множеством экземпляров вычислительного модуля на разных узлах.

Пусть z_i и z_j – параметры-списки, созданные на основе параметров z_k и z_l, с числом элементов $n = z_e$. Операции f_η и f_ξ предназначены соответственно для последовательной и параллельной обработки параметра z_i. Интерпретация операции f_η заключается в агрегировании данных, представленных параметром z_i (рисунок 2.2 а). Интерпретация операции f_ξ выполняется следующим образом (рисунок 2.2 б): осуществляется параллельный запуск n экземпляров операции f_ξ, r-му экземпляру операции $f_{\xi r}$ передается r-й элемент параметра z_i; результат присваивается r-му элементу параметра z_j.



а) б)

Рисунок 2.2 – Интерпретация операций f_η (а) и f_ξ (б)

Операцию f_ξ будем называть параллельной операцией.

Схемные знания об алгоритмах исследования предметной области

представляются в вычислительной модели двудольным ориентированным ациклическим графом, который включает только два типа вершин (параметры и операции) и два типа дуг между этими вершинами: входная дуга соединяет вершину первого типа (входной параметр) с вершиной второго типа (операцию), выходная дуга соединяет вершину второго типа (операцию) с вершиной первого типа (выходным параметром).

В множествах Z, F и M выделены подмножества системных объектов $Z^s \subset Z$, $F^s \subset F$ и $M^s \subset M$. Такие объекты создаются в вычислительной модели администраторами ГРВС.

Модули из M функционируют на ресурсах из R, которые представлены в ГРВС агентами из A. Один агент может представлять несколько ресурсов, которые функционируют под управлением установленных в них СУПЗ. Агенты могут оценивать состояние загрузки ресурсов и в зависимости от этого состояния выбирать и добавлять в очереди СУПЗ задания, поступающие в ГРВС, и осуществлять их дальнейшую обработку (контролировать статус выполнения заданий, а также приостанавливать, возобновлять, перемещать и завершать задания, резервировать ресурсы и производить их мониторинг).

Продукции из PR определяют условия применения операций.

Процессы решения задач представлены схемами из S, которые строятся на вычислительной модели с использованием методов планирования вычислений и информационного планирования.

Схема $s \in S$ выполнения операций из F является аналогом строгой параллельной формы графа алгоритма, а ее построение сходно с построением данной формы [248]. Множество вершин, получивших одинаковый индекс в соответствии с процедурой построения схемы, называется ярусом схемы. Число вершин с одинаковым индексом – шириной яруса. Число ярусов схемы называется ее высотой, а максимальное число вершин ярусов – шириной схемы. Примеры схем решения задач в виде ярусно-параллельных форм представлены на рисунке 2.3.

Схему $s \in S$ будем представлять в модели следующим образом:

$$s\colon [f_{i_1}|f_{i_2}|\dots] \to [f_{j_1}|f_{j_2}|\dots] \to \dots \to [f_{l_1}|f_{l_2}|\dots],$$

где квадратными скобками […] обозначены ярусы схемы, а символы ‘→’ и ‘|’ определяют последовательность и параллельность выполнения операций.

Обработка схемы решения задачи осуществляется в асинхронном режиме по готовности данных для ее операций или в режиме fork/join. В асинхронном режиме операция на $(i+1)$-м ярусе может быть выполнена сразу после того, как в результате выполнения любых операций на предыдущих ярусах с номерами $1,2,\dots,i$ вычислены значения всех ее входных параметров. В режиме fork/join операция на $(i+1)$-м ярусе может быть выполнена, когда выполнены все операции на i-м ярусе.

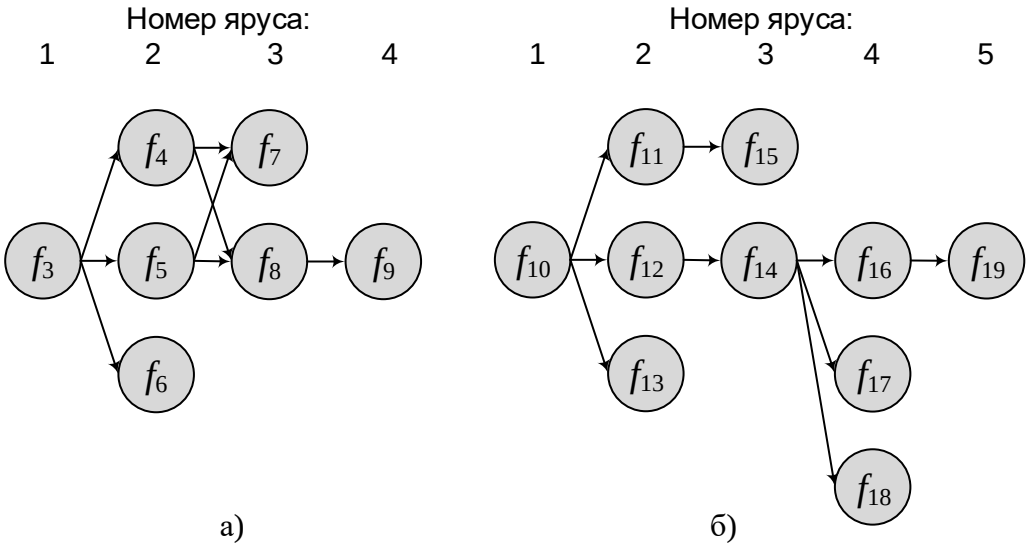


Рисунок 2.3 – Схемы решения задачи в ярусно-параллельной форме: схема решения задачи с высотой и шириной, равными соответственно 4 и 3 (а), схема решения задачи с высотой и шириной, равными соответственно 5 и 4 (б)

В рамках рассматриваемой вычислительной модели схема решения задачи является связным подграфом ориентированного ациклического двудольного графа, представляющего схемные знания об алгоритмах исследования предметной области. На рисунке 2.4 изображен двудольный ориентированный граф для схемы решения задачи, представленной на рисунке 2.3 а. Операции и параметры изображены на нем соответственно закрашенными и незакрашенными кружками.

С целью поддержки общности планирования вычислений при построении схем решения задач с использованием их постановок в вычислительной модели выделяются две специальные операции f_1 и f_2 [290], моделирующие в дальнейшем условия задачи, подмножество параметров, значения которых заданы, и подмножество параметров, значения которых нужно вычислить. Операция f_1 определяет подмножества $Z_1^{in} = \emptyset$ и $Z_1^{out} \neq \emptyset$, а операция f_2 обуславливает подмножества $Z_2^{in} \neq \emptyset$ и $Z_2^{out} = \emptyset$. В приведенном выше примере $Z_1^{in} = \emptyset$ и $Z_1^{out} = \{z_1\}$, а $Z_2^{in} = \{z_7, z_9, z_{10}\}$ и $Z_2^{out} = \emptyset$.

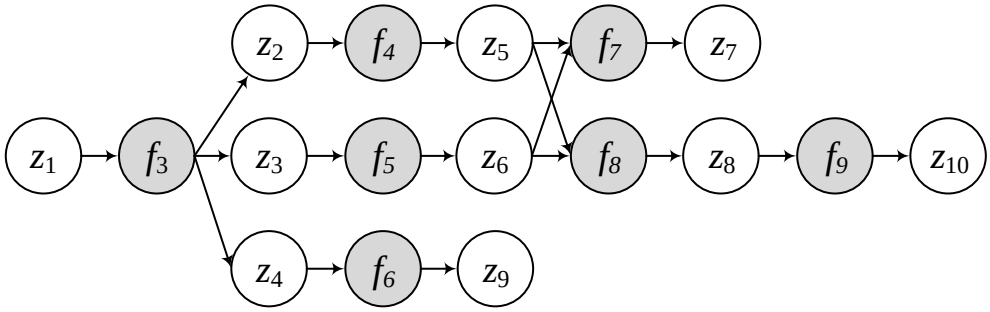


Рисунок 2.4 – Двудольный ориентированный граф схемы решения задачи

Требования к вычислительной среде по выполнению схем решения задач оформляются в виде заданий из J. Множество Q включает ограничения и квоты на выполнение заданий пользователей ГРВС и использование ими ее ресурсов. Эти ограничения и квоты определяются административными политиками, применяемыми в узлах ГРВС.

Формализованное описание вычислительной модели с помощью расширенной формы Бэкуса-Наура [218] приведено в Приложении Б.

2.2. Постановка задачи управления вычислениями

Модель $\mathbb{M}$ позволяет сформулировать многокритериальную постановку задачи управления распределенными вычислениями: «вычислить $Y = \{y_1, y_2, \dots, y_{n\_out}\}$ по $X = \{x_1, x_2, \dots, x_{n\_in}\}$, выполнив $F = \{f_1, f_2, \dots, f_{n\_f}\}$ с

критериями качества решения задачи $C^u = \{c_1^u, c_2^u, \dots, c_{n\_cu}^u\}$, указанными пользователем ГРВС, на ресурсах $R = \{r_1, r_2, \dots, r_{n\_r}\}$ с критериями качества их использования $C^o = \{c_1^o, c_2^o, \dots, c_{n\_co}^o\}$, заданными владельцами этих ресурсов, и учетом $Q = \{q, q_2, \dots, q_{n\_q}\}$».

В общем случае критерии $c_1^u, c_2^u, \dots, c_{n\_cu}^u$ и $c_1^o, c_2^o, \dots, c_{n\_co}^o$ представляют собой субъективные пожелания пользователей и владельцев ресурсов и могут накладывать противоречивые ограничения на схему решения задачи. В качестве критериев качества решения задачи в диссертации рассматриваются стоимость, время и надежность схемы решения задач. К критериям качества использования ресурсов относятся эффективность их использования, средняя загрузка, а также степень ее балансировки. Предложены модели определения качества решения задачи на основе сочетания использования экспериментальных данных и предсказательного моделирования. Как показывает практика, такой подход позволяет получать хорошие результаты в процессе распределения вычислительных ресурсов [65]. Показатели качества использования ресурсов рассчитываются традиционными методами [248].

Наличие неопределенностей различного рода в условиях задачи (в множествах Y, X, F, C^u, C^o, R и Q) приводит к многообразию вариантов ее постановки. Программные комплексы, использующие вычислительную модель $\mathbb{M}$, должны включать средства устранения таких неопределенностей методами планирования вычислений и распределения ресурсов в статическом или динамическом режимах.

Будем различать процедурные и непроцедурные постановки задачи. Процедурная постановка задачи означает, что множество F задано в ней, а множества X и Y не определены. При непроцедурной постановке задачи множества X и Y заданы, а множество F требуется установить. На основе процедурной или непроцедурной постановки задачи, сформулированной на модели $\mathbb{M}$, агент пользователя выполняет построение схемы s решения задачи.

Схему решения задачи будем представлять в виде булевой матрицы **S**

размерности $k \times n$, где k – число ярусов плана, а n – число операций в F. Элемент матрицы $s_{ij} = 1$ означает, что операция f_j размещена на i-м ярусе схемы.

В случае процедурной постановки задачи пользовательский агент производит информационное планирование с целью формирования множеств исходных (значения которых известны) и целевых (значения которых нужно вычислить) параметров.

Пусть на основе процедурной постановки задачи задана матрица **S** размерности $k \times n$. $A^* = \emptyset$, $B^* = \emptyset$, $A^{**} = \emptyset$ и $B^{**} = \emptyset$ – вспомогательные множества. Требуется определить Z_1^{in}, Z_1^{out}, Z_2^{in} и Z_2^{out}. Основные этапы алгоритма А.1, реализующего процесс информационного планирования, приведены ниже.

А.1.1. **for** $j = \overline{1,n}$ **step** 1 **do**

if $(s_{2j} = 1)$ **then** $A^* = A^* \cup Z_j^{in}; B^* = B^* \cup Z_j^{out};$ **endif**;

enddo;

А.1.2. **for** $i = \overline{3,k-1}$ **step** 1 **do**

for $j = \overline{1,n}$ **step** 1 **do**

if $(s_{ij} = 1)$ **then** $A^{**} = A^{**} \cup Z_j^{in}; B^{**} = B^{**} \cup Z_j^{out}$ **endif**;

enddo;

$A^* = A^* \cup A^{**} \cap \overline{B^*}; B^* = B^* \cup B^{**};$

enddo;

А.1.3. $Z_1^{in} = \emptyset; Z_1^{out} = A^*; Z_2^{in} = B^*; Z_2^{out} = \emptyset;$

А.1.4. **end**.

При непроцедурной постановке задачи применяется алгоритм А.2 построения ее схемы решения.

Введем булевы переменные s_crt и f_add. Переменная $s\_crt = 1$ *(*$s\_crt = 0$*)* означает, что схема решения построена (не построена). Переменная $f\_add = 1$ *(*$f\_add = 0$*)* означает, что на очередной итерации просмотра множества операций

хотя бы одна операция была включена в схему (ни одна операция не была включена в схему). $F^* = F$, $A^* = \emptyset$ и $B^* = \emptyset$ – вспомогательные множества.

Основные этапы алгоритма А.2 приведены ниже.

/* Построение схемы */

А.2.1. $l = 1$;

А.2.2. **for** $j = \overline{1,n}: j \neq 2$ **step** 1 **do** $s_{lj} = 0$ **enddo**;

А.2.3. $s_{crt} = 0$; $f\_add = 0$;

А.2.4. **for** $j = \overline{1,n}: f_j^* \neq 0, f_j^* \in F^*$ **step** 1 **do**

if $\left(\left(Z_j^{in} = \emptyset\right) \vee \left(Z_j^{in} \subseteq A^*\right)\right)$ **then**

$s_{lj} = 1; f\_add = 1; B^* = B^* \cup Z_j^{out}; f_j^* = 0;$

if $(j = 2)$ **then** $s_{crt} = 1;$ **endif**;

endif;

enddo;

А.2.5. **if** $(s_{crt} \vee f\_add = 0)$ **then goto** $A.2.6;$ **endif**; /* задача неразрешима */

if $(\bar{s}_{crt} \vee f\_add = 0)$ **then goto** $A.2.6;$ /* схема построена */

else $A^* = A^* \cup B^*; B^* = \emptyset; l = l + 1;$

for $j = \overline{1,n}$ **step** 1 **do** $s_{lj} = 0;$ **enddo**;

$f\_add = 0;$ **goto** $A.2.4;$

endif;

/* Удаление ненужных вычислений */

А.2.6. **for** $j = \overline{1,n}: j \neq 2$ **step** 1 **do** $s_{kj} = 0$ **enddo**;

А.2.7. $A^* = \emptyset; B^* = Z_2^{in};$

А.2.8. **for** $l = \overline{k-1,1}$ **step** -1 **do**

for $j = \overline{1,n}$ **step** 1 **do**

if $\left(s_{lj} = 1\right)$ **then**

if $\left(z_j^{out} \cap B^* = \emptyset\right)$ **then** $s_{lj} = 0;$

else $B^* = B^* \backslash Z_j^{out}; A^* = A^* \cup Z_j^{in};$

endif;

endif;

$B^* = B^* \cup A^*; A^* = \emptyset;$

enddo;

A.2.9. **enddo**;

A.2.10. **end**.

В результате работы алгоритма А.2 может быть получена поливариантная схема решения задачи, включающая один или m сценариев решения задачи. Условием единственности схемы решения задачи является

$$Z_i^{out} \cap Z_j^{out} = \emptyset \; \forall i, j: s_{ei} = 1, s_{lj} = 1, i \neq j, e \in \overline{1,k}, l \in \overline{1,k}.$$

Если оно не выполняется, то построение всего множества $\mathbf{S}_1, \mathbf{S}_2, \ldots \mathbf{S}_m$ схем решения задачи из $\mathbf{S}$ осуществляется методами полного перебора (см., например, [291]). Проверка удовлетворения схем заданным критериям их выполнения и использования ресурсов, а также выбор рациональной схемы решения задачи осуществляется в рамках тендера вычислительных работ, рассматриваемого в четвертой главе.

2.3. Критерии качества выполнения заданий

Прогнозная оценка времени выполнения модуля. Время выполнения задания является одним из основных критериев пользователя. Данный критерий во многом определяется временем выполнения программы, указанной в задании.

Задача оценки времени выполнения программы на целевом вычислительном узле ГРВС, назначаемом заданию, возникает из-за наличия большого числа узлов среды, обладающих разными вычислительными характеристиками, и, как правило, отсутствия практической возможности исследовать данную программу на каждом из них в силу их постоянной загруженности. Таким образом, появляется необходимость оценки времени выполнения программы, используя другой (эталонный) узел.

Существуют различные способы оценки времени выполнения программы на основе статических, динамических и комбинированных методов анализа программ [216]. На практике хорошо зарекомендовал себя метод частотных характеристик [214, 397], базирующийся на использовании специальных инструментальных средств для динамического анализа программ. Алгоритмы динамического анализа программ, основанные на использовании данного метода, различаются наборами исследуемых программных и аппаратных характеристик, а также зависимостей между ними [6, 228].

Подавляющее число традиционных систем управления заданиями (PBS, SLURM, HTCondor, GridWay и др.), а также многие системы управления workflow (Condor DAGMan, GridFlow, Taverna, UNICORE и др.) базируется на использовании оценок времени выполнения программ, заданных пользователем. Это простой и очень гибкий подход. Однако, как показывает практика, погрешность таких оценок, как правило, очень большая. Поэтому в целях дополнительного улучшения оценок времени выполнения программ на узле вычислительной среды применяются три основных подхода [316]: статический, динамический и статико-динамический.

Применение методов и средств статического анализа кода (без реального выполнения программы) (см., например, [115]) в гетерогенных ГРВС сопряжено с большими временны́ми и людскими затратами, обусловленными необходимостью разработки программных комплексов, моделирующих процессы компиляции, ассемблирования и функционирования процессора целевого вычислительного узла для широкого спектра программ, написанных на различных языках программирования [316]. Кроме того, методы и средства статического анализа зачастую существенно ограничены специфическими классами программ (например, они зачастую не пригодны для исследования унаследованного или нетиражируемого ПО) и типами базовой программно-аппаратной архитектуры, доступными для такого исследования с точки зрения наличия всей необходимой информации о функционировании ее компонентов [302]. Таким образом, применение данного подхода в системах разработки РППП, предназначенных для

выполнения в ГРВС, является достаточно проблематичным.

В рамках динамического подхода к прогнозированию времени выполнения программы осуществляется ее профилирование с целью получения информации о временны́х затратах на операции, связанные с использованием различных компонентов узла, в котором выполняется данная программа.

Наиболее применимым на практике является метод профилирования программы на некотором эталонном узле с последующей интерполяцией полученных результатов относительно характеристик целевого узла. При использовании данного метода точность оценки времени выполнения программы во многом зависит от выбора коэффициентов масштабирования ускорения вычислений, обуславливаемых соотношениями характеристик эталонного и целевого узлов [115].

Метод оценки времени выполнения программы путем ее тестовых запусков на целевом узле вычислительной системы дает наиболее точные результаты [225]. Однако, учитывая загруженность ресурсов ГРВС, такой метод применим для отдельных классов программ с небольшим временем счета.

Статико-динамический анализ базируется на комбинированном применении элементов двух других видов анализа, рассмотренных выше. Следовательно, методам и средствам его реализации, как правило, свойственны недостатки статического анализа.

В настоящее время данному направлению исследований в России и за рубежом уделяется пристальное внимание в силу его чрезвычайной актуальности [10, 41, 101, 129, 203, 209]. Однако прогнозирование времени выполнения программ при разработке и выполнении РППП в гетерогенных средах по-прежнему остается нетривиальной проблемой.

В диссертационной работе разработана методика оценки времени выполнения программ в ГРВС, базирующаяся на применении метода частотных характеристик [352]. В случае многоядерной архитектуры процессора предполагается, что один экземпляр программы выполняется на одном ядре. Разработанная методика включает восемь основных этапов.

Этап I. Выявление набора вычислительных характеристик $c_{r,1}, c_{r,2}, \dots, c_{r,l}$ компонентов эталонного узла, влияющих на время выполнения программы.

Этап II. Определение параметров $h_1(x), h_2(x), \dots, h_m(x)$ процесса выполнения программы, отражающих вычислительную нагрузку на компоненты эталонного узла, возникающую при работе данной программы. Значение $h_j(x)$ зависит от объема входных данных x, обрабатываемых программой, $j \in \overline{1,m}$.

Этап III. Формирование зависимостей

$$t_{r,i}\left(c_{r,1}, c_{r,2}, \dots, c_{r,l}, h_1(x), h_2(x), \dots, h_m(x)\right)$$

между вкладом i-го компонента эталонного узла в общее время выполнения программы, характеристиками $c_{r,1}, c_{r,2}, \dots, c_{r,l}$ и параметрами $h_1(x), h_2(x), \dots, h_m(x)$, $i = \overline{1,n}$.

Этап IV. Нахождение оценки $\hat{T}_r(x)$ теоретического времени выполнения программы на эталонном узле

$$\hat{T}_r(x) = \sum_{i=1}^{n} t_{r,i}\left(c_{r,1}, c_{r,2}, \dots, c_{r,l}, h_1(x), h_2(x), \dots, h_m(x)\right).$$

Этап V. Динамический анализ процесса выполнения программы в эталонном узле. Определение значений параметров $h_1(x), h_2(x), \dots, h_m(x)$ и времени $T_r(x)$ выполнения программы

$$T_r(x) = \hat{T}_r(x) + \alpha, \tag{1}$$

где α – это коэффициент, отражающий погрешность оценки $\hat{T}_r(x)$ относительно значения $T_r(x)$.

Этап VI. Формирование набора вычислительных характеристик $c_{t,1}, c_{t,2}, \dots, c_{t,l}$ компонентов целевого узла, соответствующих набору характеристик $c_{r,1}, c_{r,2}, \dots, c_{r,l}$.

Этап VII. Установление зависимостей

$$\hat{t}_{t,i}\left(c_{t,1}, c_{t,2}, \dots, c_{t,l}, h_1(x), h_2(x), \dots, h_m(x)\right)$$

между оценкой вклада i-го компонента целевого узла в общую оценку времени выполнения программы, характеристиками $c_{t,1}, c_{t,2}, \dots, c_{t,l}$ и параметрами $h_1(x), h_2(x), \dots, h_m(x)$, $i = \overline{1,n}$.

Этап VIII. Оценка $\hat{T}_t(x)$ теоретического времени выполнения программы в целевом узле

$$\hat{T}_t(x) = \sum_{i=1}^{n} t_{t,i}\left(c_{t,1}, c_{t,2}, \dots, c_{t,l}, h_1(x), h_2(x), \dots, h_m(x)\right).$$

Этап IX. Итоговая оценка $T_t'(x)$ времени выполнения программы в целевом узле записывается в виде

$$\hat{T}_t'(x) = \hat{T}_t(x) + \alpha\frac{\hat{T}_r(x)}{\hat{T}_t(x)}. \quad (2)$$

Этап X. Оценка времени выполнения программы для переменного объема данных $x \in (x_0, x_k)$ осуществляется с помощью функции

$$\hat{T}_t^*(x) = f_t^*\left(x_0, x_k, x, \hat{T}_t'(x)\right),$$

где f_t^* – это некоторая интерполяционная функция, которая, как правило, индивидуально подбирается для каждого класса программ.

Пример. В качестве примера применения рассмотренной выше методики рассмотрим оценку времени выполнения программы для перемножения квадратных матриц размерности $n \times n$. Программа использует «наивный» алгоритм перемножения матриц, вычислительная сложность которого составляет $O(n^3)$.

Как правило, при оценке производительности узла выделяют следующие его основные компоненты [296]: процессор, кэш-память, модули оперативной памяти; жесткие диски; графическую подсистему и сетевые устройства. В данном примере основными компонентами эталонного узла, осуществляющими основной вклад в общее время выполнения программы, является процессор, оперативная и кэш-память, а также жесткий диск. В связи с этим в качестве характеристик данного

узла выбираются и определяются следующие показатели:

- частота $c_{r,1}$ процессора;
- число $c_{r,2}$ и $c_{r,3}$ целочисленных операций и операций с плавающей точкой выполняемых процессором в секунду;
- пропускная способность $c_{r,4}$ и латентность $c_{r,5}$ кэш-памяти первого уровня;
- пропускная способность $c_{r,6}$ и латентность $c_{r,7}$ кэш-памяти второго уровня;
- пропускная способность $c_{r,8}$ и латентность $c_{r,9}$ оперативной памяти;
- среднее время поиска дорожки на диске $c_{r,10}$;
- пропускные способности $c_{r,11}$ и $c_{r,12}$ диска при чтении и записи данных.

Частота $c_{r,1}$ процессора является важным параметром, отражающим его производительность в целом. Однако арифметические операции требуют использования разного числа тактов работы процессора. Поэтому в алгоритме используются параметры $c_{r,2}$ и $c_{r,3}$, показывающие число целочисленных операций и операций с плавающей точкой выполняемых процессором в секунду.

В соответствии с выбранными параметрами основные параметры процесса выполнения программы отражают следующие характеристики:

- число $h_1(x)$ инструкций, обрабатываемых процессором узла при выполнении программы;
- число $h_2(x)$ и $h_3(x)$ обращений к кэш-памяти первого и второго уровней;
- число $h_4(x)$ и $h_5(x)$ промахов обращений к кэш-памяти первого и второго уровня;
- доли $h_6(x)$ и $h_7(x)$ целочисленных операций и операций с плавающей точкой, выполняемых в программе;
- число $h_8(x)$ и $h_9(x)$ сессий чтения данных с диска и записи на диск;
- число $h_{10}(x)$ и $h_{11}(x)$ байтов данных, считанных с диска и записанных на диск.

Характеристики $h_1(x) - h_{11}(x)$ вычисляются при выполнении программы с определенным объемом данных x.

Время использования процессора $t_{r,1}(x)$, памяти $t_{r,2}(x)$ и жесткого диска

$t_{r,3}(x)$ эталонного узла вычисляется с помощью выражений

$$t_{r,1}(x) = \frac{h_1(x)h_6(x)}{c_{r,2}} + \frac{h_1(x)h_7(x)}{c_{r,3}}, \quad (3)$$

$$t_{r,2}(x) = t_{c1}(x) + t_{c2}(x) + t_m(x), \quad (4)$$

$$t_{r,3}(x) = c_{r,10}\big(h_8(x) + h_9(x)\big) + \frac{h_{10}(x)}{c_{r,11}} + \frac{h_{11}(x)}{c_{r,12}}, \quad (5)$$

где значения времени $t_{c1}(x)$, $t_{c2}(x)$ и $t_m(x)$ представляют соответственно время использования кэш-памяти первого и второго уровней, а также оперативной памяти. Поскольку количество данных, загруженных в кэш-память первого уровня, неизвестно, будем считать, что каждая инструкция требует, по крайней мере, два байта данных в качестве своих операндов. В этом случае значения $t_{c1}(x)$, $t_{c2}(x)$ и $t_m(x)$ вычисляются следующим образом:

$$t_{c1}(x) = 2\frac{h_2(x)}{c_{r,4}} + c_{r,5}h_2(x), \quad (6)$$

$$t_{c2}(x) = 2\frac{\big(h_3(x) + h_4(x)\big)}{c_{r,6}} + c_{r,7}\big(h_3(x) + h_4(x)\big), \quad (7)$$

$$t_m(x) = 2\frac{h_5(x)}{c_{r,8}} + c_{r,9}h_5(x). \quad (8)$$

Параметры $h_1(x) - h_{11}(x)$ и время $T_r(x)$ определяются в процессе динамического анализа программы с помощью профилировщика производительности [162].

Оценка $\hat{T}_r(x)$ теоретического времени выполнения программы определяется по формуле

$$\hat{T}_r(x) = t_{r,1}(x) + t_{r,2}(x) + t_{r,3}(x). \quad (9)$$

Коэффициент α находится из соотношения (1).

Далее определяются характеристики $c_{t,1} - c_{t,12}$ целевого узла, соответствующие аналогичным характеристикам $c_{r,1} - c_{r,12}$ эталонного узла.

В соответствии с (3)-(9) формируются зависимости (10)-(16) для вычисления

оценок $\hat{t}_{t,1}(x)$, $\hat{t}_{t,2}(x)$, $\hat{t}_{t,3}(x)$, $\hat{t}_{c1}(x)$, $\hat{t}_{c2}(x)$, $\hat{t}_m(x)$ и $\hat{T}_t(x)$:

$$\hat{t}_{t,1}(x) = \frac{h_1(x)\left(c_{t,2}h_6(x) + c_{t,3}h_7(x)\right)}{c_{t,1}}, \tag{10}$$

$$\hat{t}_{t,2}(x) = \hat{t}_{c1}(x) + \hat{t}_{c2}(x) + \hat{t}_m(x), \tag{11}$$

$$\hat{t}_{t,3}(x) = c_{t,10}\big(h_8(x) + h_9(x)\big) + \frac{h_{10}(x)}{c_{t,11}} + \frac{h_{11}(x)}{c_{t,12}}, \tag{12}$$

$$\hat{t}_{c1}(x) = \frac{\big(2 + c_{t,4}c_{t,5}\big)h_2(x)}{c_{t,4}}, \tag{13}$$

$$\hat{t}_{c2}(x) = 2\frac{h_3(x) + h_4(x)}{c_{t,6} + c_{t,7}\big(h_3(x) + h_4(x)\big)}, \tag{14}$$

$$\hat{t}_m(x) = 2\frac{h_5(x)}{c_{t,8} + c_{t,9}h_5(x)}, \tag{15}$$

$$\hat{T}_t(x) = \hat{t}_{t,1}(x) + \hat{t}_{t,2}(x) + \hat{t}_{t,3}(x). \tag{16}$$

Итоговая оценка $\hat{T}_t'(x)$ времени выполнения программы в целевом узле находится из соотношения (2).

При изменении значения x в интервале (x_0, x_k) оценка времени выполнения программы осуществляется следующим образом:

$$\hat{T}_t^*(x) = \hat{T}_t'(x_0)\left(\frac{x}{x_0}\right)^y,$$

$$y = log_b a, \;\; a = \frac{\hat{T}_t'(x_0)}{\hat{T}_r'(x_k)}, \;\; b = \frac{x_0}{x_k}.$$

Функция $\hat{T}_t^*(x)$ подобрана на основе экспериментальных данных для рассматриваемого класса программ. Она отражает нелинейную зависимость изменения времени выполнения программы от объема данных.

Характеристики $c_{r,1} - c_{r,12}$ и $c_{t,1} - c_{t,12}$ узлов, использованных в экспериментах, и показатели использования компонентов эталонного узла при решении задач перемножения целочисленных и вещественных матриц приведены в Приложении В (таблицы В.1-3).

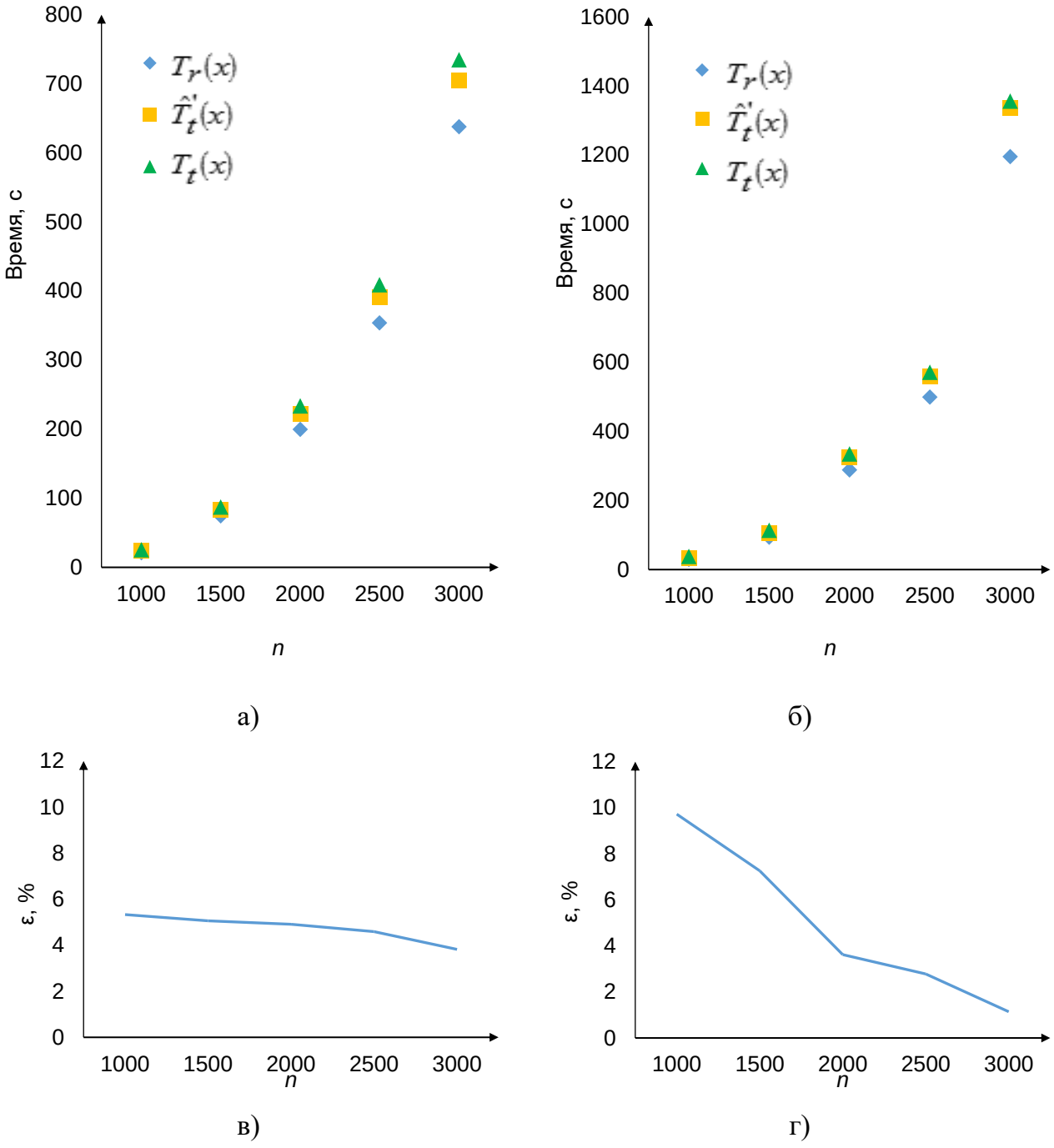


Рисунок 2.5 – Оценки времени выполнения программы перемножения целочисленных (а) и вещественных (б) матриц, а также погрешности полученных оценок для целочисленных (в) и вещественных (г) матриц

На рисунках 2.5 а и 2.5 б приведены значения времени $T_r(x)$ выполнения программы перемножения целочисленных и вещественных матриц в эталонном узле, а также оценки $\hat{T}_t'(x)$ и реального времени $T_t(x)$ ее выполнения в целевом узле для $n = 1000, 1500, 2000, 2500, 3000$. Погрешность ε полученных оценок

приведены на рисунках 2.5 в и 2.5 г.

Полученные результаты демонстрируют уменьшение погрешности оценки времени выполнения программы по мере увеличения размерности матриц как для целочисленных, так и для вещественных значений. Следует отметить, что для данного примера погрешность ε не превышает 10%.

Общая схема прогнозирования времени выполнения модуля с использованием профилировщика программ показана на рисунке 2.6. Вычислительные характеристики узлов кластеров ГРВС, являющихся основными компонентами инфраструктуры среды, приведены в Приложении В (таблица В.4).

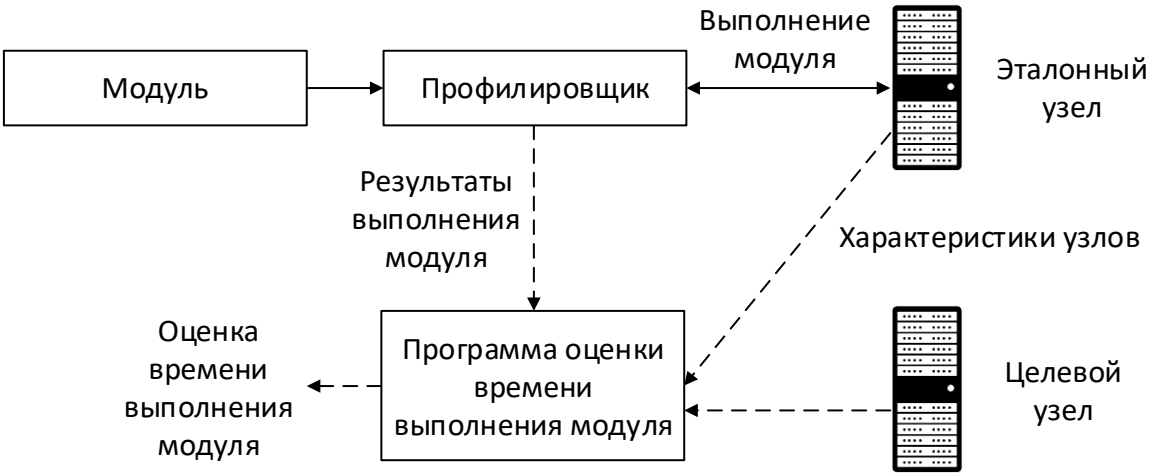


Рисунок 2.6 – Прогнозирование времени выполнения схемы решения задачи

Прогнозируемое время выполнения схемы решения задачи в асинхронном режиме по готовности данных. Прогнозирование времени выполнения схем решения задач (заданий) здесь и ниже базируются на подходе к оценке производительности вычислительных устройств, предложенном в [334].

Пусть матрица **P** размерности $k \times k$ представляет сведения о предшествовании k операций схемы решения задачи. Элемент $p_{ij} = 1$ ($p_{ij} = 0$) матрицы **P** означает, что выполнение операции f_j в схеме решения задачи предшествует (не предшествует) выполнению операции f_i и $Z_i^{in} \cap Z_j^{out} \neq \emptyset$. Матрица $\widehat{\mathbf{D}}$ размерности $k \times k$ отражает оценки объемов передаваемых данных

между операциями. Элемент матрицы $\hat{d}_{ij} \geq 0$ показывает объем данных, передаваемых операцией f_i операции f_j. Матрица $\mathbf{W}$ размерности $k \times k$ предоставляет информацию о пропускной способности интерконнекта между узлами, в которых запускаются модули, реализующие операции схемы решения задачи. Элемент матрицы $w_{ij} \geq 0$ демонстрирует пропускную способность интерконнекта между узлами, в которых работают модули, реализующие операции f_i и f_j. Тогда оценка $\hat{T}_s$ времени выполнения схемы решения задачи в асинхронном режиме по готовности данных определяется следующим образом:

$$\hat{T}_s = \max_{i=\overline{1,k}} \hat{\tau}_i ,$$

$$\hat{\tau}_i = \hat{q}_i + \hat{t}_i + \max_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \left(\hat{\tau}_j + \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right),$$

где $\hat{\tau}_i$ $(\hat{\tau}_j)$ – оценка времени, прошедшего с начала обработки схемы до завершения операции f_i (f_j), $\hat{q}_i$ – оценка времени нахождения в очереди модуля, реализующего операцию f_i (в то время, когда все данные, необходимые для его выполнения, готовы), $\hat{t}_i$ – прогнозная оценка времени выполнения этого модуля, k – число операций схемы решения задачи, $0 < \omega_{ji}(t) \leq 1$ – коэффициент снижения пропускной способности интерконнекта в момент времени t между узлами, в которых функционируют модули, реализующие операции f_i и f_j.

Прогнозируемое время выполнения схемы решения задачи в режиме fork/join. Пусть матрица $\mathbf{S}$ размерности $m \times k$ представляет собой ярусно-параллельную форму, описывающую обработку схемы решения задачи в режиме fork/join, m – число ярусов схемы. Элемент $s_{li} = 1$ означает, что операция f_i должна быть выполнена на *l*-м ярусе. Переход к операциям (*l*+1)-го яруса возможен при условии завершения всех операций на *l*-м ярусе. Тогда оценка $\hat{T}_s'$ времени выполнения схемы в режиме fork/join находится следующим образом:

$$\hat{T}_s' = \sum_{l=1}^{m} \hat{\tau}_l,$$

$$\hat{\tau}_l = \max_{\forall i \in \overline{1,k}: s_{li}=1} \left(\hat{q}_i + \hat{t}_i + \sum_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Прогнозируемое время выполнения схемы решения задачи в среде с виртуализированными ресурсами в асинхронном режиме. Оценка $\hat{T}_{vs}$ времени выполнения схемы решения задачи в среде с виртуализированными ресурсами в асинхронном режиме определяется следующим образом:

$$\hat{T}_{vs} = \max_{i=\overline{1,k}} \hat{\tau}_i ,$$

$$\hat{\tau}_i = \hat{q}_i + \hat{t}_i^{vm\_launch} + \hat{t}_i + \hat{t}_i^{vm\_remove} + \max_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \left(\hat{\tau}_j + \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right),$$

где $\hat{t}_i^{vm\_launch}$ и $\hat{t}_i^{vm\_remove}$ – это соответственно оценки времени на запуск и завершение ВМ для выполнения модуля, реализующего операцию f_i. Оценки $\hat{t}_i^{vm\_launch}$ и $\hat{t}_i^{vm\_remove}$ определяются экспериментальным путем для ВМ различной конфигурации.

Прогнозируемое время выполнения схемы решения задачи в среде с виртуализированными ресурсами в режиме fork/join. Оценка $\hat{T}'_{vs}$ времени выполнения схемы решения задачи в среде с виртуализированными ресурсами соответственно в режиме fork/join определяется следующим образом:

$$\hat{T}'_{vs} = \sum_{l=1}^{m} \hat{\tau}_l,$$

$$\hat{\tau}_l = \max_{\forall i \in \overline{1,k}: s_{li}=1} \left(\hat{q}_i + \hat{t}_i^{vm_{launch}} + \hat{t}_i + \hat{t}_i^{vm_{remove}} + \sum_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Прогнозируемое время выполнения схемы решения задачи в асинхронном режиме с рестартом модулей. Оценка $\hat{T}_{rs}$ времени выполнения схемы решения задачи в асинхронном режиме с рестартом модулей определяется следующим образом:

$$\hat{T}_{rs} = \max_{i=\overline{1,k}} \hat{\tau}_i ,$$

$$\hat{\tau}_i = \hat{q}_i + \hat{t}_i^{fault} + \hat{t}_i + \tilde{t}_i + \hat{t}_i^{rest} + \max_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \left(\hat{\tau}_j + \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right),$$

где $\hat{t}_i^{fault}$ и $\hat{t}_i^{rest}$ – это соответственно оценки времени обнаружения и идентификации отказа, а также повторного запуска (рестарта) модуля, реализующего операцию f_i, в случае программно-аппаратного отказа. Оценки $\hat{t}_i^{fault}$ и $\hat{t}_i^{rest}$ определяются средним временем выполнения таких процессов системой метамониторинга для разных видов программно-аппаратных отказов, $\tilde{t}_i$ – время выполнения модуля до отказа.

Прогнозируемое время выполнения схемы решения задачи в режиме fork/join с рестартом модулей. Оценка $\hat{T}'_{rs}$ времени выполнения схемы решения задачи в среде с виртуализированными ресурсами соответственно в режиме fork/join определяется следующим образом:

$$\hat{T}'_{rs} = \sum_{l=1}^{m} \hat{\tau}_l,$$

$$\hat{\tau}_l = \max_{\forall i \in \overline{1,k}: s_{li}=1} \left(\hat{q}_i + \hat{t}_i^{fault} + \hat{t}_i + \tilde{t}_i + \hat{t}_i^{rest} + \sum_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Прогнозируемое время выполнения схемы решения задачи в асинхронном режиме по готовности данных на основе пользовательских оценок времени выполнения модулей. Оценка $\hat{T}_{us}$ времени выполнения схемы решения задачи в асинхронном режиме по готовности данных на основе пользовательских оценок времени выполнения модулей определяется следующим образом:

$$\hat{T}_{us} = \max_{i=\overline{1,k}} \hat{\tau}_i \,,$$

$$\hat{\tau}_i = \hat{q}_i + \hat{t}'_i + \max_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \left(\hat{\tau}_j + \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right),$$

где $\hat{t}'_i$ – пользовательская оценка времени выполнения модуля.

Прогнозируемое время выполнения схемы решения задачи в режиме fork/join на основе пользовательских оценок времени выполнения модулей. Оценка $\hat{T}'_{us}$

времени выполнения схемы решения задачи в режиме fork/join определяется следующим образом:

$$\hat{T}'_{us} = \sum_{l=1}^{m} \hat{\tau}_l,$$

$$\hat{\tau}_l = \max_{\forall i \in \overline{1,k}: s_{li}=1} \left(\hat{q}_i + \hat{t}'_i + \sum_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Прогнозируемое временя выполнения схемы решения задачи в среде с виртуализированными ресурсами в асинхронном режиме на основе пользовательских оценок времени выполнения модулей. Оценка $\hat{T}_{uvs}$ времени выполнения схемы решения задачи в среде с виртуализированными ресурсами в асинхронном режиме определяется следующим образом:

$$\hat{T}_{uvs} = \max_{i=\overline{1,k}} \hat{\tau}_i \,,$$

$$\hat{\tau}_i = \hat{q}_i + \hat{t}_i^{vm\_launch} + \hat{t}'_i + \hat{t}_i^{vm\_remove} + \max_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \left(\hat{\tau}_j + \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Прогнозируемое время выполнения схемы решения задачи в среде с виртуализированными ресурсами в режиме fork/join на основе пользовательских оценок времени выполнения модулей. Оценка $\hat{T}'_{uvs}$ времени выполнения схемы решения задачи в среде с виртуализированными ресурсами соответственно в режиме fork/join определяются следующим образом:

$$\hat{T}'_{uvs} = \sum_{l=1}^{m} \hat{\tau}_l,$$

$$\hat{\tau}_l = \max_{\forall i \in \overline{1,k}: s_{li}=1} \left(\hat{q}_i + \hat{t}_i^{vm_{launch}} + \hat{t}'_i + \hat{t}_i^{vm_{remove}} + \sum_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Прогнозируемое время выполнения схемы решения задачи в асинхронном режиме с рестартом модулей на основе пользовательских оценок времени их выполнения. Оценка $\hat{T}_{urs}$ времени выполнения схемы решения задачи в

асинхронном режиме с рестартом модулей определяется следующим образом:

$$\hat{T}_{urs} = \max_{i=\overline{1,k}} \hat{\tau}_i ,$$

$$\hat{\tau}_i = \hat{q}_i + \hat{t}_i^{fault} + \hat{t}_i' + \hat{t}_i^{rest} + \max_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \left(\hat{\tau}_j + \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Прогнозируемое время выполнения схемы решения задачи в режиме fork/join с рестартом модулей на основе пользовательских оценок времени их выполнения. Оценка $\hat{T}'_{urs}$ времени выполнения схемы решения задачи в среде с виртуализированными ресурсами соответственно в режиме fork/join определяется следующим образом:

$$\hat{T}'_{urs} = \sum_{l=1}^{m} \hat{\tau}_l,$$

$$\hat{\tau}_l = \max_{\forall i \in \overline{1,k}: s_{li}=1} \left(\hat{q}_i + \hat{t}_i^{fault} + \hat{t}_i' + \hat{t}_i^{rest} + \sum_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Прогнозируемое время выполнения схемы решения задачи в асинхронном режиме по готовности данных на основе скорректированных пользовательских оценок времени выполнения модулей. Оценка $\hat{T}_{uss}$ времени выполнения схемы решения задачи в асинхронном режиме по готовности данных на основе пользовательских оценок времени выполнения модулей определяется следующим образом:

$$\hat{T}_{uss} = \max_{i=\overline{1,k}} \hat{\tau}_i ,$$

$$\hat{\tau}_i = \hat{q}_i + \beta_i \hat{t}_i'' + \max_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \left(\hat{\tau}_j + \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right),$$

где $\hat{t}_i''$ – скорректированная пользовательская оценка времени выполнения модуля, $\beta_i > 0$ – это корректирующий коэффициент, вычисляемый на основе вычислительной истории:

$$\beta_i = \psi(i, MH, T_h).$$

Функция $\psi(i, MH, T_h)$ вычисляет значение коэффициента β_i, используя среднее или медианное значения по выборке времени выполнения модуля, реализующего операцию f_i, за определенный период T_h.

Прогнозируемое время выполнения схемы решения задачи в режиме fork/join на основе скорректированных пользовательских оценок времени выполнения модулей. Оценка $\hat{T}'_{uss}$ времени выполнения схемы решения задачи в режиме fork/join определяется следующим образом:

$$\hat{T}'_{uss} = \sum_{l=1}^{m} \hat{\tau}_l,$$

$$\hat{\tau}_l = \max_{\forall i \in \overline{1,k}: s_{li}=1} \left(\hat{q}_i + \beta_i \hat{t}''_i + \sum_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Прогнозируемое временя выполнения схемы решения задачи в среде с виртуализированными ресурсами в асинхронном режиме на основе скорректированных пользовательских оценок времени выполнения модулей. Оценка $\hat{T}_{usvs}$ времени выполнения схемы решения задачи в среде с виртуализированными ресурсами в асинхронном режиме определяется следующим образом:

$$\hat{T}_{usvs} = \max_{i=\overline{1,k}} \hat{\tau}_i \,,$$

$$\hat{\tau}_i = \hat{q}_i + \hat{t}_i^{vm\_launch} + \beta_i \hat{t}''_i + \hat{t}_i^{vm\_remove} + \max_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \left(\hat{\tau}_j + \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Прогнозируемое время выполнения схемы решения задачи в среде с виртуализированными ресурсами в режиме fork/join на основе скорректированных пользовательских оценок времени выполнения модулей. Оценка $\hat{T}'_{usvs}$ времени выполнения схемы решения задачи в среде с виртуализированными ресурсами соответственно в режиме fork/join определяются следующим образом:

$$\hat{T}'_{usvs} = \sum_{l=1}^{m} \hat{\tau}_l,$$

$$\hat{\tau}_l = \max_{\forall i \in \overline{1,k}: s_{li}=1} \left(\hat{q}_i + \hat{t}_i^{vm_{launch}} + \beta_i \hat{t}_i'' + \hat{t}_i^{vm_{remove}} + \sum_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Прогнозируемое время выполнения схемы решения задачи в асинхронном режиме с рестартом модулей на основе скорректированных пользовательских оценок времени их выполнения. Оценка $\hat{T}_{usrs}$ времени выполнения схемы решения задачи в асинхронном режиме с рестартом модулей определяется следующим образом:

$$\hat{T}_{usrs} = \max_{i=\overline{1,k}} \hat{\tau}_i ,$$

$$\hat{\tau}_i = \hat{q}_i + \hat{t}_i^{fault} + \beta_i \hat{t}_i'' + \tilde{t}_i + \hat{t}_i^{rest} + \max_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \left(\hat{\tau}_j + \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Прогнозируемое время выполнения схемы решения задачи в режиме fork/join с рестартом модулей на основе скорректированных пользовательских оценок времени их выполнения. Оценка $\hat{T}'_{usrs}$ времени выполнения схемы решения задачи в среде с виртуализированными ресурсами соответственно в режиме fork/join определяется следующим образом:

$$\hat{T}'_{usrs} = \sum_{l=1}^{m} \hat{\tau}_l,$$

$$\hat{\tau}_l = \max_{\forall i \in \overline{1,k}: s_{li}=1} \left(\hat{q}_i + \hat{t}_i^{fault} + \beta_i \hat{t}_i'' + \tilde{t}_i + \hat{t}_i^{rest} + \sum_{\forall j \in \overline{1,k}: p_{ij}=1, i \neq j} \frac{\hat{d}_{ji}}{\omega_{ji}(t) w_{ji}} \right).$$

Пример. В качестве примера проведем сравнение погрешности различных способов прогнозирования времени выполнения заданий для задач линейной алгебры (перемножение матриц размерности $n \times n$). На рисунке 2.7 приведены результаты времени выполнения заданий по перемножению матриц для $n = 1000, 1500, 2000, 2500, 3000$ одной и той же программой на эталонном и исследуемом узлах, а также время выполнения этих заданий, определенное с помощью трех способов его прогнозирования, рассмотренных выше.

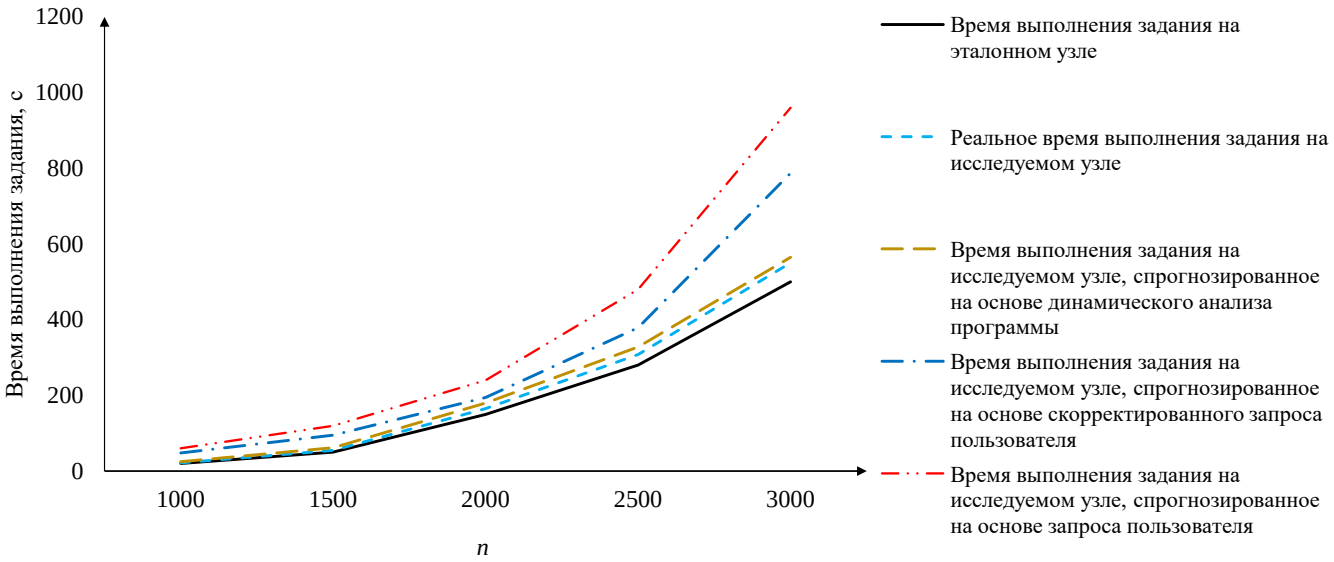


Рисунок 2.7 – Результаты реального и прогнозируемого времени выполнения заданий по перемножению матриц

Очевидно, что наиболее точный прогноз формируется на основе динамического анализа программ. На рисунке 2.8 приведены погрешности различных способов прогнозирования времени выполнения заданий в сравнении с реальным временем их выполнения в исследуемом узле. Результаты, приведенные на рисунке 2.8, показывают, что погрешность времени выполнения задания, спрогнозированного на основе динамического анализа программы, убывает с увеличением размерности матриц. В данном примере она не превышает 13.64%.

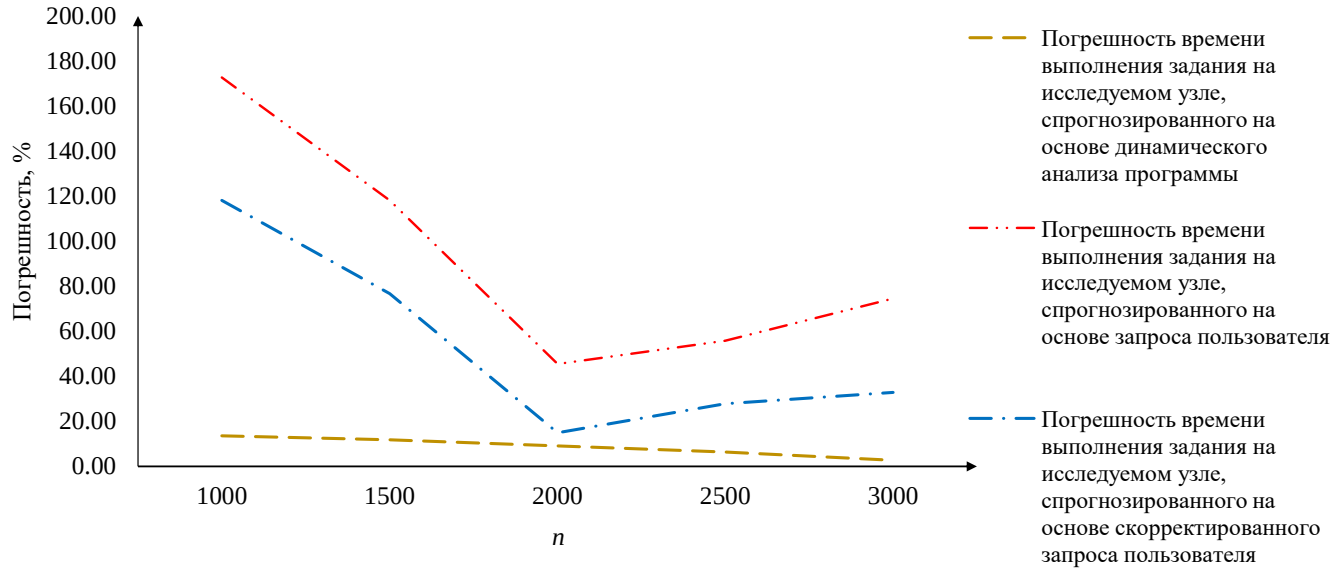


Рисунок 2.8 – Погрешность прогнозируемого времени выполнения заданий

В то же время изменение погрешности времени выполнения задания, спрогнозированного на основе запроса пользователя, может иметь немонотонный характер. На практике время выполнения задания, спрогнозированного на основе запроса пользователя, как правило, завышается. Корректировка времени в запросе пользователя на основе вычислительной истории его заданий может уменьшить погрешность прогнозирования.

На рисунке 2.9 приведены результаты прогнозирования времени выполнения для потока из 110 заданий различных пользователей по решению задач линейной алгебры. В разных задачах матрицы имели размерности 1000×1000, 1500×1500, 2000×2000, 2500×2500 и 3000×3000.

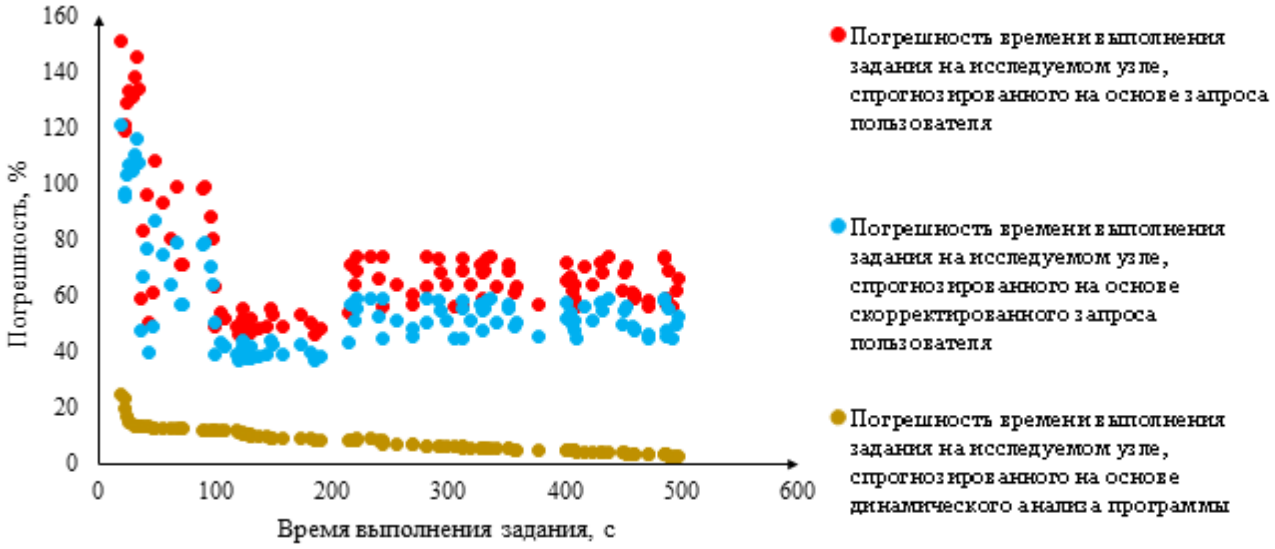


Рисунок 2.9 – Погрешность времени выполнения заданий потока

На рисунке 2.10 приведено среднеквадратическое отклонение погрешности прогнозного времени выполнения заданий, полученной разными способами. Для варианта с прогнозированием времени выполнения заданий на основе динамического анализа программы значения погрешностей сгруппированы достаточно близко к среднему значению 8.27, что говорит о возможности применения данного метода прогнозирования на практике.

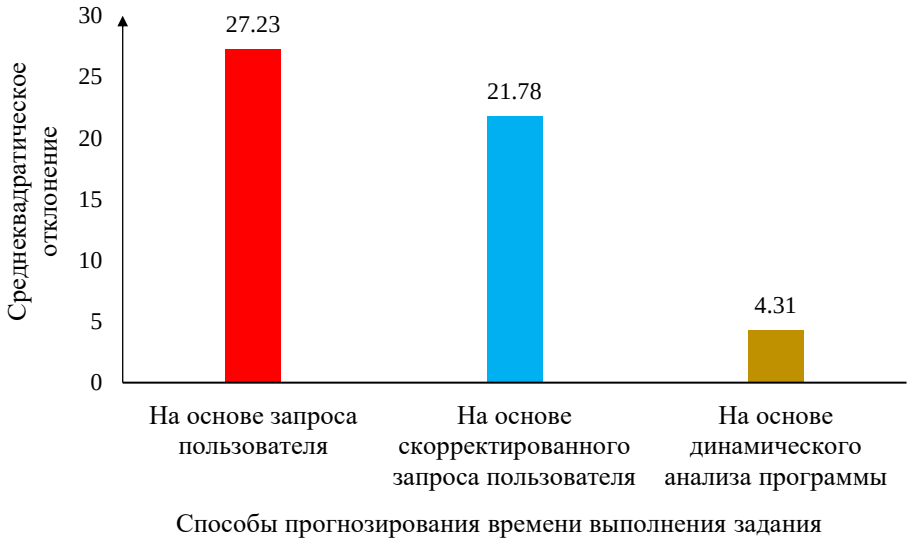


Рисунок 2.10 – Среднеквадратическое отклонение погрешности прогнозного времени выполнения заданий

Следует отметить, что для приложений разрабатываемых на основе непрерывной интеграции их ПО [64] можно использовать результаты тестирования модулей в узлах среды для получения оценок времени выполнения модулей вместо соответствующих прогнозных оценок. В [85] экспериментально показано, что оценки, полученные путем тестирования ПО, могут существенно улучшать пользовательские оценки времени выполнения модулей, а также такие оценки, скорректированные с учетом вычислительной истории.

Прогнозируемая надежность схемы решения задачи. Основные аспекты предложенного автором подхода к оценке надежности схем решения задач рассмотрены в [82, 334, 349].

Узлы ГРВС имеют различную степень надежности. В общем случае в среде существуют простейшие потоки отказов различных видов. Отказы каждого вида возникают с некоторой постоянной интенсивностью $\lambda = const$. Будем считать, что i-й узел представляет собой восстанавливаемое устройство без резервирования, а время его работы до отказа имеет экспоненциальное распределение

$$g_i(t) = p_i(t) = 1 - e^{-\lambda_i t},$$

где $\lambda_i > 0$, t – время возникновения отказа, $T_i < t$ – случайная величина наработки

до отказа i-го узла. Функция надежности i-го узла в этом случае имеет вид

$$\bar{g}_i(t) = 1 - F_i(t) = e^{-\lambda_i t}.$$

Стационарный коэффициент готовности [321], характеризующий i-й узел в произвольный момент времени t_0 по статистическим результатам его работы, определяется соотношением

$$K_i^{stat} = \frac{t_i^w}{t_i^w + t_i^r},$$

где $t_i^w > 0$ – суммарное время работы i-го узла, $t_i^r > 0$ – суммарное время восстановления i-го узла.

Тогда коэффициент интервальной готовности, показывающий вероятность того, что i-й узел окажется работоспособным в некоторый момент времени t_0 и проработает безотказно в течение интервала времени длительностью t, будет определен как

$$K_i^{in}(t_0, t) = K_i^{stat} g_i(t).$$

Модель надежности схемы решения задачи базируется на использовании методов логико-вероятностного анализа, применяемых к сложным техническим системам в работах И.А. Рябинина. Расчет надежности схемы решения задачи производится на основе логико-вероятностного метода [349], базирующегося на реализации перехода от описания условий надежности функционирования сложной системы с помощью функций алгебры логики к вероятностным функциям для определения показателей этой надежности.

Резервирование – один из простых и достаточно часто используемых методов повышения надежности вычислительных систем [321]. В нашем случае резервирование заключается в использовании дополнительных узлов, которые в случае отказа основных узлов при выполнении плана решения задачи могут взять на себя их функции. Дополнительные узлы являются нагруженным резервом и в такой же степени, что и основные узлы, применяются в ГРВС. Использование нагруженного резерва обусловлено тем, что другие виды резервирования, такие как

ненагруженный резерв, горячий резерв или многоверсионная реализация задания [262], приводят к существенным накладным расходам и больше подходят для систем реального времени. Поскольку при таком резервировании для достижения требуемого показателя надежности может потребоваться слишком большое число узлов, в статье рассматривается задача формирования дополнительного набора узлов, обеспечивающих достижение показателя надежности вычислительного процесса, максимально близкого заданному критерию надежности с учетом ограничений на число выделяемых узлов.

Пусть $s \in S$ – схема решения задачи, $p^*(t)$ – требуемый показатель надежности схемы в момент времени t, n_s – число сегментов узлов ГРВС, участвующих в выполнении схемы s, а x – набор булевых переменных (параметров) x_{ijk}, отражающих события завершения ($x_{ijk} = 1$) или незавершения ($x_{ijk} = 0$) операции $f_i \in F$ в k-м узле j-го сегмента, $i = \overline{1, n_f}$, $j = \overline{1, n_s}$. Индекс k переменной x_{ijk} указывает соответственно основной ($k = 1$) и резервные ($k \geq 2$) узлы j-го сегмента, выделенные для операции f_i. В каждом j-м сегменте можно назначить n_j резервных узлов для схемы s. Предполагается, что все узлы одного и того же сегмента являются однородными. Возможность запуска модулей в основных или резервных узлах обуславливает различные сценарии выполнения схемы s.

Введем следующие обозначения: $y_i(x)$ – булева функция, определяющая условия применения операции f_i в процессе вычислений по схеме s; $p_{ijk}(t)$ – вероятность выполнения операции f_i на k-м узле j-го сегмента в момент времени t (стационарный коэффициент готовности, характеризующий k-й узел в произвольный момент времени t по статистическим результатам его работы). Логическая схема надежности схемы s описывается формулами

$$y_1(x) \equiv 1, \tag{17}$$

$$y_i(x) = \begin{cases} h_i(x), & \text{если } i = 2, \\ x_{ij_i1} h_i(x), & \text{если } i > 2, \end{cases} \tag{18}$$

$$h_i(x) = \bigwedge_{\forall k: w_{ik}=1} y_k(x), \tag{19}$$

где $i = \overline{1, n_f}$, $j_i = \overline{1, n_s}$, $k = \overline{1, n_f}$. Первоначально в этой схеме основной узел j-го сегмента выделяется для операции f_i.

Булева функция $y_2(x)$, определяющая условия выполнения целевого модуля, вычисляет показатель надежности схемы s. После выполнения всех подстановок, определенных формулами (17)-(19), функция $y_2(x)$ принимает вид

$$y_2(x) = \bigwedge_{i=3}^{n_f} x_{ij_i1}, \tag{20}$$

где $j_i = \overline{1, n_a}$.

Для получения показателя надежности схемы s производится переход [299] с помощью соответствующих правил (таблица 2) от функции $y_2(x)$ к вероятностной функции $P(t)$ следующего вида:

$$P(t) = \prod_{i=3}^{n_f} p_{ij_i1}(t).$$

Таблица 2 – Правила символьного перехода от булевой функции к вероятностной функции

Элемент булевой формулы	Элемент формулы расчета вероятности
x_{ijk}	$p_{ijk}(t)$
$\bar{x}_{ijk}$	$1 - p_{ijk}(t)$
∧	·
∨	+

Функция $P(t)$ вычисляет вероятность осуществления единственного имеющегося сценария выполнения схемы s. Если $P(t) < p^*(t)$, то требуется преобразование функции $y_2(x)$ путем улучшения показателей надежности элементов ее структуры. Функция $y_2(x)$ из (20) для схемы s без резервирования

узлов преобразуется в функцию $y_2'(x)$ в дизъюнктивной нормальной форме без отрицаний, которая является монотонной и соответствует схеме с резервированием узлов. Свойство монотонности обеспечивает отсутствие в структуре функции $y_2'(x)$ элементов, для которых улучшение показателей их надежности ухудшает показатели надежности плана s в целом [287].

Пусть $J = \{j: n_j = 0, j \in \overline{1, n_s}\}$ – множество индексов сегментов, в которых нет резервных узлов. Алгоритм А.3 преобразования функции $y_2(x)$, соответствующей схеме s без резервирования узлов, в функцию $y_2'(x)$, соответствующую схеме s с резервированием узлов, включает следующие этапы.

А.3.1. Если

$$\sum_{j=1}^{n_s} n_j = 0,$$

т.е. отсутствует возможность резервирования узлов, то завершение преобразования, иначе – определение индекса операции с минимальной вероятностью ее выполнения

$$k = \underset{i=\overline{3,n_f}, j_i \notin J}{\operatorname{argmin}} \left(1 - \prod_{l=1}^{e} \left(1 - \bar{p}_{ij_i1}(t)\right)\right),$$

где $e = n_{ij_i}$, n_{ij_i} – число узлов, выделенных в j-м сегменте для операции f_i.

А.3.2. Уменьшение n_{j_k} на единицу.

А.3.3. Осуществление резервирования дополнительного узла j_k-го сегмента для операции f_k путем замены элемента x_{kj_ke} преобразуемой булевой формулы элементом $x_{kj_ke} \vee x_{kj_k(e+1)}$, где $e = n_{kj_k}$, n_{kj_k} – число узлов, выделенных в j_k-м сегменте для операции f_k.

А.3.4. Увеличение n_{kj_k} на единицу.

А.3.5. Приведение функции $y_2'(x)$, полученной в результате преобразования, к виду

$$y_2'(x) = \bigvee_{l=1}^{n} K_l \,, \tag{21}$$

$$K_l = \bigwedge_{i=3}^{n_f} x_{i j_i e} \,,$$

$$n = \prod_{i=3}^{n_f} \sum_{j=1}^{n_s} n_{ij} \,,$$

где $e = k_{j_i}$, $k_{j_i} \in \overline{1, n_{\iota J_\iota}}$. Каждая элементарная конъюнкция в формуле (21) представляет собой один из сценариев возможного выполнения схемы s. Элементарные конъюнкции функции $y_2'(x)$ пронумерованы от 1 до n в соответствии с их рангом по возрастанию. С целью обеспечения несовместности этих сценариев производится ортогонализация функции $y_2'(x)$ с помощью алгоритма, предложенного в [300]. Функция $\tilde{y}_2'(x)$, ортогональная функции $y_2'(x)$, принимает вид

$$\tilde{y}_2'(x) = K_1 \vee \overline{K}_1 K_2 \vee \ldots \vee \overline{K}_1 \overline{K}_2 \ldots \overline{K}_{n-1} K_n.$$

А.3.6. Упрощение функции $\tilde{y}_2'(x)$ путем удаления тождественно равных нулю и поглощаемых конъюнкций.

А.3.7. Переход с помощью соответствующих правил (таблица 2) от функции $\tilde{y}_2'(x)$ к вероятностной функции $P'(t)$ следующего вида:

$$P'(t) = \sum_{i \in I} p_i'(t) \,,$$

где $p_i'(t)$ – вероятность осуществления i-го сценария выполнения схемы s.

А.3.8. Вычисление показателя надежности схемы s с помощью функции $P'(t)$.

А.3.9. Если $P'(t) < p^*(t)$, то переход на этап А.3.1. Иначе – завершение алгоритма преобразования.

Рассмотренный выше процесс резервирования узлов обеспечивает достижение показателя надежности вычислительного процесса, максимально

приближенного к заданному критерию надежности с учетом ограничений на число выделяемых резервных узлов. Данные ограничения гарантируют сходимость процесса резервирования узлов.

Пример. Рассмотрим поливариантную схему решения задачи, включающую 4 сценария решения задачи “вычислить z_4 по z_1” (рисунок 2.11). Она включает следующие схемы:

$$s_1: [f_1] \rightarrow [f_3|f_4] \rightarrow [f_6] \rightarrow [f_2];$$
$$s_2: [f_1] \rightarrow [f_3|f_4] \rightarrow [f_7] \rightarrow [f_2];$$
$$s_3: [f_1] \rightarrow [f_3|f_5] \rightarrow [f_6] \rightarrow [f_2];$$
$$s_4: [f_1] \rightarrow [f_3|f_5] \rightarrow [f_7] \rightarrow [f_2].$$

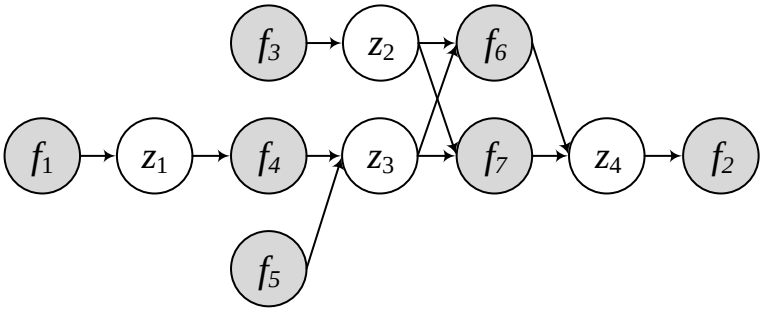


Рисунок 2.11 – Поливариантная схема решения задачи

Пусть задан показатель надежности плана $p^*(t) = 0.995$. В таблице 3 представлены логические схемы надежности для $s_1 - s_4$. В этих логических схемах функция $y_2(x)$ отражает также показатели надежности вычислений без резервирования узлов. В таблицах 4 и 5 представлены время t_s и стоимость c_s выполнения модулей в узлах пяти различных сегментов. Значения t_s и c_s приведены в условных единицах измерения. Вероятности выполнения модулей равны соответственно 0.9999, 0.9999, 0.9999, 0.9999 и 0.9900 в узлах пяти различных сегментов. При минимизации времени выполнения наилучшим является схема s_1 (время ее выполнения равно 4.00). При минимизации стоимости наилучшим является схема s_2 (время ее выполнения равна 7.20).

Таблица 3 – Логические схемы надежности для $s_1 - s_4$

s_1	s_2	s_3	s_4
$y_1 \equiv 1$	$y_1 \equiv 1$	$y_1 \equiv 1$	$y_1 \equiv 1$
$y_3 = y_1 x_{3 j_3 1}$	$y_3 = y_1 x_{3 j_3 1}$	$y_3 = y_1 x_{3 j_3 1}$	$y_3 = y_1 x_{3 j_3 1}$
$y_4 = y_1 x_{4 j_4 1}$	$y_4 = y_1 x_{4 j_4 1}$	$y_5 = y_1 x_{5 j_5 1}$	$y_5 = y_1 x_{5 j_5 1}$
$y_6 = y_3 y_4 x_{6 j_6 1}$	$y_7 = y_3 y_4 x_{7 j_7 1}$	$y_6 = y_3 y_5 x_{6 j_6 1}$	$y_7 = y_3 y_5 x_{7 j_7 1}$
$y_2 = y_6 = = x_{3 j_3 1} x_{4 j_4 1} x_{6 j_6 1}$	$y_2 = y_7 = = x_{3 j_3 1} x_{4 j_4 1} x_{7 j_7 1}$	$y_2 = y_6 = = x_{3 j_3 1} x_{5 j_5 1} x_{6 j_6 1}$	$y_2 = y_7 = = x_{3 j_3 1} x_{5 j_5 1} x_{7 j_7 1}$

Таблица 4 – Время выполнения модулей

Модуль	Сегмент				
	1	2	3	4	5
f_3	5.00	5.00	2.50	3.50	7.50
f_4	2.00	2.00	1.00	1.40	2.70
f_5	3.00	3.00	1.50	2.10	4.50
f_6	3.00	3.00	1.50	2.10	4.50
f_7	4.00	4.00	2.00	2.80	6.00

Таблица 5 – Стоимость выполнения модулей

Модуль	Сегмент				
	1	2	3	4	5
f_3	8.33	8.33	2.50	3.50	75.00
f_4	3.33	3.33	5.00	7.00	2.73
f_5	3.33	3.00	3.75	4.20	9.00
f_6	3.33	3.00	3.75	4.20	9.00
f_7	6.67	6.67	2.00	2.80	30.00

Распределение модулей по сегментам представлено в таблице 6. Вероятности выполнения схем s_1 и s_2 без резервирования узлов соответственно равны 0.9987 и

0.9898. Таким образом, в плане s_2 необходимо произвести резервирование дополнительного узла.

Функция $y_2(x)$, соответствующая плану s_2, после распределения модулей по сегментам имеет вид

$$y_2(x) = x_{331}x_{451}x_{731}.$$

Таблица 6 – Распределение модулей по сегментам

Схема решения задачи	Модуль	Сегмент				
		1	2	3	4	5
s_1	f_3	–	–	+	–	–
	f_4	–	+	–	–	–
	f_5	–	–	+	–	–
s_2	f_3	–	–	+	–	–
	f_4	–	–	–	–	+
	f_7	–	–	+	–	–

Резервирование выполняется в пятом сегменте, так как его ресурсы обеспечивают минимальную вероятность выполнения модуля f_4 по сравнению с вероятностями выполнения модулей f_3 и f_7.

После резервирования функции $y_2'(x)$ и $\tilde{y}_2'(x)$ принимают вид

$$y_2'(x) = x_{331}x_{451}x_{731} \vee x_{331}x_{452}x_{731},$$

$$\tilde{y}_2'(x) = x_{331}x_{451}x_{731} \vee x_{331}x_{452}x_{731}\bar{x}_{451}.$$

Далее осуществляются переход к вероятностной функции $P'(t)$ и расчет показателя надежности плана s_2:

$$P'(t) = p_{331}p_{451}p_{731} \vee p_{331}p_{452}p_{731}(1 - p_{451}) = 0.9997.$$

Так как $P'(t) > p^*(t)$, процесс резервирования узлов завершен.

Прогнозируемая стоимость выполнения задания. Оценка $\hat{C}_s$ стоимости выполнения схемы решения задачи определяется по следующей формуле с учетом прогнозных оценок времени выполнения модулей, реализующих операции схемы:

$$\hat{C}_s = \sum_{i=1}^{k} \pi_i \hat{t}_i + \pi_i^* ,$$

где $\hat{t}_i$ – оценка (прогнозная, пользовательская, скорректированная или полученная на основе тестирования) времени выполнения модуля, реализующего операцию f_i, $\pi_i \geq 0$ – себестоимость единицы процессорного времени узла, в котором он выполняется, $\pi_i^* \geq 0$ – торговая наценка, закладываемая владельцем данного узла.

Критерии эффективности использования ресурсов.

Пусть r_i^{peak} – пиковая производительность *i*-го ресурса, а матрицы **R** и **L** размерности $n_r \times n_j$ представляют соответственно производительность и загруженность ресурсов при выполнении заданий за определенный период времени τ. Элементы r_{ij} и l_{ij} матриц отражают производительность и загруженность *i*-го ресурса при выполнении *j*-го задания.

$$r^{sys} = \sum_{i=1}^{n_r} \sum_{j=1}^{n_j} r_i^{peak} \gamma_j \, l_{ij} \rightarrow max,$$

$$l^{sys} = \sum_{i=1}^{n_r} \sum_{j=1}^{n_j} \delta_{ij} \gamma_j \, l_{ij} \rightarrow max,$$

$$\delta_{ij} = \frac{r_{ij}}{\sum_{i=1}^{n_r} r_{ij}} .$$

2.4. Технология многокритериального выбора управленческих решений

В процессе принятия управленческих решений в ГРВС зачастую возникает необходимость выбора вариантов значений критериев качества (времени выполнения, эффективности, надежности, стоимости и других показателей) вычислительных процессов в текущей ситуации и сформировать на их основе управляющие воздействия на системы планирования вычислений и распределения ресурсов. Как правило, управление осуществляется на основе нескольких, зачастую противоречащих критериев. В связи с этим разработка средств

поддержки многокритериального выбора для предметного специалиста является актуальной проблемой в области ситуационного управления [301].

Важными характеристиками такого механизма для лица, принимающего решение (специалиста предметной области), являются легкость его использования и скорость вычислений, необходимых для выработки управляющих воздействий. Простота механизма обуславливает в свою очередь требование минимального объема входной информации, предоставляемой специалистом предметной области и необходимой для его применения. В диссертации предложена реализация многокритериального выбора рассматриваемых показателей качества вычислительных процессов [358, 381, 390, 391].

Пусть имеется n критериев $y_1 = \phi_1(\mathbf{x}), y_2 = \phi_2(\mathbf{x}), \dots, y_n = \phi_n(\mathbf{x})$, показывающих качество вычислительного процесса, где $\mathbf{x} = (x_1, x_2, \dots, x_l)$ – это вектор входных переменных его модели, $x_1, x_2, \dots, x_l, y_1, y_2, \dots, y_n \in Z$. Для различных y_j абстрактная связь $\phi_j(\mathbf{x})$ может являться функциональным, статистическим, неоднозначным или иным отображением, $j = \overline{1,n}$.

Для каждого критерия заданы его предельные значения y_j^{min} и y_j^{max} такие, что $y_j^{min} \le y_j \le y_j^{max}$, а также указано условие его оптимальности

$$y_j \to y_j^{min}(y_j^{max}), \tag{22}$$

где $j = \overline{1,n}$.

Предельные значения и условия оптимальности представлены матрицей $\mathbf{Y}^*$ размерности $3 \times n$, где y_{1j}^* и y_{2j}^* содержат соответственно y_j^{min} и y_j^{max}, а $y_{1j}^* = 0$ ($y_{1j}^* = 1$) определяет условие оптимальности $y_j \to y_j^{min}$ ($y_j \to y_j^{max}$).

Матрица $\mathbf{V}$ размерности $m \times n$ содержит множество значений критериев $y_1, y_2, \dots, y_n$, полученных путем варьирования значений переменных $x_1, x_2, \dots, x_l$ при моделировании вычислительного процесса. Ее i-я строка $\mathbf{V}(i,) = (v_{i1}, v_{i2}, \dots, v_{in})$ соответствует i-му варианту значений этих критериев, а ее j-й столбец $\mathbf{V}(,j) = \left(v_{1j}, v_{2j}, \dots, v_{mj}\right)$ включает значения критерия $y_j(\mathbf{x})$ в разных вариантах, $i = \overline{1,m}$, $j = \overline{1,n}$.

Матрица $\widehat{\mathbf{V}}$ размерности $m \times n$ включает оценки значений критериев $y_1, y_2, \dots, y_n$. Ее i-я строка $\widehat{\mathbf{V}}(i,) = (\hat{v}_{i1}, \hat{v}_{i2}, \dots, \hat{v}_{in})$ соответствует $\mathbf{V}(i,)$, а ее j-й столбец $\widehat{\mathbf{V}}(,j) = (\hat{v}_{1j}, \hat{v}_{2j}, \dots, \hat{v}_{mj})$ соответствует $\mathbf{V}(,j)$, $i = \overline{1,m}$, $j = \overline{1,n}$.

Для оценки значений критерия y_j множество элементов столбца $\mathbf{V}(,j)$ разбивается на подмножества $V_1, V_2, \dots, V_k$, попарно непересекающихся. Если $y_j \to y_j^{max}$, то эти подмножества упорядочиваются по возрастанию значений $v_{1j}, v_{2j}, \dots, v_{mj}$. В противном случае, когда $y_j \to y_j^{min}$, они упорядочиваются по убыванию значений $v_{1j}, v_{2j}, \dots, v_{mj}$. В соответствии с этим упорядочением каждое подмножество V_s получает свой индекс $s \in \overline{1,k}$, который используется для оценки значений $v_{ij} \in V_s$. Таким образом, $\hat{v}_{ij} = s$, $\forall v_{ij} \in V_s$, $i = \overline{1,m}$, $j = \overline{1,n}$, $s \in \overline{1,k}$. Чем больше значение s, тем лучше качество значений $v_{ij} \in V_s$.

Оценка значений для разных критериев $y_1, y_2, \dots, y_n$ реализуется системной операцией $f_1^s(\mathbf{V}, m, n, \mathbf{Y}^* \to \widehat{\mathbf{V}})$, которая может выполняться параллельно.

Сформулированы следующие правила многокритериального выбора вариантов значений критериев $y_1, y_2, \dots, y_n$, представленных строками матрицы $\mathbf{V}$:

$$\mathbf{V}(i,): \left(\nexists p: \left(\hat{v}_{i1} \le \hat{v}_{p1}\right) \wedge \left(\hat{v}_{i2} \le \hat{v}_{p2}\right) \wedge \dots \wedge \left(\hat{v}_{in} \le \hat{v}_{pn}\right) \wedge \left(\exists q: \hat{v}_{iq} \ne \hat{v}_{pq}\right)\right), \quad (23)$$

$$y_j^{min} \le v_{ij} \le y_j^{max},\ i \in \overline{1,m},\ j = \overline{1,n},\ p \in \overline{1,m},\ i \ne p,\ q \in \overline{1,n};$$

$$\mathbf{V}(i,): \left(\nexists p: \sum_{j=1}^{n} \operatorname{sign}\left(\hat{v}_{pj} - \hat{v}_{ij}\right) > 0\right), \quad (24)$$

$$y_j^{min} \le v_{ij} \le y_j^{max},\ \ i \in \overline{1,m},\ j = \overline{1,n},\ p \in \overline{1,m},\ i \ne p;$$

$$\mathbf{V}(i,): \forall \mathbf{V}(p,) \exists q: \quad (25)$$

$$\left(\hat{v}_{i1} = \hat{v}_{p1}\right) \wedge \left(\hat{v}_{i2} = \hat{v}_{p2}\right) \wedge \dots \wedge \left(\hat{v}_{iq} = \hat{v}_{pq}\right) \wedge \left(\hat{v}_{i(q+1)} > \hat{v}_{p(q+1)}\right),$$

$$y_j^{min} \le v_{ij} \le y_j^{max},\ i \in \overline{1,m},\ j = \overline{1,n},\ p \in \overline{1,m},\ i \ne p,\ q \in \overline{1,n-1}.$$

Правило (23) определяет оптимальный выбор по Парето в случае отсутствия информации о значимости критериев. Правило (24) применяется при условии равнозначности критериев. Оно предполагает отбор вариантов значений

критериев, превосходящих по числу оптимальных оценок другие варианты. Правило (25) предназначено для упорядоченного множества критериев. Использование данных правил обусловлено тем, что они обладают наименьшей сложностью с вычислительной точки зрения по сравнению с другими известными методами решения подобной задачи [399], просты в реализации и требуют минимальную дополнительную информацию от эксперта.

Если в результате применения любого из правил (23)-(25) ни один вариант значений критериев $y_1, y_2, \dots, y_n$ не выбран, то задача неразрешима. В этом случае нужно менять постановку задачи и проводить новый эксперимент.

В противном случае в результате выполнения каждого из правил (23)-(25) формируется матрица $\mathbf{V}^*$ размерности $m \times r$, содержащая $r \le m$ строк матрицы $\mathbf{V}$, представляющих варианты значений критериев $y_1, y_2, \dots, y_n$ и удовлетворяющих условиям отбора. Если $r > 1$, то дополнительно необходимо избрать единственный подходящий вариант.

Пусть $y_j^{opt} = \min\limits_{i=\overline{1,m}}(\max)\, v_{ij}^*$, $j = \overline{1,n}$ в соответствии с условием (22) оптимальности критерия $y_j(\mathbf{x})$. Значения $y_1^{opt}, y_2^{opt}, \dots, y_n^{opt}$ принимаются в качестве идеальных значений критериев $y_1, y_2, \dots, y_n$. Для каждой строки $\mathbf{V}^*(i,)$ определяется эвклидова метрика относительно варианта с идеальными значениями $y_1^{opt}, y_2^{opt}, \dots, y_n^{opt}$:

$$d_i = \sqrt{\sum_{j=1}^{n} \left(y_j^{opt} - v_{ij}^*\right)^2}.$$

Тогда индекс i_{opt} строки матрицы $\mathbf{V}^*$ с оптимальным вариантом значений критериев $y_1, y_2, \dots, y_n$ находится с помощью следующего выражения:

$$i_{opt} = \operatorname*{argmin}_{i=\overline{1,r}} d_i.$$

Пусть булевы параметры $z_1, z_2, \dots, z_n$ отражают факт упорядочения критериев $y_1, y_2, \dots, y_n$ в соответствии с их значимостью. Если критерий y_i

упорядочен (не упорядочен), то $z_i = 1$ ($z_i = 0$). Введем также булев параметр z_{n+1}, служащий признаком равноценности критериев. Если параметр $z_{n+1} = 1$, то критерии являются равнозначными, в противном случае, когда $z_{n+1} = 0$, равнозначность критериев не определена.

В вычислительной модели определены следующие продукции, регулирующие применение рассмотренных правил многокритериального выбора:

$$Pr_1: \left(z_{n+1} \wedge \left(\bigvee_{j=1}^{n} z_j \right) = 0 \right); \left(z_{n+1} \vee \bigwedge_{j=1}^{n} z_j = 0 \right) \Rightarrow f_2^S\left(\mathbf{V}, \widehat{\mathbf{V}}, m, n \to \mathbf{V}^*, r, i_{opt}\right),$$

$$Pr_2: \left(z_{n+1} \wedge \left(\bigvee_{j=1}^{n} z_j \right) = 0 \right); \left(\bar{z}_{n+1} = 0 \right) \Rightarrow f_3^S\left(\mathbf{V}, \widehat{\mathbf{V}}, m, n \to \mathbf{V}^*, r, i_{opt}\right),$$

$$Pr_3: \left(z_{n+1} \wedge \left(\bigvee_{j=1}^{n} z_j \right) = 0 \right); \left(\bigvee_{j=1}^{n} \bar{z}_i = 0 \right) \Rightarrow f_4^S\left(\mathbf{V}, \widehat{\mathbf{V}}, m, n \to \mathbf{V}^*, r, i_{opt}\right).$$

где f_2^S, f_3^S, f_4^S – системные операции, $r > 0$ ($r = 0$) и $i_{opt} > 0$ ($i_{opt} = 0$) – если задача разрешима (неразрешима).

В зависимости от значений параметров $z_1, z_2, \ldots, z_{n+1}$, предусловие

$$z_{n+1} \wedge \left(\bigvee_{j=1}^{n} z_j \right) = 0$$

обеспечивает однозначный выбор продукции. Поэтому задание приоритетов продукций не требуется.

2.5. Выводы

Данная глава посвящена формализации вычислительной модели (фрагмента АМ среды) ГРВС. Во второй главе получены следующие основные результаты:

– рассмотрена вычислительная модель, используемая для формулировки постановок задач, построения схем их решения, определения критериев качества выполнения вычислительных процессов и разработки механизмов

многокритериального выбора управляющих воздействий;

- сформулирована постановка задачи управления вычислениями в ГРВС на вычислительной модели, обеспечивающая учет предпочтений владельцев ресурсов и их пользователей, а также существующих административных политик в узлах среды;
- определены основные критерии качества (время, надежность, стоимость выполнения заданий, показатели эффективности использования ресурсов) решения задач в ГРВС с точки зрения предпочтений владельцев ресурсов и их пользователей;
- разработаны модели определения показателей качества выполнения заданий в ГРВС, отличительной чертой которых является поддержка механизмов многокритериального выбора этих показателей в процессе принятия управленческих решений;
- разработаны механизмы многокритериального выбора показателей качества выполнения заданий.

Результаты исследований, представленные в данной главе, опубликованы в [65, 80, 82, 85, 334, 349, 352, 358, 378, 381, 390, 391].

Глава 3. Система классификации заданий

Современные ГРВС характеризуются высокой степенью конкуренции приложений за общие ресурсы таких сред, возникающей в процессе выполнения заданий, порождаемых этими приложениями и их пользователями. Проблема рационального распределения ресурсов заданиям актуализирует решение задач выявления, описания, классификации и применения информации о характеристиках и свойствах приложений и заданий, а также установления их соответствия распределяемым ресурсам. В настоящее время известные системы управления распределенными вычислениями, используемые на практике в рамках СУПЗ, метапланировщиков, платформ для виртуализации ресурсов и другого ПО для организации распределенных вычислений, зачастую не обладают всеми необходимыми средствами для решения вышеперечисленных задач.

В третьей главе рассматривается система классификации заданий, позволяющая использовать экспертные знания администраторов узлов ГРВС для детализации характеристик заданий различных классов с их привязкой к особенностям вычислительных ресурсов среды. Целью исследований, представленных в данной главе, является обеспечение возможности рационального распределение ресурсов системами управления вычислениями, используемыми в ГРВС, на основе предоставленных им дополнительных знаний, извлекаемых в процессе классификации. В свою очередь рациональное распределение ресурсов, как правило, ведет к повышению эффективности использования ресурсов и качества выполнения заданий.

В процессе классификации РППП и их заданий применяется признаковое описание классифицируемых объектов с использованием числовых и нечисловых характеристик. В случае классификации заданий в качестве признаков задействуются вычислительные характеристики заданий, при классификации пакетов и потоков заданий – их структурные и поведенческие свойства. Распознавание характеристик и свойств классифицируемых объектов осуществляется с помощью набора специализированных характеристических

функций. Детальная настройка требований, содержащихся в классифицированных заданиях, производится на основе методов конкретизирующего программирования, применяемых к спецификациям заданий.

3.1. Основные понятия классификации заданий

Задача в самом общем смысле ее понимания – это ситуация, определяющая действия некоторой решающей системы, в состав которой входят как люди, так и автоматы (машины) [263]. Чтобы осуществить решение задачи, такая система должна обладать средствами и способами решения. Основными этапами процесса решения задачи являются [282].

- постановка (формулировка) задачи;
- построение плана (схемы) решения задачи;
- выполнение процесса решения задачи в соответствии с планом (схемой);
- анализ результатов решения задачи.

Для этапов со второго по четвертый возможен возврат на один из предыдущих этапов с целью уточнения промежуточных результатов процесса решения задачи.

К вычислительным задачам относятся задачи, процесс решения которых осуществляется путем взаимодействия специалиста предметной области с вычислительной системой. Специалист-предметник выступает в качестве пользователя такой системы. Процесс решения вычислительной задачи характеризуется, как правило, относительно небольшим набором данных и большим объемом вычислений, выполняемых над этими данными, и включает два основных этапа:

- формирование пользователем задания вычислительной системе для решения задачи;
- выполнение этого задания в вычислительной системе.

Задание представляет собой спецификацию процесса решения задачи, содержащую информацию о требуемых вычислительных ресурсах, исполняемых прикладных программах, входных/выходных данных, а также другие сведения,

необходимые вычислительной системе для успешного выполнения процесса решения задачи. Множество заданий пользователя, поступающих в вычислительную систему, составляют поток заданий, который в дальнейшем может сливаться с потоками заданий других пользователей, образуя новые потоки. Поток заданий характеризуется следующими свойствами: мощность и структура потока, порядок поступления и обслуживания заданий, степень изменения перечисленных свойств во времени, характер взаимосвязи заданий, уровень платформенной независимости РППП, выполнение которых требуется в задании. Задания могут объединяться в наборы заданий, обладающие совокупностью однородных характеристик.

Задача является ресурсоемкой, если для ее решения пользователю не хватает каких-либо ресурсов доступной ему индивидуальной вычислительной системы (например, ПК): производительности процессоров, объемов оперативной и дисковой памяти, пропускной способности телекоммуникационной среды. В этом случае требуемое число операций, выполняемых в соответствии с алгоритмом решения ресурсоемкой задачи, должно, как правило, на несколько порядков превышать быстродействие вышеупомянутой вычислительной системы.

Процесс решения ресурсоемких вычислительных задач может требовать применения систем разного уровня – от относительно простых, имеющих в своем составе совокупность типовых вычислительных модулей, до специализированных высокопроизводительных серверов и суперкомпьютеров [248], а также распределенных вычислительных сред. Ускорение процесса решения ресурсоемких вычислительных задач достигается за счет применения параллельных и распределенных вычислений.

Здесь под распределенными вычислениями понимается способ решения задачи с использованием нескольких вычислительных устройств. Парадигма параллельных вычислений включает всю совокупность вопросов, относящихся к разработке параллельных алгоритмов решения задачи, организации параллельного использования вычислительных ресурсов для решения задачи и гибкому управлению параллельными процессами решения задачи с целью достижения

наибольшей эффективности использования вычислительной системы. Любые вычислительные устройства можно считать параллельной вычислительной системой, если они работают одновременно и их можно использовать для решения одной задачи. Параллельные или распределенные вычисления являются высокопроизводительными, если существенно ускоряют процесс решения задачи.

При решении задачи в распределенной или параллельной вычислительной системе требуется, во-первых, построить план (схему) решения задачи и, во-вторых, распределить ресурсы системы с целью выполнения процесса решения задачи с заданными критериями качества. Эффективность использования того или иного ресурса для выполнения конкретного задания зависит от степени соответствия ресурса требованиям (вычислительным характеристикам) задания. Когда ресурсы вычислительной системы разнородны, возникает необходимость классификации заданий с целью повышения эффективности использования ресурсов по результатам их распределения. Если каждому ресурсу поставлены в соответствие определенные классы заданий, то выбор такого ресурса для выполнения задания конкретного класса существенно упрощается.

Наиболее простым способом классификации объектов является их признаковое описание с использованием числовых и/или нечисловых признаков. В случае классификации заданий в качестве признаков используются вычислительные характеристики заданий.

3.2. Вычислительные характеристики заданий

В настоящее время вычислительные кластеры являются неотъемлемым технологическим элементом процесса решения фундаментальных и прикладных задач. Вычислительные кластеры организуются на базе различных программно-аппаратных платформ, архитектур и коммуникационных сред, и вследствие этого существенно отличаются по своим показателям производительности, эффективности и надежности. При объединении разнородных вычислительных ресурсов (например, нескольких кластеров) в рамках ГРВС возникает проблема оптимизации распределения потока заданий, поступающих в среду, по этим

ресурсам.

Формирование и выполнение задания осуществляется с помощью специальных утилит той системы управления ресурсами, которая используется в вычислительной среде. К таким системам относятся: во-первых, глобальные планировщики (метапланировщики) заданий на уровне Грид типа GridWay; во-вторых, локальные СУПЗ вычислительных кластеров, такие, например, как SLURM, PBS Torque или HTCondor.

Анализ результатов исследований (таблица 7), затрагивающих в той или иной степени вопросы классификации вычислительных задач и способов их постановки [231, 248, 263, 284], а также рассмотрение способов спецификации процессов решения задач в известных метапланировщиках Грид и кластерных СУПЗ позволяет выявить целый ряд важных характеристик заданий. В их числе:

- математический метод решения задачи;
- вычислительная сложность задачи;
- способ решения задачи (использование готовой программы, построение алгоритма решения задачи на основе библиотек стандартных программ, исполнение программы в режиме интерпретации или компиляции);
- место размещения (в машине пользователя или в машине системы) программы, используемой для решения задачи;
- число прогонов программы;
- наличие взаимосвязанных подзаданий;
- вид параллелизма алгоритма решения задачи (мелкозернистый, крупноблочный и смешанный параллелизм);
- способ распараллеливания вычислений: по данным или по управлению;
- степень ресурсоемкости вычислений (малые, средние и большие задания);
- необходимость управления процессом вычислений в пакетном или интерактивном режимах;
- срочность вычислений и другие особенности процесса выполнения заданий.

Таблица 7 – Подходы к классификации вычислительных задач

Подход к классификации заданий	Основы классификации
Используемые численные методы [231, 284]	Методы численного дифференцирования и интегрирования, решения систем линейных алгебраических уравнений, решения дифференциальных уравнений в частных производных и другие вычислительные методы
Способы решения задач [260, 274, 284, 290, 318, 396]	Использование готовой программы; программирование алгоритма решения задачи; построение алгоритма решения задачи на основе библиотек стандартных программ; синтез программ
Числовые характеристики [249, 263]	Малые, средние и большие задачи; задачи вычислительные (мало данных, много вычислений) и задачи обработки данных (много данных, мало вычислений)
Возможность одновременного выполнения вычислений [248]	Параллельные и последовательные вычисления
Степень параллелизма алгоритмов решения задач [229]	Мелкозернистый, крупноблочный и смешанный параллелизм
Вид вычислительных заданий [374]	Одиночные, взаимосвязанные, интерактивные и другие задания.
Поддержка контрольных точек в заданиях [402]	Пользовательские, системные – на уровне операционной системы (ОС) или уровне процесса пользователя
Структура взаимосвязанного задания [222]	Ориентированный ациклический граф или отличная от него структура
Требуемая вычислительная инфраструктура [74]	Кластер, Грид, облачная (или мультиоблачная) платформа, интегрированная среда и т.п.
Уровень виртуализации заданий [102, 165]	ВМ, контейнер, гибридная виртуализация

Значение той или иной характеристики может быть явно задано пользователем вычислительной среды, формирующим свое задание, получено из переменных окружения этой среды, установлено системой управления заданиями как значение по умолчанию или взято из конфигурационных и статистических файлов (системных или пользовательских).

Вычислительные характеристики потоков заданий задаются администратором ГРВС, а сами потоки заданий связываются с РППП, порождающими эти потоки.

3.3. Подходы к классификации вычислительных заданий в системах управления прохождением заданий

Известные на сегодняшний день метапланировщики Грид базируются на двух способах распределения ресурсов [316]: путем взаимодействия метапланировщика с кластерными СУПЗ (например, GridWay) либо за счет реализации планировщика, занимающегося поиском свободных ресурсов в Грид для выполнения конкретного приложения (например, AppLeS [20]). В обоих случаях, как правило, они не производят классификацию заданий на основе характеристик заданий и осуществляют оптимизацию выделения ресурсов, разделяемых между всеми заданиями потока, для выполнения конкретного задания, оперируя информацией, касающейся только этого задания.

Такой подход зачастую не позволяет добиться высоких показателей функционирования ГРВС в целом: ее пропускной способности, эффективности использования ресурсов и времени выполнения некоторого набора задач. Отдельные системы, которые поддерживают классификацию заданий в том или ином виде (например, система управления заданиями программного комплекса gLite), часто имеют достаточно ограниченный, нерасширяемый набор встроенных типов (классов) заданий и не осуществляют предварительную виртуальную декомпозицию ресурсов относительно этих типов с целью оптимизации процесса выбора ресурса для выполнения заданий определенного типа.

Ниже приведены примеры классификации заданий в некоторых

используемых на практике системах управления заданиями.

В системе Condor явно выделяются два класса заданий: простые и взаимосвязанные. Задания первого класса запускаются с помощью утилиты *condor_sabmit*. Задания второго класса запускаются с помощью утилиты *condor_sabmit_dag*. Дальнейшая классификация осуществляется неявно, путем указания значений различных параметров в паспорте задания.

Программный комплекс gLite разрабатывался в качестве связующего ПО проекта Enabling Grids for E-sciencE (EGEE), завершенного в 2010 г. В настоящее время развитие и поддержка системы gLite осуществляется в рамках платформы European Middleware Initiative (EMI), используемой для поддержки крупномасштабных научных исследований, например, осуществляемых в рамках проекта Worldwide Large Hadron Collider Computing Grid.

Основная и наиболее развитая часть в составе комплекса gLite – это система управления заданиями Workload Management System (WLMS) [271]. Ее назначение – поддержка выполнения программ на распределенных компьютерах, организованных в единую систему. Программный код задания (программа) выполняется на исполнительном компьютере без участия пользователя (в пакетном режиме). Сам код в рядовом случае не требует адаптации к условиям распределенной вычислительной среды. В некоторых случаях может потребоваться его дополнение прологом/эпилогом, которые выполняют подготовительные/завершающие операции, например, доставку обрабатываемых данных.

Задание представляется WLMS в виде формализованного описания, составленного на языке Job Description Language (JDL). Ресурсы распределенной вычислительной среды используются коллективно множеством пользователей, поэтому задание не обязательно начинает выполняться сразу после запуска: оно может ждать освобождения ресурсов, занятых другими заданиями. Ожидающие ресурсов задания хранятся в очередях. WLMS поддерживает работу как с простыми заданиями, так и с составными.

При классификации заданий в WLMS тип задания описывается двумя

атрибутами: Type и JobType. Атрибут Type имеет, соответственно, значения «Job» или «DAG». Значение «DAG» (Direct Acyclic Graph of dependent jobs) атрибута Type определяет задание, в котором должны быть выполнены в определенной последовательности ряд простых заданий. Для простого задания применим второй атрибут – JobType, который может принимать следующие значения:

– «Normal» – обыкновенное задание;
– «Interactive» – интерактивное задание;
– «MPICH» – параллельное задание;
– «Checkpointable» – задание с контрольными точками;
– «Partitionable» – сериализуемое (многовариантное) задание.

Перечисленные характеристики могут сочетаться друг с другом.

В одной из первоначальных версий инструментального комплекса DISCENT [345] поддерживаются более гибкие возможности классификации заданий. В нем для описания свойств заданий используется структура $(a/b/c/d)$, где параметры a, b, c и d имеют следующую интерпретацию:

– параметр a представляет собой характеристику алгоритма решения задачи с точки зрения наличия в ней подзадач; фиксированными значениями для параметра a являются 1 (отсутствие подзадач) и K (наличие подзадач);
– параметр b отражает степень параллелизма алгоритма решения задачи; фиксированными значениями параметра b являются величины L (крупноблочный параллелизм), F (мелкозернистый параллелизм) и S (последовательный алгоритм);
– параметр c характеризует процесс решения задачи с точки зрения необходимости выполнения многовариантных расчетов; фиксированными значениями параметра c являются величины: 1 (один единственный вариант данных) или N (наличие множества вариантов данных);
– параметр d специфицирует процесс решения задачи с точки зрения размещения модулей выполняемого РППП; фиксированными значениями

параметра d являются величины G (выполнение модуля, размещенного в узлах ГРВС); R (выполнение удаленного модуля).

В рамках рассмотренной выше структуры классификации заданий для каждого уже определенного типа задания могут быть образованы его подтипы с помощью задания новых значений характеристик, не являющихся базовыми характеристиками заданий.

3.4. Модель системы классификации заданий

В диссертации предложена следующая модель классификации заданий [353, 374]. Пусть имеется конечное множество $H = \{h_1, h_2, \dots, h_k\}$ характеристик заданий. Каждая характеристика h_i описывается информационной структурой, имеющей следующие компоненты:

- область D_i допустимых значений характеристики h_i, включающую символ неопределенности θ;
- целочисленный ранг $r_i \geq 1$ характеристики h_i, показывающий степень важности данной характеристики;
- вес $w_i \geq 0$ характеристики h_i, представляющий собой численное выражение важности данной характеристики.

Элементы множества H частично упорядочены по их рангу по убыванию: $r_i \geq r_{i+1}\ \forall i \in \overline{1, k-1}$.

На множестве H построено конечное множество $C = \{c_1, c_2, \dots, c_m\}$ классов заданий. Каждый класс c_j определяется базовым (обязательным) и дополнительным (необязательным) наборами характеристик из H. Характеристики базового и дополнительного набора для классов удобно представить в виде булевых матриц A и B размерности $k \times m$, элементы которых $a_{ij} = 1$ или $b_{ij} = 1$ означают, что характеристика h_i принадлежит базовому или дополнительному набору, используется в классе c_j и для этого класса имеет конкретизированную область допустимых значений $D_{ij}^* \subseteq D_i \backslash \{\theta\}$. Если $a_{ij} \vee b_{ij} = 0$, то $D_{ij}^* \equiv \{\theta\}$. Матрицы A и B должны удовлетворять следующим условиям:

$$\bigvee_{j=1}^{m}\bigwedge_{i=1}^{k}\bar{a}_{ij}=0,$$

$$\bigvee_{i=1}^{k}\bigvee_{j=1}^{m}(a_{ij}\wedge b_{ij})=0.$$

Формирование множества характеристик H (определение имен, областей допустимых значений, рангов и весов характеристик) и создание на их основе множества классов заданий C (определение для каждого класса множеств обязательных и необязательных характеристик из множества H с указанием их областей значений, допустимых для конкретного класса) осуществляется администратором ГРВС. Как правило, характеристикам классов заданий, используемым в качестве обязательных характеристик, присваиваются более высокие ранги и веса.

Каждому классу сопоставляются ресурсы ГРВС, подходящие для выполнения заданий данного класса с точки зрения предпочтений администратора. При поступлении нового задания в среду специальная программа (классификатор заданий) автоматически проверяет, какие характеристики из множества H использованы в его спецификации, и извлекает для каждой из них указанную область значений.

Задание со всеми своими характеристиками представляется булевым вектором x размерности k. Между индексами элементов вектора x и индексами характеристик из H установлено взаимно однозначное соответствие. Значение i-го элемента вектора x находится следующим образом:

$$x_i=\begin{cases}0, & \text{если } D_i'\equiv\{\theta\},\\ 1, & \text{если } D_i'\subseteq D_i\backslash\{\theta\},\end{cases}$$

где D_i' – это область значений характеристики h_i, требуемых в данном задании. Вектор x должен удовлетворять условию

$$\bigwedge_{i=1}^{k}\bar{x}_i=0.$$

Соответствие требуемых областей допустимых значений характеристик задания областям допустимых значений характеристик j-го класса устанавливается с помощью характеристической функции

$$x = \begin{cases} 0, & \text{если } \exists i: \left(a_{ij} \vee b_{ij} = 1\right) \wedge \left(D_i' \cap \overline{D}_{ij}^* \neq \emptyset\right), \\ 1 & \text{в противном случае,} \end{cases}$$

где $i \in \overline{1,k}$, $j \in \overline{1,m}$.

Для выполнения первичной классификации задания достаточно применения функции χ. В результате первичной классификации задание может быть соотнесено сразу с несколькими классами заданий.

Однако может возникнуть необходимость в более детальной конкретизации классификации задания, полученной с помощью функции χ. В этом случае следует принимать во внимание число характеристик задания, требуемые области допустимых значений которых удовлетворяют соответствующим областям допустимых значений характеристик того или иного класса, и учитывать ранги и веса этих характеристик.

Ниже введен ряд вспомогательных функций.

Будем говорить, что i-я характеристика используется как в спецификации задания, так и в описании класса c_j, если выполняется условие

$$\bar{x}_i \vee \bar{a}_{ij}\bar{b}_{ij} = 0. \tag{26}$$

Область требуемых значений i-й характеристики соответствует области допустимых значений этой характеристики для класса c_j, если выполняется условие

$$D_i' \subseteq D_{ij}^*. \tag{27}$$

Функция $\rho_j(x)$ вычисляет оценку возможности (вероятность) отнесения задания к классу c_j на основе характеристик этого задания, удовлетворяющих по всем параметрам характеристикам данного класса c_j. Значение функции определяется выражениями

$$\rho_j(x) = \frac{V_j}{S},$$

$$V_j = \frac{k_j}{n_j},$$

$$S = \sum_{l=1}^{m} \frac{k_l}{n_l},$$

где k_j (k_l) – число характеристик задания, удовлетворяющих условиям (26) и (27) относительно j-го (l-го) класса, n_j и n_l – число характеристик, определяющих классы c_j и c_l соответственно, $k_j, k_l \in \overline{0,k}$, $n_j, n_l \in \overline{1,m}$. Нетрудно заметить, что

$$\sum_{j=1}^{m} \rho_j(x) = 1, \;\; \rho_j(x) \in [0,1].$$

Функция $\sigma_j(x)$ вычисляет агрегированный числовой показатель важности характеристик задания для класса c_j с учетом рангов этих характеристик. Значение функции определяется выражением

$$\sigma_j(x) = n_t s_t + \sum_{l=1}^{t-1} (n_l s_l + k s_{l+1}),$$

где $s_1, s_2, \dots, s_t$ – упорядоченные по убыванию значения рангов, n_t и n_l – число характеристик задания, удовлетворяющих условиям (1) и (2), $\sigma_j(x) \geq 1$.

Функция $\omega_j(x)$ вычисляет сумму весов характеристик задания для класса c_j. Значение функции определяется выражением

$$\omega_j(x) = \sum_{i \in I} w_i \,,$$

где I – это множество индексов характеристик задания, удовлетворяющих условиям (26) и (27), $\omega_j(x) \geq 0$.

Функция $\varphi_j(x, z)$ вычисляет оценку возможности (вероятность) отнесения задания к классу c_j с учетом его вычислительной истории z – информационной

структуры, содержащей статистику о выполнении заданий с аналогичными характеристиками в ГРВС. Данная информация извлекается из статистических и информационных файлов СУПЗ, применяемых в узлах ГРВС. Значение функции определяется выражением

$$\varphi_j(x,z) = \begin{cases} \dfrac{1}{m}, & \text{если } \forall l \in \overline{1,m}\ \gamma_l(x,z) = 0, \\ \dfrac{\gamma_j(x,z)}{\sum_{l=1}^{m} \gamma_l(x,z)} & \text{в противном случае,} \end{cases}$$

где $\gamma_l(x,z)$ и $\gamma_j(x,z)$ – функции, возвращающие общее число выполненных заданий классов c_l и c_j с аналогичными характеристиками в ГРВС. Нетрудно заметить, что

$$\sum_{j=1}^{m} \varphi_j(x,z) = 1, \ \ \varphi_j(x,z) \in [0,1].$$

Определены верхние границы δ_ρ, δ_σ, δ_ω и δ_φ эквивалентности значений функций ρ, σ, ω и φ. Два значения одной и той же функции будем считать эквивалентными, если модуль их разницы будет меньше соответствующей верхней границы эквивалентности, либо равен ей.

Пусть с помощью функции χ сформирован булев вектор y размерности m, элемент которого $y_j = 1$ означает, что задание относится к j-му классу. Введены следующие характеристические функции, конкретизирующие принадлежность задания j-му классу с учетом перечисленной выше дополнительной информации о характеристиках задания:

$$\chi_j^{\rho}(x,y) = \begin{cases} 0, & \text{если } \max\limits_{\forall l: y_l = 1} \{\rho_l(x)\} - \rho_j(x) > \delta_\rho, \\ 1 & \text{в противном случае,} \end{cases}$$

$$\chi_j^{\sigma}(x,y) = \begin{cases} 0, & \text{если } \max\limits_{\forall l: y_l = 1} \{\sigma_l(x)\} - \sigma_j(x) > \delta_\sigma, \\ 1 & \text{в противном случае,} \end{cases}$$

$$\chi_j^{\omega}(x,y) = \begin{cases} 0, & \text{если } \max\limits_{\forall l: y_l = 1} \{\omega_l(x)\} - \omega_j(x) > \delta_\omega, \\ 1 & \text{в противном случае,} \end{cases}$$

$$\chi_j^{\varphi}(x,y,z) = \begin{cases} 0, & \text{если } \max\limits_{\forall l: y_l=1} \{\varphi_l(x,z)\} - \varphi_j(x,z) > \delta_{\varphi}, \\ 1 & \text{в противном случае.} \end{cases}$$

Следует заметить, что наличие у задания отдельно взятой характеристики с высокой значимостью (как обязательной, так и необязательной) не гарантирует отнесение этого задания к классу, в описании которого присутствует данная характеристика. Классификация осуществляется с учетом агрегированной значимости всех характеристик задания для каждого класса заданий.

3.5. Алгоритм классификации заданий

Рассмотрим алгоритм А.4 классификации задания. На вход данному алгоритму поступает вектор x, представляющий задание. На выходе получаем вектор y, содержащий информацию о классах, к которым относится данное задание. Алгоритм включает следующие этапы работы.

А.4.1. Инициализация элементов вектора y: $y_j = 0\ \forall j \in \overline{1,m}$.

А.4.2. Выполнение первичной классификации – проверка принадлежности задания каждому из m классов: $y_j = \chi_j(x)\ \forall j \in \overline{1,m}$.

А.4.3. Если $y_j = 0\ \forall j \in \overline{1,m}$, то завершение работы алгоритма (задание не может быть классифицировано).

А.4.4. Иначе – выполнение конкретизации результатов классификации, полученной на этапе А.4.2.

А.4.4.1. Редукция множества классов, к которым относится задание, с учетом числа характеристик классов: $y_j = \chi_j^{\rho}(x,y)\ \forall j: y_j = 1$.

А.4.4.2. Редукция множества классов, к которым относится задание, с учетом рангов характеристик $y_j = \chi_j^{\sigma}(x,y)\ \forall j: y_j = 1$.

А.4.4.3. Редукция множества классов задания, к которым относится задание, с учетом весов характеристик $y_j = \chi_j^{\omega}(x,y)\ \forall j: y_j = 1$.

А.4.4.4. Редукция множества классов задания, к которым относится задание,

с учетом вычислительной истории данного задания $y_j = \chi_j^{\varphi}(x, y, z)\ \forall j: y_j = 1$.

А.4.5. Завершение работы алгоритма (задание классифицировано).

Порядок выполнения этапов А.4.4.1.-А.4.4.4. работы данного алгоритма, приведенный выше, определен исходя из степени доступности информации, необходимой для применяемых на этих этапах функций ρ, σ, ω и φ. Чем выше степень доступности информации, требуемой для применения той или иной функции, тем раньше выполняется соответствующий этап работы алгоритма.

В программной реализации алгоритма классификации заданий предусмотрена возможность исключения любого из этапов А.4.4.1.-А.4.4.4. из алгоритма и/или изменения порядка применения этих этапов в алгоритме путем конфигурирования программы-классификатора администратором ГРВС.

Исключение того или иного этапа А.4.4.1.-А.4.4.4. работы алгоритма может быть обусловлено рядом причин. В их числе администратор системы не обладает достаточной информацией для назначения рангов и весов характеристик (по умолчанию характеристики имеют одинаковые ранги и веса), нет возможности получить полную вычислительную историю о выполнении заданий СУПЗ, работу с которыми программа-классификатор не поддерживает, результаты применения исключаемого этапа работы алгоритма являются избыточными с точки зрения администратора системы.

Изменение порядка применения этапов А.4.4.1.-А.4.4.4. работы алгоритма может быть обусловлено приоритетностью (с точки зрения администратора системы) информации о характеристиках задания, используемой на каждом из этих этапов.

3.6. Смягчение неопределенности

Для оценки степени смягчения неопределенности в процессе распределения заданий по ресурсам в диссертационной работе используется информационная

двоичная энтропия. Данный показатель часто используется для оценки степени неопределенности сложной системы. Предполагается, что уменьшение энтропии ведет к смягчению неопределенности.

Пусть по результатам первичной классификации найдены m классов, характеристикам которых удовлетворяет задание. Среда включает n разнородных ресурсов. Одному классу может соответствовать k ресурсов, $k \in \overline{1,n}$. Информация о соответствии между классами и ресурсами представлена булевой матрицей Y размерности $n \times m$. Элемент матрицы $y_{ji} = 1$ ($y_{ji} = 0$) показывает, что j-й ресурс соответствует (не соответствует) i-му классу.

Определим энтропию $E^c(x_c) = \sum_{i=1}^{m} p_i(x_c) E_i^c(x_c)$, отражающую степень неопределенности принадлежности задания одному из классов, где x_c – случайное событие, которое обуславливает принадлежность задания одному из m классов, $p_i(x_c)$ – вероятность такого события относительно i-го класса, $\sum_{i=1}^{m} p_i(x_c) = 1$, $E_i^c(x_c)$ – энтропия, соответствующая данному событию. Значение $E_i^c(x_c)$ вычисляется следующим образом:

$$E_i^c(x_c) = \begin{cases} 0, \text{если } p_i(x_c) = 0, \\ -log_2 p_i(x_c) \text{ в противном случае.} \end{cases}$$

Тем же способом определим энтропию $E^r(x_r) = \sum_{j=1}^{n} p_j(x_r) E_j^r(x_r)$, отражающую степень неопределенности принадлежности ресурса одному из классов, где x_r – случайное событие, которое обуславливает возможность назначения заданию одного из n ресурсов, $p_j(x_r)$ – вероятность такого события относительно j-го ресурса, $E_j^r(x_r)$ – энтропия, соответствующая данному событию. Значения $p_j(x_r)$ и $E_j^r(x_r)$ вычисляются следующим образом:

$$p_j(x_r) = \sum_{\forall l: y_{jl}=1} p_l(x_C) / \sum_{j=1}^{n} \sum_{\forall l: y_{jl}=1} p_l(x_C), \quad l \in \overline{1,m}, \quad \sum_{j=1}^{n} p_j(x_r) = 1,$$

$$E_j^r(x_r) = \begin{cases} 0, \text{если } p_j(x_r) = 0, \\ -log_2 p_j(x_r) \text{ в противном случае.} \end{cases}$$

Рассмотрим иллюстративный пример смягчения неопределенности в процессе распределения заданий по ресурсам. Пусть по результатам первичной классификации найдены 8 классов, характеристикам которых удовлетворяет

задание, поступившее в среду для выполнения указанного в задании модуля. Эти классы различаются между собой областью допустимых значений одной характеристики, определяющей допустимое время выполнения заданий. Значение данной характеристики не указано в спецификации задания.

Среда включает 3 разнородных ресурса (3 кластера, узлы которых различаются по вычислительным характеристикам). Соответствие ресурсов классам представлено матрицей

$$Y = \begin{bmatrix} 1\,1\,1\,1\,0\,0\,0\,0 \\ 0\,0\,0\,1\,1\,1\,1\,1 \\ 0\,0\,0\,0\,0\,0\,0\,0 \end{bmatrix}.$$

Имеется вычислительная история выполнения такого задания на ресурсах среды. В таблице 8 приведены значения $p_1(x_c)$, $p_2(x_c)$, …, $p_8(x_c)$, E^c, $p_1(x_r)$, …, $p_3(x_c)$ и E^c для каждого этапа смягчения неопределенности в распределение заданий по ресурсам.

Таблица 8 – Вероятности событий и показатели энтропии

№	$\boldsymbol{p_1(x_c)}$	$\boldsymbol{p_2(x_c)}$	$\boldsymbol{p_3(x_c)}$	$\boldsymbol{p_4(x_c)}$	$\boldsymbol{p_5(x_c)}$	$\boldsymbol{p_6(x_c)}$	$\boldsymbol{p_7(x_c)}$	$\boldsymbol{p_8(x_c)}$	E^c	$p_1(x_r)$	$p_2(x_r)$	$p_3(x_r)$	E^r
1	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	3.00	0.333	0.333	0.333	1.06
2	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125	3.00	0.444	0.556	0	0.99
3	0.115	0.741	0.132	0.012	0	0	0	0	1.14	0.988	0.012	0	0.09
4	0	0.936	0.064	0	0	0	0	0	0.34	1.000	0	0	0

На первом этапе вероятности принадлежности задания одному из 8 классов равны между собой, так же как равны и вероятности назначения заданию одного из 3 ресурсов. В этом случае $E^c = 3.00$ и $E^r = 1.06$.

На следующем этапе выбираются два ресурса, наиболее подходящие для выполнения задания с точки зрения экспертного опыта администраторов ресурсов. При этом степень неопределенности назначения ресурсов заданию понижается до $E^r = 0.99$.

Вторичная классификация задания на основе анализа статистической информации о вычислительной истории конкретизирует вероятности

принадлежности задания классам, полученные на третьем этапе. Значение E^c становится равным 1.14. В соответствие с этим изменяются вероятности назначения ресурсов заданию. Степень неопределенности назначения ресурсов заданию уменьшается до $E^r = 0.09$.

На последнем этапе результаты вторичной классификации уточняются на основе прогнозного времени выполнения модуля или полученных ранее данных его тестирования. В результате этого $E^c = 0.34$ и $E^r = 0$. Таким образом, 1-й ресурс назначается для выполнения задания.

3.7. Классификатор заданий

На основе приведенных выше теоретических исследований автором диссертационной работы выполнена программная реализация имитационного прототипа классификатора заданий [382] с использованием языков программирования General Purpose Simulation System (GPSS) и Programming Language Under Simulation (PLUS) [238]. Имитационное моделирование работы классификатора заданий было проведено на потоке заданий, соответствующем реальному потоку заданий, выполненных на кластере 1 с СУПЗ Cleo [306] (таблица 9). Кластеры 2 и 3 функционируют под управлением СУПЗ HTCondor и СУППЗ [309]. Все три кластера являются ресурсами ЦКП ИСКЦ. В таблице 9 для каждого кластера указаны его пиковая производительность R_{max}, процессор, установленный в узлах, число n_n узлов, число n_p процессоров и число n_c ядер, а также используемая СУПЗ и операционная система.

Таблица 9 – Вычислительные ресурсы ГРВС

Ресурс ГРВС	R_{max}, Тфлопс	Процессор	n_n/ n_p/ n_c	СУПЗ	Операционная система
Кластер 1	1.50 TFlops	Intel Xeon E5345	20/40/160	Cleo	Linux
Кластер 2	0.77 TFlops	AMD Athlon II X4	16/16/64	HTCondor	Windows
Кластер 3	0.17 TFlops	Intel Xeon	16/32/32	СУППЗ	Linux

Кластеры 1 и 3 включают выделенные узлы, которые используются только в составе кластера. Невыделенные узлы кластера 2 используются как их владельцами, так и пользователями. В СУПЗ Cleo все ядра узла выделяются заданию независимо от числа запрошенных в нем ядер. Часто это приводит к неэффективному использованию ресурсов. Ниже показано, что эта проблема успешно решается путем применения системы классификации.

В процессе имитационного моделирования в качестве унифицированного формата спецификации потока заданий использовался формат Standard Workloads Format [40]. Всего было обработано 62249 заданий. Список классов заданий, использованных в рамках эксперимента, приведен в таблице 10. Для каждого класса заданий приведены диапазоны допустимых значений ряда его базовых характеристик: число вариантов данных n_v, минимальное число требуемых ядер n_c, минимальное и максимальное время t_{min} и t_{max}, запрашиваемое для выполнения задания.

Таблица 10 – Классы заданий

Класс заданий	Тип класса	Характеристика			
		n_v	n_c	t_{min}, с	t_{max}, с
Класс 1	Базовый	1	1	–	–
Класс 2	Базовый	1	2	–	–
Класс 3	Дополнительный	1	1	1	300
Класс 4	Дополнительный	1	1	301	3600
Класс 5	Дополнительный	1	1	3601	604800
Класс 6	Дополнительный	1	2	1	300
Класс 7	Дополнительный	1	2	301	3600
Класс 8	Дополнительный	1	2	3601	604800

В системе классификации определены два базовых и шесть дополнительных классов заданий на основе анализа вычислительной истории. Дополнительные классы созданы на основе базовых. Дополнительные классы отличаются числом

требуемых ядер и временем, требуемым для выполнения заданий. Каждый дополнительный класс связан с одним из кластеров, представленных в таблице 9.

Все задания были классифицированы. Для части заданий выявлена потенциальная возможность их перемещения с кластера 1 на кластеры 2 и 3 с целью оптимизации загрузки вычислительных ресурсов. Перемещение задания возможно только на кластер, удовлетворяющий характеристикам класса задания, и при условии, что время выполнения перемещенного задания останется в диапазоне времени выполнения заданий данного класса. Число классифицированных заданий n_{job} и количество перемещенных заданий n_r для каждого дополнительного класса приведены в таблице 11.

Таблица 11 – Результаты классификации заданий

Класс заданий	n_{job}	n_r
Класс 3	10221	4675
Класс 4	7927	5861
Класс 5	1211	972
Класс 6	33453	492
Класс 7	6125	50
Класс 8	3312	95

С целью улучшения использования ресурсов в процессе модельной классификации было перемещено более 12000 заданий по сравнению с их первоначальной обработкой. Перемещенные задания сохранили все характеристики классов, определенных для них. Вследствие моделируемых перемещений заданий в модели была выполнена часть заданий, снятых по времени их ожидания в очереди, находившихся в очереди или не завершивших свое выполнение в рамках реального потока заданий. Рисунки 3.1 и 3.2 иллюстрируют изменение распределения заданий в течение суток без применения классификации и с ее использованием.

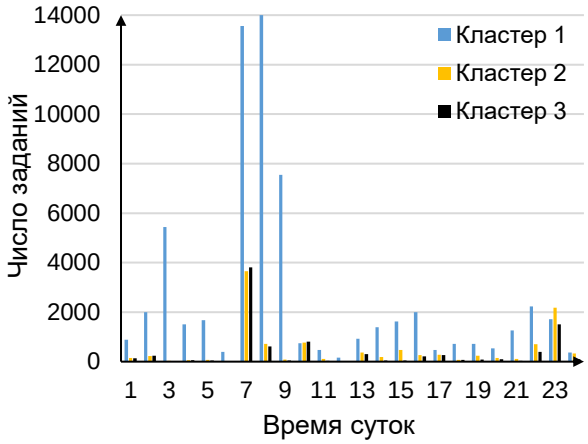


Рисунок 3.1 – Распределение заданий без их классификации

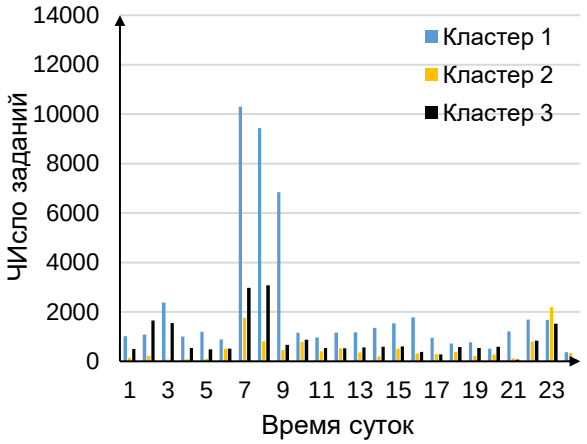


Рисунок 3.2 – Распределение заданий на основе их классификации

Полученные результаты показали, что дополнительное применение классификатора заданий совместно с СУПЗ ГРВС позволило бы повысить среднюю загрузку ресурсов кластеров при обработке исследуемого потока заданий на 18, 25 и 30% соответственно (рисунок 3.3).

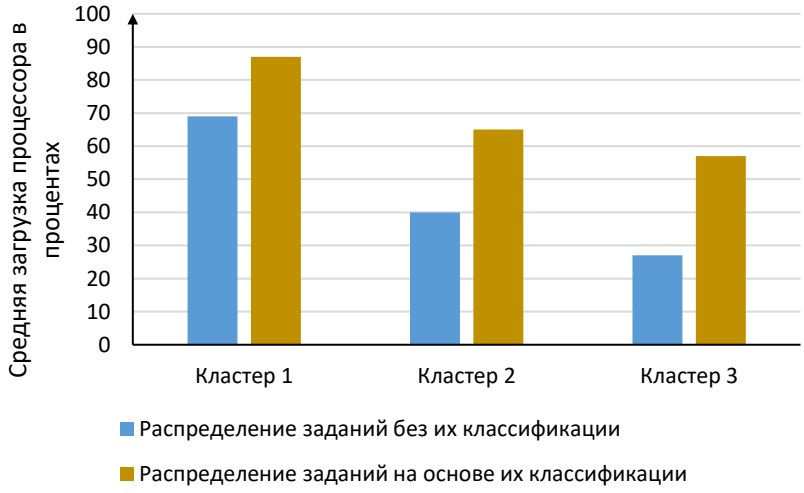


Рисунок 3.3 – Средняя загрузка процессора

В настоящее время под руководством автора диссертационной работы реализован действующий классификатор заданий и их потоков в виртуализированной ГРВС, организованной на базе ресурсов ЦКП ИСКЦ. Интерфейс классификатора заданий приведен в Приложении Г. Схема

взаимодействия компонентов классификатора заданий представлена на рисунке 3.4. Система виртуальной декомпозиции ресурсов используется администратором ГРВС для назначения классам заданий наиболее подходящих для их выполнения вычислительных узлов. На основе такого распределения все задания, относящиеся к конкретному классу, могут быть выполнены только на назначенных ему ресурсах среды.

Конструктор – компонент, обеспечивающий возможность создания в базе данных элементов (характеристик, классов и т.д.) системы классификации.

Конфигуратор – компонент классификатора заданий, обеспечивающий возможность выбора требуемых дополнительных этапов классификации заданий и определения порядка их применения.

Классификатор – компонент, реализующий процесс классификации заданий.

Веб-интерфейс – средство обеспечения доступа пользователя к компонентам классификатора заданий.

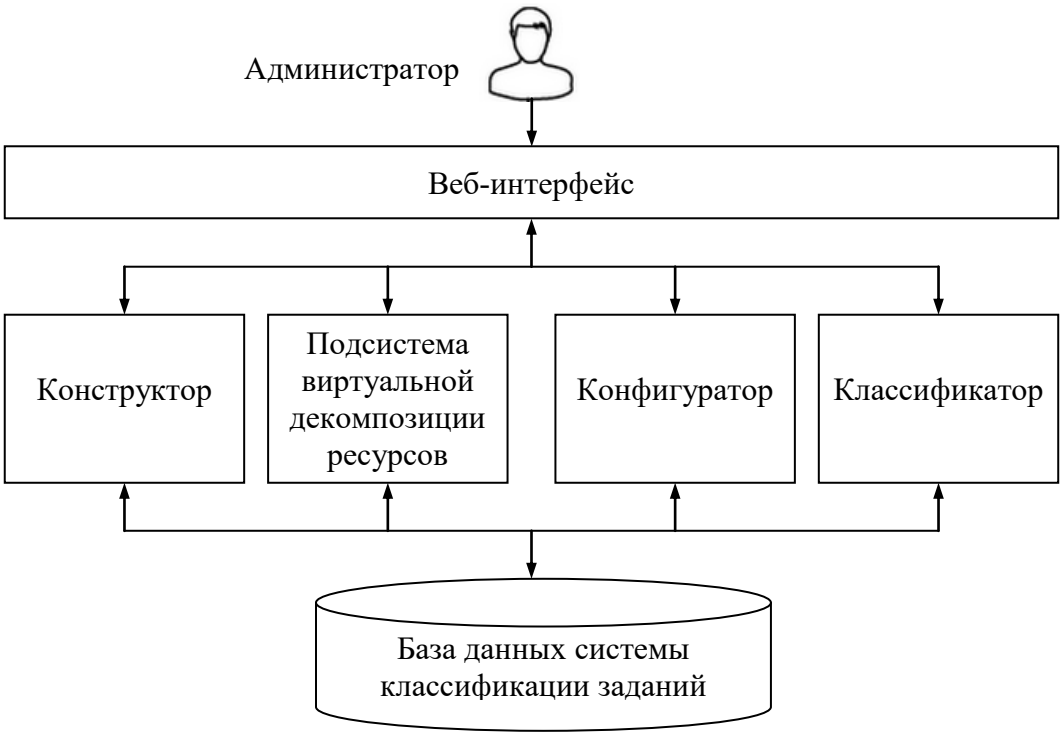


Рисунок 3.4 – Схема взаимодействия компонентов системы классификации заданий

База данных – компонент классификатора заданий, ответственный за хранение всех необходимых данных (характеристики заданий, классы и т.д.).

3.8. Методика классификаций заданий и их потоков

Предложенная в диссертационной работе методика классификации заданий и их потоков включает два основных этапа: построение и применение системы классификации. На первом этапе осуществляется разработка множества характеристик заданий, формирование множества классов заданий на основе подмножеств выбранных характеристик и определение соответствия классов заданий и ресурсов ГРВС (рисунок 3.5).

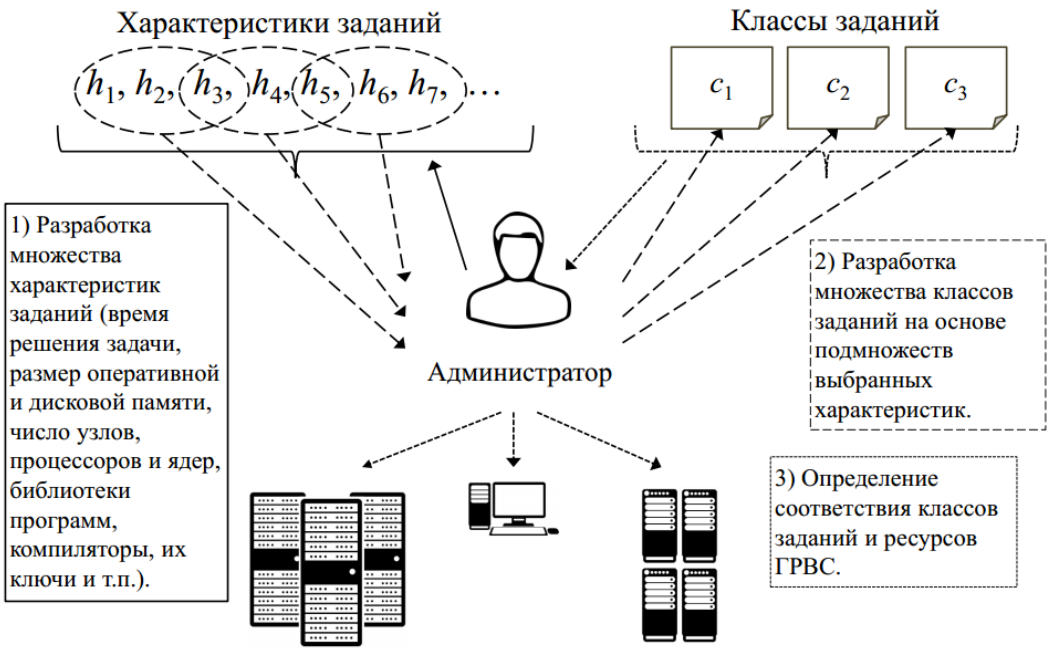


Рисунок 3.5 – Схема построения системы классификации

На следующем этапе выполняется классификация поступающих в ГРВС заданий и их конкретизация. Средства конкретизации заданий разработаны для более детальной настройки требований к ГРВС, содержащихся в заданиях. Схема взаимодействия данных средств между собой и с СУПЗ представлена на рисунке 3.6.

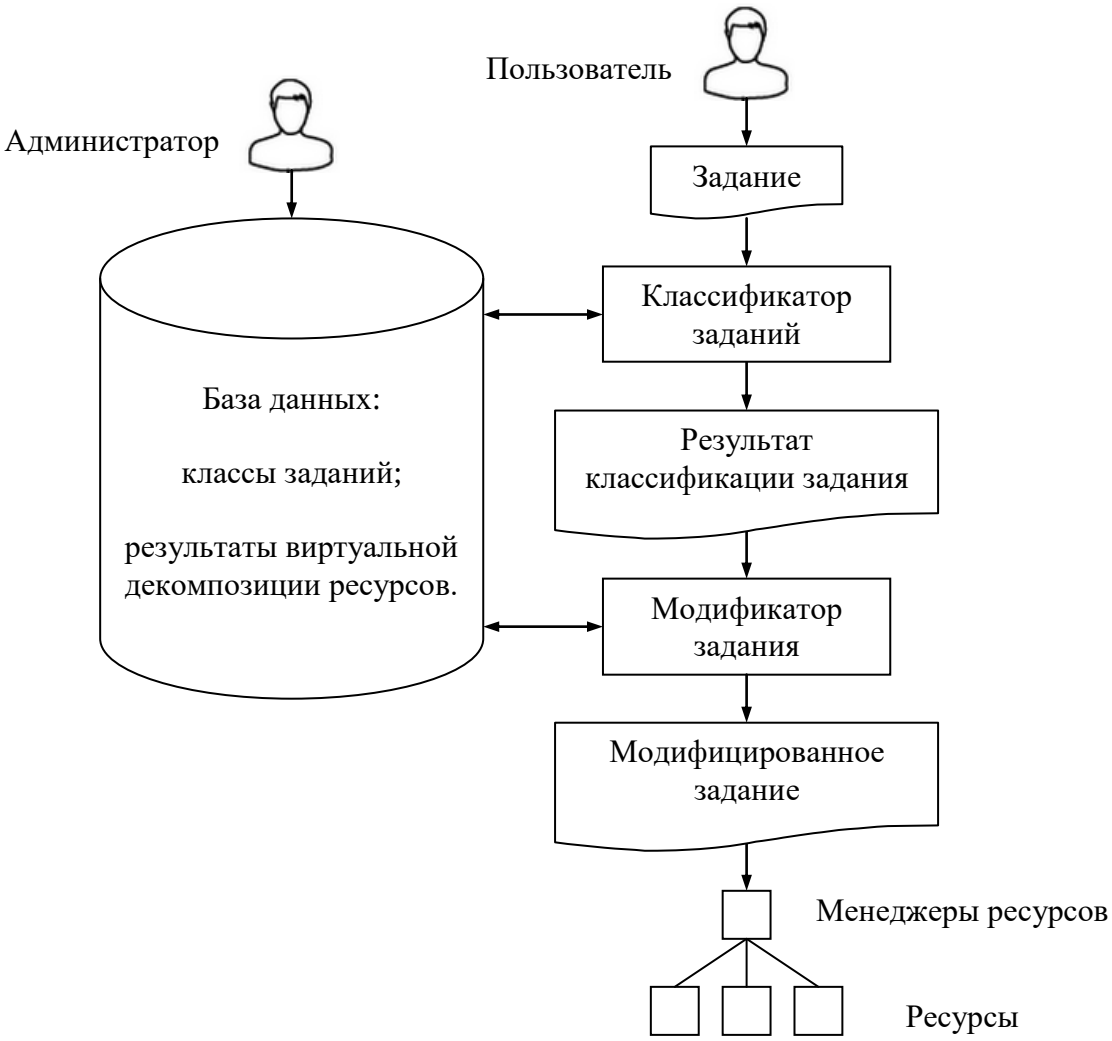


Рисунок 3.6 – Схема взаимодействия систем виртуальной декомпозиции ресурсов и классификации заданий

Система виртуальной декомпозиции ресурсов позволяет администратору ГРВС классифицировать задания, которые могут поступать в эту среду, по их характеристикам и определить для каждого ресурса этой среды те классы заданий, которые по своим характеристикам наиболее подходят вычислительным возможностям этих ресурсов. Результат виртуальной декомпозиции ресурсов сохраняется в базе данных.

Система классификации заданий осуществляет перехват заданий, поступающих в ГРВС. Данная система включает две основные подсистемы: классификатор задания и модификатор задания. Классификатор задания выполняет унификацию задания и определение дополнительных его характеристик,

извлекаемых из переменных окружения вычислительной среды, конфигурационных параметров СУПЗ или статистических файлов (системных и/или пользовательских). Затем он определяет класс задания и передает исходное задание и результат его классификации модификатору задания.

Модификатор задания добавляет в задание дополнительные параметры, определяющие множество ресурсов, соответствующих классу задания, и передает модифицированное задание в СУПЗ. В дальнейшем планировщик СУПЗ будет выбирать ресурсы для выполнения задания только из множества ресурсов, заданного дополнительными параметрами модифицированного задания.

Проведенные эксперименты по применению средств конкретизации вычислительных заданий в процессе управления ГРВС показывают возможность существенного улучшения целого ряда важных показателей ее функционирования, таких как характеристики обслуживания очередей заданий, коэффициенты надежности вычислений, эффективности полезного использования вычислительных ресурсов (см., например, [68, 79, 382]).

3.9. Выводы

В третьей главе получены следующие основные результаты:

- выполнен обзор известных подходов к классификации вычислительных задач и заданий;
- разработана система классификации заданий в ГРВС;
- предложена модель концептуализации и классификации заданий в ГРВС, включающая специализированные модели и методы представления и использования знаний о заданиях и ресурсах, а также технологию практического применения этих моделей и методов;
- с целью исследования преимуществ применения предложенной системы создан имитационный прототип классификатора заданий в ГРВС;
- реализован действующий классификатор заданий в виртуализированной ГРВС, организованной на базе ресурсов ЦКП ИСКЦ;

– проведены модельные и практические вычислительные эксперименты, продемонстрировавшие достоинства применения предложенных в диссертации модели и системы классификации заданий в ГРВС.

Представленная система классификации заданий позволяет расширить функциональные возможности СУПЗ и средств виртуализации ГРВС и осуществлять оптимизацию распределения вычислительных ресурсов среды не только на уровне отдельно взятого задания, но также на уровне целого класса заданий. Использование представленных средств классификации заданий в качестве надстройки к СУПЗ позволяет существенно смягчить неопределенность относительно свойств заданий и улучшить результаты распределения ресурсов.

Результаты исследований, представленные в данной главе, опубликованы в [68, 79, 348, 353, 374, 382, 390].

Глава 4. Мультиагентное управление в гетерогенной распределенной вычислительной среде

Технологии облачных и грид-вычислений ориентированы на создание и применение открытых распределенных систем. В их рамках разрабатываются инфраструктура, интерфейсы, инструментальные средства и приложения для удобного, надежного и безопасного использования разделяемых ресурсов в динамических и географически распределенных виртуальных организациях. В то же время мультиагентные технологии, поддерживая разработку и функционирование распределенных систем, направлены на интеллектуализацию автономных программных сущностей, которые могут представлять интересы субъектов виртуальных организаций, а также гибко действовать в динамичной среде в условиях неопределенности.

В рамках мультиагентных технологий агент представляет собой аппаратную или программную сущность, способную действовать в интересах достижения целей, поставленных перед ним субъектами виртуальной организации [219]. Он может выступать в качестве посредника между человеком (владельцем ресурсов, их пользователем, разработчиком приложений, администратором среды или другим субъектом), другими программами и внешней средой [13].

Перспективной моделью взаимодействия агентов является их самоорганизация [254, 268]. Под самоорганизацией в технической системе (к которой относятся как Грид, так и облачная инфраструктура) понимается процесс автономного формирования оптимальной структуры и оптимального алгоритма ее функционирования в соответствии с поставленной перед системой целью, некоторым критерием качества и внешними условиями [267]. В условиях распределения ограниченных разделяемых ресурсов системы, когда агентам известна лишь локальная информация о вычислительном процессе, перспективной моделью самоорганизации является использование рыночных механизмов [268].

Наряду с децентрализованностью и наличием сетевого взаимодействия МАС могут характеризоваться сложной иерархической структурой, на каждом уровне

которой имеется разнообразие моделей и методов, которые агенты могут применять для решения той или иной задачи многоуровневого управления [289].

Анализ текущего состояния исследований [89, 113, 171], связанных с вышеупомянутыми направлениями создания распределенных систем, показывает, что наблюдается тенденция сближения мультиагентных систем, нуждающихся в надежной программно-аппаратной инфраструктуре и стандартизированных интерфейсах, с Грид и облачными средами, в которых существует насущная потребность в автономном гибком поведении элементов их управляющих систем с целью повышения эффективности использования ресурсов, а также выполняемых с их помощью вычислений.

В четвертой главе проведен сравнительный анализ известных мультиагентных методов и средств, используемых для управления распределенными вычислениями. Разработаны модели, алгоритмы и методы мультиагентного управления в ГРВС.

4.1. Анализ методов и средств организации мультиагентного управления распределенными вычислениями

В настоящее время мультиагентное управление распределенными вычислениями является чрезвычайно актуальным направлением исследований в мировой и отечественной литературе [47, 114, 120, 161, 286, 313]. Известны различные агентно-ориентированные методы и средства управления распределенными вычислениями в ГРВС [8, 54], которые зачастую показывают определенные преимущества по сравнению с традиционными системами управления [135, 196].

Поскольку направление является новым, в нем, как правило, развиваются эвристические методы управления вычислительными ресурсами на основе экономических механизмов регулирования спроса и предложения этих ресурсов [27, 28, 94, 152, 288]. Однако используемые на практике МАС зачастую представлены смышленными агентами [159], не обладающими всеми необходимыми свойствами интеллектуальных агентов [250].

В настоящее время ряд МАС разрабатывается и успешно применяется для управления распределенными вычислениями [11, 123, 126]. Анализ функциональных возможностей ряда известных систем [31, 32, 33, 90, 122, 134, 173, 185, 187, 221], реализованных на основе мультиагентных технологий и используемых для управления распределенными вычислениями (таблица 12), позволяет сделать вывод о том, что необходима разработка дополнительных возможностей МАС, обеспечивающих существенное повышение качества управления. В их числе:

- планирование действий агентов;
- адаптивное обучение агентов путем параметрической настройки алгоритмов их функционирования;
- комбинированное управление, обеспечивающее планирование вычислений и распределение ресурсов агентами на локальном и глобальном уровнях вычислительной среды на основе единой информационной структуры (концептуальной модели);
- моделирование работы агентов и др.

В таблице 12 рассмотрены следующие характеристики МАС, отражающие их функциональные возможности [372]:

1) наличие встроенных агентов;
2) обеспечение высокоуровневых инструментов разработки пользовательских агентов, включая их конфигурирование и настройку на работу с предметно-ориентированными знаниями;
3) формирование процедурной постановки задачи и построение схемы ее решения;
4) формулирование непроцедурной постановки задачи и автоматический синтез схемы ее решения;
5) распределение ресурсов;
6) мониторинг среды;
7) поддержка модели Агент-как-Сервис;

8) моделирование агентами поведения других агентов с целью повышения качества принятия решений при их взаимодействии;
9) управление в рамках как локальной (например, вычислительного кластера), так и глобальной распределенной системы (например, грид-системы) с целью оптимизации распределения ресурсов с учетом всех запросов пользовательских приложений;
10) управление на уровне приложений с целью оптимизации распределения ресурсов для конкретного приложения;
11) комбинированное управление, обеспечивающее планирование вычислений и распределение ресурсов на локальном и глобальном уровнях, а также на уровне приложений на основе единой информационной структуры;
12) централизованный алгоритм взаимодействия агентов;
13) децентрализованный алгоритм взаимодействия агентов;
14) обеспечение повышенной надежности обмена сообщениями между агентами за счет использования системы логического времени;
15) применение агентами экономических механизмов управления;
16) возможность исполнения нескольких ролей одним агентом;
17) обучение агентов.

Анализ результатов исследований в области мультиагентного управления вычислительными системами (см., например, работы [2, 3, 39, 49, 51, 103, 141, 170, 181, 182, 184, 254, 255, 267, 268]) показывает, что самоорганизация позволяет существенно улучшить качество принятия решений и их результатов за счет формирования оптимальных алгоритмов функционирования и конфигураций МАС, которые определяются заданной общей целью, критериями качества и особенностями предметной области решаемых задач пользователей, а также условиями внешней среды.

Таблица 12 – Функциональные возможности МАС

Система	Характеристика																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Application Level Scheduling (AppLes) [134]	+	–	+	–	+	+	–	–	–	+	–	+	–	–	–	–	–
Agent-based Resource Management System (ARMS) [32]	+	–	+	–	+	+	+	–	+	–	–	–	+	–	–	–	–
Condor-G [90]	+	–	+	–	+	+	–	–	–	+	–	+	–	–	+	–	–
GridFlow [33]	+	–	–	–	+	+	–	+	+	+	+	+	–	–	–	–	–
GridSolve [221]	+	–	+	–	+	+	–	–	–	+	–	+	–	–	–	–	–
Multi-Agent Architecture for Grid Environment (MAAG) [173]	+	–	+	–	+	+	–	–	–	+	–	–	+	–	–	–	–
Agent-oriented programming environment (MAGE) [185]	+	+	+	–	+	+	+	+	–	+	–	–	+	–	–	–	+
Nimrod/G	+	–	+	–	+	+	+	–	–	+	–	+	–	–	+	–	–
Agent Based Framework [187]	+	–	+	–	+	+	+	–	+	–	–	+	–	–	+	–	–
YAKA [122]	+	–	–	–	+	+	–	–	–	–	–	–	+	–	–	–	–
High-Performance Computing Service-Oriented Multi-Agent System (HPCSOMAS) [305]	+	+	+	+	+	+	+	–	–	+	–	+	+	–	–	–	–

4.2. Мультиагентная система

Предлагаемая иерархическая структура МАС может включать два или более уровней агентов. Развитие этой структуры отражено в [83, 326, 333, 334, 354, 359, 360, 364, 366, 370, 372, 377].

На каждом уровне агенты играют различные роли и соответственно выполняют различные функции. Роли агентов носят постоянный или временный характер и возникают в дискретные моменты времени в связи необходимостью организации коллективного взаимодействия в процессе решения задачи, поставленной пользователем ГРВС. Уровни иерархии агентов отличаются объемом их знаний: агенты более высокого уровня иерархии обладают большим объемом знаний по сравнению с агентами более низкого уровня иерархии и, кроме того, обращаются к агентам ниже лежащих уровней с запросами на получение локальных знаний этих агентов. На каждом уровне иерархии агенты могут объединяться в виртуальные сообщества, кооперироваться и конкурировать в рамках этих сообществ.

МАС включает агентов постановки задачи, планирования вычислений, мониторинга и распределения ресурсов, классификации, конкретизации и выполнения заданий, а также агентов, управляющих агентами распределения ресурсов (рисунок 4.1). Работа агентов основывается на комплексном использовании знаний, представленных в АМ ГРВС.

Работа МАС базируется на основных принципах самоорганизации [254, 255]. В их числе автономность (готовность взаимодействовать с окружающей средой, управлять собственным поведением, направленным на достижение своих локальных целей без внешнего вмешательства), способность восприятия внешней среды и ее проявлений, а также локального воздействия на нее, распределенное взаимодействие с другими агентами системы, достижение глобального порядка в системе на основе локальных взаимодействий между ее агентами, обеспечение эмерджентности (возникновения определенных свойств системы в процессе ее функционирования, не присущих ее отдельным агентам), иерархичность, большое

число агентов системы, ее нечувствительность к отказам отдельных агентов, простота правил взаимодействия агентов в процессе сложного поведения системы и ее адаптивность (изменение своего поведения при смене организационно-функциональной структуры).

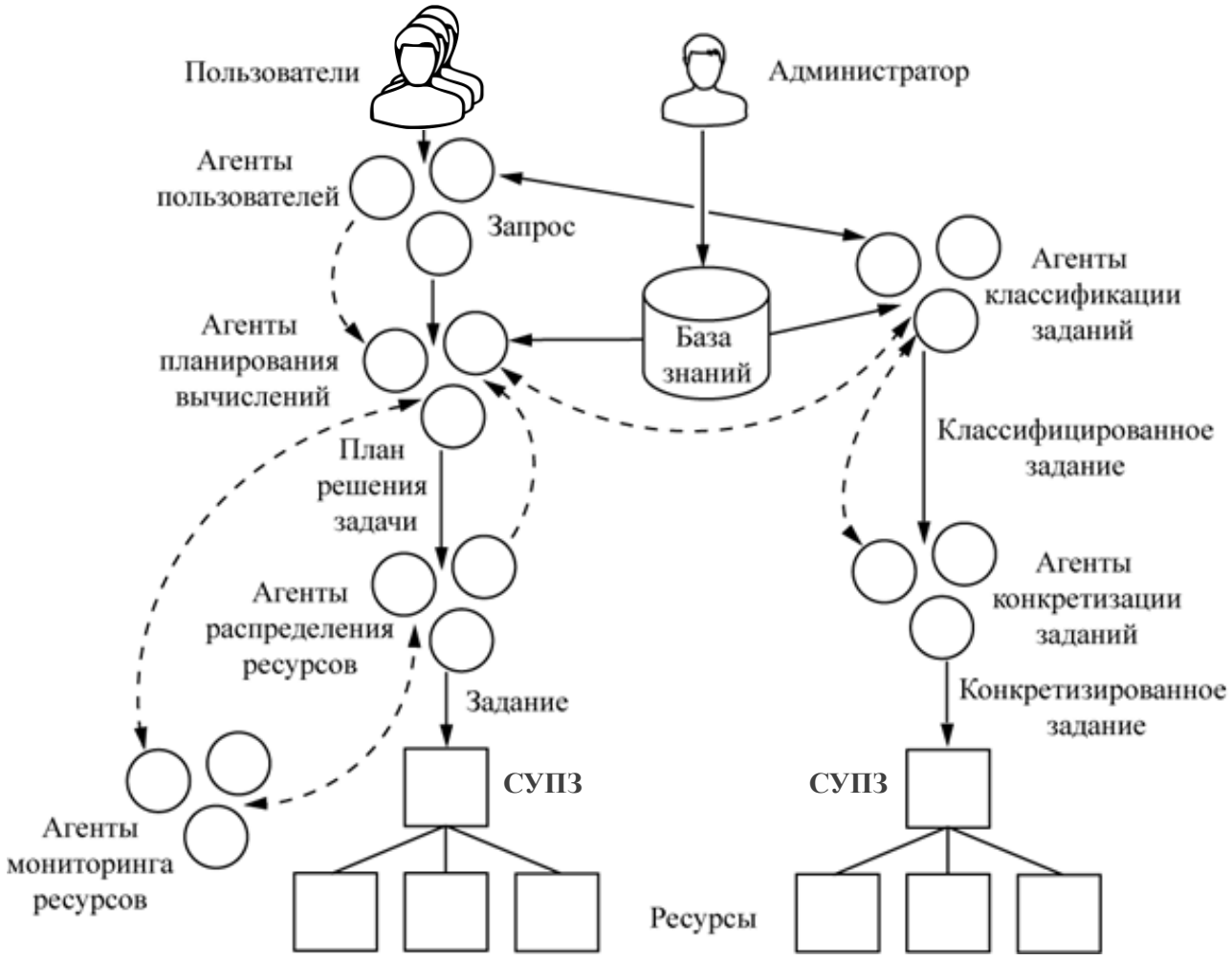


Рисунок 4.1 – Структура мультиагентной системы

МАС обеспечивает выполнение функциональных возможностей 1-9 и 13-17 (таблица 12). В процессе интеграции с системами управления вычислениями в инструментальных комплексах DISCOMP и Orlando Tools она может поддерживать функции 10 и 11. Поддержка функции 12 является нецелесообразной из-за известных недостатков централизованного управления вычислениями в ГРВС.

4.3. Постановка задачи мультиагентного управления заданиями

Пусть на модели $\mathbb{M}$ построена схема решения задачи, представленная матрицей **S**, описывающей схему решения задачи в виде ярусно-параллельной формы, и матрицей **P**, отражающей сведения о предшествовании операций схемы решения задачи (см. разделы 2.2 и 2.3 диссертации). Для операций $f_1, f_2, \dots, f_{n\_o}$, входящих в схему решения задачи, найдены оценки времени их выполнения на эталонном узле ГРВС. Значения параметров передаются между операциями в виде файлов данных, для которых получены оценки их размера.

Агентам необходимо распределить свои ресурсы для модулей, реализующих операции схемы решения задачи.

Процесс вычислений должен удовлетворять заданным критериям $c_1, c_2, \dots, c_{n_c}$ качества решения задачи. Для каждого критерия приведены его предельные значения $c_l^{min} \geq 0$ и $c_l^{max} \geq 0$ такие, что $c_l^{min} \leq c_l \leq c_l^{max}$, а также указано условие его оптимальности

$$c_l \to c_l^{min}(c_l^{max}), \tag{28}$$

где $l = \overline{1, n_c}$. Предельные значения и условия оптимальности критериев пользователя определяются им при формулировке постановки задачи. Предельные значения и условия оптимальности предпочтений владельцев ресурсов задаются администратором, представляющим данные ресурсы.

Задание по решению задачи включает набор взаимосвязанных подзаданий разных классов. Постановки задач мультиагентного управления, отражающие данную формулировку в той или иной степени, приведены в [80, 83, 325, 362, 363].

4.4. Тендер вычислительных работ

Наблюдающаяся в настоящее время тенденция превращения вычислительных ресурсов в товар, как, например, в рамках облачных вычислений, актуализирует разработку и применение экономических механизмов (правил) регулирования их спроса и предложения. Широко используемым на практике

подходом к регулированию спроса и предложения товаров и услуг является применение различных аукционов и тендеров.

Аукцион представляет собой форму публичной продажи каких-либо объектов (лотов), например, товаров и услуг, по заранее определенным правилам на основе конкуренции между потенциальными покупателями этих товаров и услуг. Субъект, получивший право на приобретение лота в соответствии с правилами проведения аукциона, становится его победителем.

Тендер является конкурентной формой отбора предложений на поставку товаров, оказание услуг или выполнение работ в соответствии с заранее установленными условиями. В отличие от аукционов, где единственным критерием является стоимость лота, тендер позволяет определить дополнительные условия выполнения работ. Победителем тендера становится его участник, чья заявка содержит наилучшие предложения, соответствующие установленным условиям.

Тендер может быть организован в виде аукциона или конкурса. В обоих случаях участники тендера подают заявки на выполнение работ. В первом случае распределение заявок осуществляется по результатам торгов (сравнение агрегированных показателей заявок, процесс которого легко формализуется и автоматизируется). Существует большое разнообразие готовых решений по проведению стандартных аукционов. Во втором случае требуется выполнение дополнительных сложных этапов, включая разработку условий проведения конкурса, организацию конкурсной комиссии и экспертную оценку заявок на проведение работ. Очевидно, что ввиду меньшей сложности и более низких накладных затрат на проведение тендера его проще реализовать на основе аукциона.

В диссертационной работе используется тендер [83, 362], в рамках которого в качестве покупателя выступает пользователь ГРВС, предлагающий вычислительные работы (задания), а в качестве продавцов услуг – представители (агенты) вычислительных ресурсов среды, претендующие на выполнение работ пользователя.

Исходя из сформулированной выше постановки задачи мультиагентного

управления, можно определить следующие требования к модели тендера вычислительных работ:

- в рамках торгов предлагается вычислительная работа – выполнение набора взаимосвязанных подзаданий различных классов для решения задачи пользователя ГРВС;
- пользователь задает критерии качества выполнения схемы решения задачи (стоимость, время выполнения, надежность, безопасность и другие условия);
- ресурсы представляются агентами системы управления вычислениями ГРВС;
- агенты должны иметь возможность выполнить одно или несколько подзаданий;
- агенты оперируют общими сведениями о схеме решения задачи, ее модулях и информационно-логических связях между ними, а также своими локальными знаниями о характеристиках представляемых ими ресурсов;
- по итогам тендера должно быть найдено некоторое рациональное распределение ресурсов (предоставление услуг) для выполнения каждого подзадания;
- принятие решения должно проходить с минимальными затратами на проведение торгов.

Результаты сравнительного анализа [14, 15, 27, 43, 100] различных моделей аукционов показывают, что наиболее востребованными, обсуждаемыми и широко применимыми на практике являются следующие аукционы [42]: английский (прямой или обратный) [156], закрытый аукцион первой цены [105], голландский [16] и закрытый аукцион Викри второй цены [210]. Принципы их работы детально представлены в [37, 148].

Каждый из четырех вышеперечисленных стандартных аукционов отдает выставленный на торги лот участнику, который ценит его больше всех. Несмотря на то, что на всех четырех аукционах результат торгов выбирается из множества Парето-оптимальных решений, аукцион Викри и английский аукцион, реализующие доминирующие стратегии, являются более эффективными в том

смысле, что никакие новые предложения других участников торгов не могут изменить ситуацию [179].

В реальном мире английский аукцион является на практике одним из самых популярных. Он также часто применяется и в вычислительных системах для распределения ресурсов (см., например, [52, 130]) при выполнении независимых заданий или предварительно сформированных пакетов заданий, рассматриваемых в качестве неделимых лотов. Зачастую специалисты, поработав с каким-либо аукционом, например, закрытым аукционом первой цены [230], переходят в дальнейшем на использование английского аукциона [314].

Однако при распределении ресурсов для схем решения задач (workflow) в рамках английского аукциона возникают дополнительные накладные расходы, связанные с организацией отдельных торгов по каждому подзаданию схемы, и необходимость многокритериального выбора оптимальных временны́х ограничений на проведение торговых раундов с целью обеспечения, с одной стороны, равных условий ожидания ставок и отправки ответов для всех участников аукциона, которые могут находиться на разном расстоянии от аукционера и их интерконнект может иметь различные характеристики быстродействия и степень загрузки, и, с другой стороны, быстрого принятия решений по распределению ресурсов. Другой проблемой, зачастую неразрешимой в полной мере в рамках данного аукциона, является учет взаимосвязей между подзаданиями в случае, когда стоимость выполнения некоторого набора подзаданий может отличаться от суммарной стоимости выполнения этих подзаданий в отдельности.

В рамках аукциона Викри проведение торгов осуществляется за один раунд. Все участники аукциона равноправны и действуют по единым правилам. Они делают свои закрытые (неизвестные другим участникам торгов) предложения по лоту. Все предложения рассматриваются организатором торгов. Побеждает участник, представивший наилучшие предложения. Победитель получает лот на условиях, предложенных участником, оказавшимся вторым, по результатам торгов. Особенностью аукциона Викри является то, что каждый его участник формирует предложение по лоту, отражающее истинную ценность лота и

максимальную полезность сформированного предложения для этого участника. Согласованное устойчивое состояние участников аукциона достигается по окончании торгов. В [288] показано, что это состояние в определенной степени является аналогом равновесия по Нэшу в теоретико-игровых моделях [154].

Доходность участников в рамках английского аукциона могут несущественно превышать соответствующие доходы в рамках аукциона Викри.

Утверждение 1. Разница в доходах аукционера в рамках аукциона Викри и прямого английского аукциона с фиксированным значением δ увеличения ставки не превышает этого значения.

Доказательство. Пусть ν_1 и ν_2 $(\nu_1 < \nu_2)$ – это истинные ценности лота для агентов a_1 и a_2, а δ – фиксированное значение увеличения ставки в процессе торгов.

Тогда в рамках английского аукциона агент a_2 станет победителем со ставкой $\nu_2^* = \nu_1^* + \delta$, где $\nu_2^* \leq \nu_2$, а $\nu_1^* \leq \nu_1$ – ставка агента a_1, с которой он завершит участие в торгах. Итоговая стоимость лота (доход аукционера) равна ν_2^*.

В рамках аукциона Викри агент a_1 завершит торги со ставкой ν_1. Агент a_2 станет победителем со ставкой ν_2, так как они правдиво отражают свои истинные ценности. В соответствии с правилом второй цены итоговая стоимость лота (доход аукционера) равна ν_1.

По условиям формирования ставок $\nu_1^* \leq \nu_1$. Тогда

$$\nu_2^* - \nu_1 \leq \nu_2^* - \nu_1^*. \quad (29)$$

Так как $\nu_2^* = \nu_1^* + \delta$, то после подстановки данного выражения для ν_2^* в правой части неравенства (29) получим

$$\nu_2^* - \nu_1 \leq \delta \blacksquare$$

То же самое справедливо для обратного английского аукциона, по предоставлению, например, услуг.

Утверждение 2. Разница в доходах агентов предоставления услуг в рамках аукциона Викри и обратного английского аукциона с фиксированным значением δ уменьшения ставки не превышает этого значения.

Доказательство. Пусть v_1 и v_2 $(v_2 < v_1)$ – это истинные ценности лота для агентов a_1 и a_2, а δ – фиксированное значение уменьшения ставки в процессе торгов.

Тогда в рамках английского аукциона агент a_2 станет победителем со ставкой $v_2^* = v_1^* - \delta$, где $v_2^* \geq v_2$, а $v_1^* \geq v_1$ – ставка агента a_1, с которой он завершит участие в торгах. Итоговая стоимость услуги (доход агента a_2) равна v_2^*.

В рамках аукциона Викри агент a_1 завершит торги со ставкой v_1. Агент a_2 станет победителем со ставкой v_2, так они правдиво отражают свои истинные ценности. В соответствии с правилом второй цены итоговая стоимость услуги (доход агента a_2) равна v_1.

По условиям формирования ставок $v_1 \leq v_1^*$. Тогда

$$v_1 - v_2^* \leq v_1^* - v_2^*. \quad (30)$$

Так как $v_2^* = v_1^* - \delta$, то после подстановки полученного выражения для v_2^* в правой части неравенства (30) получим

$$v_1 - v_2^* \leq \delta \blacksquare$$

Комбинаторный аукцион Викри [210] предоставляет возможность формировать предложения по различным комбинациям лотов. Обширная библиография подтверждает возможность применения различных форм этого аукциона применительно к распределенным вычислениям [50, 108, 116, 131, 136, 157, 168, 199, 224].

Более того, в некоторых случаях применение комбинаторного аукциона дает определенное преимущество по сравнению с английским аукционом. Пусть два агента a_1 и a_2 участвуют в торгах за право выполнения подзаданий j_1 и j_2, которые обрабатывают один и тот же набор данных d. Стоимость выполнения подзадания j_1 (j_2) складывается из стоимости v_1 (v_2) выполнения модуля, указанного в подзадании, и стоимости v_d передачи данных. Истинные стоимости v_1, v_2 и v_d выполнения подзаданий j_1 и j_2 и получения данных d агентами a_1 и a_2 приведены в таблице 13.

Таблица 13 – Результаты распределения ресурсов на основе английского аукциона

	Стоимость выполнения				Победитель		Итоговая стоимость
Агент	j_1		j_2		j_1	j_2	
	v_1	v_d	v_2	v_d			
a_1	1.7	1.0	1.2	1.0	+	–	4.8
a_2	1.8	1.0	1.1	1.0	–	+	

На основе обратного английского аукциона, который в большей степени соответствует тендеру вычислительных работ по сравнению с прямым английским аукционом, агенты делают свои предложения, уменьшая ставки на фиксированную величину 0.1. Они перестают делать ставки, достигнув своих истинных стоимостей.

В данном примере агент a_1 становится победителем за право выполнения задания j_1 с общей стоимостью услуг 2.7. Второй агент a_2 получает право на предоставление услуг по выполнению задания j_2 на сумму 2.1. Итоговая сумма выполнения двух заданий составляет 4.8. Порядок предложения заданий на торги не влияет на их конечный результат.

Комбинаторный аукцион, в рамках которого можно сделать предложение на совместное выполнение подзаданий, позволяет улучшить полученный результат за счет учета сокращения накладных расходов на передачу данных при выполнении заданий одним агентом. В таблице 14 представлены результаты распределения ресурсов на основе комбинаторного аукциона, где появились дополнительные ставки на совместное выполнение заданий одним агентом. В этом случае итоговая сумма составляет 3.9. Такая сумма предложена обоими агентами. Поэтому победитель будет определен по времени поступления заявки или с учетом дополнительных критериев (времени выполнения заданий, надежности вычислений и других условий).

Таблица 14 – Результаты распределения ресурсов на основе комбинаторного аукциона

Агент	Стоимость выполнения					Победитель			Итоговая стоимость
	j_1		j_2		j_1, j_2	j_1	j_2	j_1, j_2	
	v_m^1	v_d	v_m^2	v_d	$v_m^1 + v_m^2 + v_d$				
a_1	1.7	1.0	1.2	1.0	3.9	–	–	+	3.9
a_2	1.8	1.0	1.1	1.0	3.9	–	–	–	

Таким образом, комбинаторный аукцион Викри и английский аукцион имеют свои преимущества и недостатки. Однако комбинаторная природа постановки задачи мультиагентного управления обуславливает выбор в пользу первого аукциона. При этом основным недостатком комбинаторного аукциона Викри, как и всех других типов комбинаторных аукционов, является вычислительная сложность принятия решения.

В диссертационной работе разработаны модели тендера вычислительных работ, базирующиеся на использовании комбинаторного аукциона Викри, модифицированного путем реализации следующих решений с целью существенного снижения вычислительной сложности:

– предварительной классификации заданий, обеспечивающей формирование виртуального сообщества агентов (участников тендера), каждый из которых удостоверен в наличии и заинтересован в выполнении соответствующих его ресурсам подзаданий вычислительной работы;
– разбиения схемы решения задачи, представленной ярусно-параллельной формой, на подсхемы, каждая из которых включает один ярус схемы, и проведения тендера для каждого яруса в отдельности, что позволяет агентам учитывать в процессе торгов для подзаданий некоторого яруса результаты распределения ресурсов для выполнения подзаданий предыдущих ярусов;

– использования параллельных операций, обеспечивающих долевое распределение вычислительной нагрузки по выполнению реализующих их модулей и, таким образом, исключающих необходимость перебора комбинаций их экземпляров при формировании ставок агентами;
– введения специальных ограничений на процесс формирования предложений по выполнению наборов подзаданий с целью исключения из рассмотрения тех их сочетаний, которые заведомо не позволяют уменьшить время выполнения этих подзаданий в отдельности с учетом их информационно-логических связей в рамках схемы решения задачи;
– реализации эвристического подхода к использованию сформированных предложений по выполнению наборов подзаданий.

С учетом вышеперечисленных решений максимальное число возможных комбинаций операций на каждом уровне схемы решения задач как для типовых рабочих процессов (Приложение Д), так и схем решения практических задач, представленных в главах 5 и 6, изменяется от одной комбинации до нескольких десятков комбинаций.

Для проведения и оценки результатов тендера используется виртуальная стоимость ресурсов. В качестве такого показателя (или наряду с ним) могут выступать другие показатели: пользовательские критерии качества решения задачи, предпочтения владельцев ресурсов и другие характеристики, включая стоимость процессорного времени, передачи и хранения данных. Предполагается, что стоимостные характеристики ресурсов подобраны таким образом, что стоимость выполнения одной и той же вычислительной работы на более производительном ресурсе не может быть выше стоимости ее выполнения на менее производительном ресурсе.

4.5. Модели тендера вычислительных работ

Предложены три модели тендера вычислительных работ, поддерживающие три уровня обслуживания на основе мультиагентного управления [77]:

1) модель, ориентированная на пользовательские критерии качества решения задачи,
2) модель, направленная на удовлетворение предпочтений владельцев ресурсов,
3) модель, поддерживающая согласование пользовательских критериев качества решения задачи с предпочтениями владельцев ресурсов.

Каждый агент (участник торгов) делает предложения по выполнению отдельных подзаданий, класс которых соответствует ресурсам данного агента. Предложение выражается в виде ставки, элементы которой отражают критерии $c_1, c_2, \dots, c_{n_c}$ применительно к подзаданию.

Дополнительно любой агент может сделать предложения по выполнению наборов подзаданий, классы которых соответствуют его ресурсам. Такое предложение осуществляется в виде ставки, элементы которой отражают критерии $c_1, c_2, \dots, c_{n_c}$ применительно к каждому подзаданию, входящему в данный набор.

Введем следующие обозначения:

– n_{job} – число подзаданий;

– $B_i^j = \left(b_{i1}^j, b_{i2}^j, \dots, b_{in_c}^j\right)$ – i-я ставка на выполнение j-го подзадания в отдельности, сделанная ξ_i-м агентом, $j \in \overline{1, n_{job}}$;

– $b_{il}^j = \varphi\left(\xi_i, j, k, l, \mathbf{S}, \mathbf{P}, S_{\xi_i}\right)$ – элемент ставки B_i^j, относящийся к l-му критерию выполнения j-го подзадания, $l \in \overline{1, n_c}$;

– S_{ξ_i} – расписание обслуживания заданий ξ_i-м агентом;

– I_j – множество индексов агентов, подавших предложения по j-му подзаданию;

– $\tilde{B}_\iota = \left(\tilde{b}_{\iota 1}^{j_1}, \tilde{b}_{\iota 2}^{j_1}, \dots, \tilde{b}_{\iota n_c}^{j_1}, \tilde{b}_{\iota 1}^{j_2}, \tilde{b}_{\iota 2}^{j_2}, \dots, \tilde{b}_{\iota n_c}^{j_2}, \dots \tilde{b}_{\iota 1}^{j_{n_{\eta_\iota r_\iota}}}, \tilde{b}_{\iota 2}^{j_{n_{\eta_\iota r_\iota}}}, \dots, \tilde{b}_{\iota n_c}^{j_{n_{\eta_\iota r_\iota}}}\right)$ – ι-я ставка на совместное выполнение подзаданий $j_1, j_2, \dots, j_{n_{\eta_\iota r_\iota}}$, сделанная η_ι-м агентом, который включил их в свой r_ι-й набор ($0 \le r_\iota \le n_{max\_set}$, где n_{max_set} – максимально допустимое число наборов подзаданий);

– $n_{\eta_\iota r_\iota}$ – число подзаданий в r_ι-м наборе η_ι-го агента;

– $\tilde{b}_{il}^{j} = \tilde{\varphi}\left(\xi_i, j, k, l, b_{il}^{j}, J_{\eta_\iota r_\iota}, \mathbf{S}, \mathbf{P}, S_{\eta_\iota}\right)$ – элемент ставки $\tilde{B}_\iota$, относящийся к l-му критерию выполнения j-го подзадания, такой, что

$$\left(\left(\forall \xi_i = \eta_\iota\ \tilde{b}_{il}^{j} \le b_{il}^{j}\right) \wedge \left(\exists k \in \overline{1, n_c}: \tilde{b}_{ik}^{j} < b_{ik}^{j}\right), c_l \to c_l^{min}\right) \vee$$

$$\vee \left(\left(\forall \xi_i = \eta_\iota\ b_{il}^{j} \le \tilde{b}_{il}^{j}\right) \wedge \left(\exists k \in \overline{1, n_c}: b_{ik}^{j} < \tilde{b}_{\iota k}^{j}\right), c_l \to c_l^{max}\right);$$

– $J_{\eta_\iota r_\iota}$ – множество индексов подзаданий, входящих в r_ι-й набор η_ι-го агента;

– S_{η_ι} – расписание обслуживания заданий ξ_i-м агентом;

– I_s – множество агентов, сделавших предложения по выполнению наборов подзаданий;

– $n_a \ge 1$ – число агентов виртуального сообщества, выполняющих общее задание;

– $n_c \ge 1$ – число критериев выполнения общего задания;

– $\xi_1, \xi_2, \dots, \xi_{n_j}, \eta_1, \eta_2, \dots, \eta_{m_j} \in \overline{1, n_a}$ – индексы агентов;

– $1 \le n_j \le n_a$ $(1 \le m_j \le n_a)$ – число агентов виртуального сообщества, сделавших ставки по выполнению j-го подзадания в отдельности (в составе набора заданий).

Реализация функций φ и $\tilde{\varphi}$ на модели $\mathbb{M}$ для вычисления ставок b_{il}^{j} и $\tilde{b}_{il}^{j}$ базируется на подходах к нахождению оценок критериев, рассмотренных в разделе 2.3 диссертации.

Тендер проводится в два этапа. На первом этапе осуществляется предварительное определение победителя по каждому подзаданию в отдельности. При наличии ставок на наборы подзаданий по возможности производится улучшение результатов, полученных на первом этапе, с учетом данных ставок.

Этап I. Предварительное выявление победителя по каждому подзаданию в отдельности.

Модель 1. В данной модели критерий c_1 – это виртуальная стоимость выполнения задания. Дополнительные критерии $c_2, c_3, \dots, c_{n_c}$ упорядочены следующим образом:

– $c_2, c_3, \dots, c_i$ – предпочтения пользователей;

– $c_{i+1}, c_{i+2}, \dots, c_{n_c}$ – предпочтения владельцев ресурсов.

Множество B претендентов за право выполнения j-го задания в отдельности определяется с помощью лексикографического правила:

$$B = \left\{B_i^j : \forall e \in \overline{1, n_j}\ \exists k \in \overline{1, n_c - 1}:\right.$$
$$\left.\left(\hat{b}_{i1}^j = \hat{b}_{e1}^j\right) \wedge \left(\hat{b}_{i2}^j = \hat{b}_{e2}^j\right) \wedge \dots \wedge \left(\hat{b}_{ik}^j = \hat{b}_{ek}^j\right) \wedge \left(\hat{b}_{i(k+1)}^j < \hat{b}_{e(k+1)}^j\right)\right\},$$

$$\hat{b}_{il}^j = \begin{cases} \dfrac{b_{il}^j - b_l^{min}}{b_l^{max} - b_l^{min}} & \text{при } c_l \to c_l^{min}, \\ \dfrac{c_l^{max} - b_{il}^j}{b_l^{max} - b_l^{min}} & \text{при } c_l \to c_l^{max}, \end{cases}$$

$$b_l^{min} = \min\left\{b_{1l}^j, b_{2l}^j, \dots, b_{n_j l}^j, \tilde{b}_{1l}^j, \tilde{b}_{2l}^j, \dots, \tilde{b}_{m_j l}^j\right\},$$

$$b_l^{max} = \max\left\{b_{1l}^j, b_{2l}^j, \dots, b_{n_j l}^j, \tilde{b}_{1l}^j, \tilde{b}_{2l}^j, \dots, \tilde{b}_{m_j l}^j\right\},$$

где $\hat{b}_{il}^j$ – оценка удаленности значения элемента ставки b_{il}^j от идеальных показателей, нормированная в диапазоне [0,1], условия оптимальности c_l определены в (28), $i \in \overline{1, n_j}$, $i \neq e$, $l = \overline{1, n_c}$.

Индекс агента-победителя за право выполнения j-го задания в отдельности находится следующим образом:

$$k_j = \underset{\forall i \in \overline{1, n_j} : B_i^j \in B}{\operatorname{argmin}} \left\{t_{\xi_i}\right\}, \tag{31}$$

где t_{ξ_i} – это время поступления ставки ξ_i-го агента.

Стоимость π_j j-го подзадания вычисляется по следующей формуле:

$$\pi_j = b_{q1}^j, \tag{32}$$

$$q = \underset{\forall i \in \overline{1, n_j}, \xi_i \neq k_j}{\operatorname{argmin}} \left\{b_{i1}^j\right\}. \tag{33}$$

Модель 2. В данной модели критерий c_1 – это стоимость задания. Дополнительные критерии $c_2, c_3, \dots, c_{n_c}$ в данной модели упорядочены следующим образом:

– $c_2, c_3, \dots, c_i$ – предпочтения владельцев ресурсов;

– $c_{i+1}, c_{i+2}, \dots, c_{n_c}$ – предпочтения пользователей.

Построение множества претендентов за право выполнения j-го задания в отдельности, получение индекса k_j агента-победителя и вычисление π_j осуществляется аналогично модели 1.

Модель 3. В данной модели критерий c_1 – это виртуальная стоимость выполнения задания. Дополнительные критерии $c_2, c_3, \dots, c_{n_c}$ в данной модели не упорядочены.

Определение. Будем говорить, что ставка $B_{i_1}^j$ предпочтительнее ставки $B_{i_2}^j$ ($B_{i_1}^j \succ B_{i_2}^j$), если выполняется следующее условие:

$$\left(\forall l \in \overline{1, n_c}\ \hat{b}_{i_1 l}^j \le \hat{b}_{i_2 l}^j\right) \wedge \left(\exists k \in \overline{1, n_c}: \hat{b}_{i_1 k}^j < \hat{b}_{i_2 k}^j\right).$$

В рамках третьей модели множество B претендентов на право выполнения j-го задания в отдельности строится следующим образом:

$$B = \left\{B_i^j \in B' : \nexists k \in I_j : B_k^j \succ B_i^j\right\},$$
$$B' = \left\{B_i^j : \nexists k \in I_j : \hat{b}_{k1}^j < \hat{b}_{i1}^j\right\},\ i \in I_j.$$

Индекс агента-победителя за право выполнения j-го задания и его стоимость находятся соответственно по формулам (31)-(33).

В рамках каждой из трех моделей определяется множество ставок агентов, победивших на первом этапе, и их оценок:

$$\varpi_1^j = \pi_j,\ \hat{\varpi}_1^j = \hat{b}_{q1}^j,$$
$$\varpi_l^j = b_{il}^j,\ \hat{\varpi}_l^j = \hat{b}_{il}^j,$$
$$\forall i : \xi_i = k_j, j = \overline{1, n_{job}}, l = \overline{2, n_c}.$$

Результаты торгов, полученные на первом этапе, представим вектором ϖ, следующего вида:

$$\varpi = \left(\varpi_1^1, \varpi_2^1, \dots, \varpi_{n_c}^1, \varpi_1^2, \varpi_2^2, \dots, \varpi_{n_c}^2, \varpi_1^{n_{job}}, \varpi_2^{n_{job}}, \dots, \varpi_{n_c}^{n_{job}}\right).$$

Критерии выполнения задания на основе ставок агентов вычисляются следующим образом:

$$c_l = \phi\left(l, \mathbf{S}, \mathbf{P}, \varpi_l^1, \varpi_l^2, \dots, \varpi_l^{n_{job}}\right),\ l = \overline{1, n_c}.$$

Реализация функции ϕ на модели $\mathbb{M}$ для получения значений $c_1, c_2, \dots, c_{n_c}$ базируется на подходах к оценке критериев, рассмотренных в разделе 2.3 диссертации. Предполагается, что функция ϕ является строго возрастающей по каждой из своих переменных $\varpi_l^1, \varpi_l^2, \dots, \varpi_l^{n_{job}}$:

$$\varpi_l^{j*} > \varpi_l^j \Rightarrow$$
$$\Rightarrow \phi\left(l, \varpi_l^1, \varpi_l^2, \dots, \varpi_l^{j-1}, \varpi_l^{j*}, \varpi_l^{j+1}, \dots, \varpi_l^{n_{job}}\right) >$$
$$> \phi\left(l, \varpi_l^1, \varpi_l^2, \dots, \varpi_l^{j-1}, \varpi_l^j, \varpi_l^{j+1}, \dots, \varpi_l^{n_{job}}\right)\ \forall l \in \overline{1, n_c}.$$

Если нет предложений агентов по наборам подзаданий, то проведение тендера завершается. В противном случае начинается второй этап тендера.

Этап II. Улучшение результатов, полученных на первом этапе.

Включим в множество $\tilde{B}'$ ставок, претендующих на улучшение результатов I этапа проведения тендера, ставки $\tilde{B}_\iota$, $\forall j_k \in I_{\eta_\iota r_\iota}$, которые удовлетворяют следующим условиям:

$$\hat{\tilde{b}}_{\iota 1}^{j_k} \le \hat{\varpi}_1^{j_k}\ \ \forall j_k \in I_{\eta_\iota r_\iota}, k = \overline{1, n_{\eta_\iota r_\iota}},$$
$$\left(\exists j_k \in I_{\eta_\iota r_\iota} : \hat{\varpi}_1^{j_k} < \hat{\tilde{b}}_{\iota 1}^{j_k}\right) \vee \left(d_{\eta_\iota r_\iota}^{\varpi} > d_{\eta_\iota r_\iota}\right),$$
$$d_{\eta_\iota r_\iota} = \sum_{k=1}^{n_{\eta_\iota r_\iota}} \sum_{l=2}^{n_c} \hat{\tilde{b}}_{\iota l}^{j_k},$$
$$d_{\eta_\iota r_\iota}^{\varpi} = \sum_{k=1}^{n_{\eta_\iota r_\iota}} \sum_{l=2}^{n_c} \hat{\varpi}_l^{j_k},$$
$$\hat{\tilde{b}}_{\iota l}^{j_k} = \begin{cases} \dfrac{\tilde{b}_{\iota l}^{j_k} - b_l^{min}}{b_l^{max} - b_l^{min}} & \text{при } c_l \to c_l^{min}, \\ \dfrac{b_l^{max} - \tilde{b}_{\iota l}^{j_k}}{b_l^{max} - b_l^{min}} & \text{при } c_l \to c_l^{max}, \end{cases}$$

где $\hat{\tilde{b}}_{ll}^{j_k}$ – оценка удаленности значения элемента ставки $\tilde{b}_{ll}^{j_k}$ от идеальных показателей, нормированная в диапазоне [0,1].

Множество $\tilde{B}'$ частично упорядочено. Для его упорядочения используются значения показателей $k_{\iota 1}$, $k_{\iota 2}$ и $k_{\iota 3}$:

$$k_{\iota 1} = \sum_{k=1}^{n_{\eta_\iota r_\iota}} \left(\hat{\tilde{b}}_{\iota 1}^{j_k} - \hat{\varpi}_1^{j_k}\right),$$

$$k_{\iota 2} = d_{\eta_\iota r_\iota} - d_{\eta_\iota r_\iota}^{\varpi},$$

$$k_{\iota 3} = t_\iota,$$

где t_ι – время поступления ι-й ставки. Данные показатели перечислены в порядке уменьшения их значимости для упорядочения множества $\tilde{B}'$.

Пусть булев вектор $\tilde{\varpi}$ размерности n_{job} отражает факт использования ставок на подзадания в составе набора для улучшения результатов торгов, полученных на первом этапе. Элемент вектора $\tilde{\varpi}(j) = 1$ ($\tilde{\varpi}(j) = 0$) показывает, что ставка по подзаданию j улучшена (не улучшена). При инициализации вектора $\tilde{\varpi}$ задаются следующие значения:

$$\tilde{\varpi}(j) = 0,\ j = \overline{1, n_{job}}.$$

Первая ставка $\tilde{B}_1'$ из упорядоченного множества $\tilde{B}'$ ставок безусловно используется для улучшения результатов торгов:

$$\varpi_l^{j_k} = \tilde{b}_{1l}^{j_k},\ k = \overline{1, n_{\eta_1 r_1}},\ l = \overline{1, n_c}.$$

После этого корректируются значения вектора $\tilde{\varpi}$:

$$\tilde{\varpi}(j_k) = 1,\ k = \overline{1, n_{\eta_1 r_1}}.$$

Циклическая процедура рассмотрения остальных ставок $\tilde{B}_\iota' \in \tilde{B}'$, $\iota = \overline{2, \left|\tilde{B}'\right|}$ приведена ниже:

$$\forall \iota: \left(\forall j_k \in J_{\eta_\iota r_\iota}\ \tilde{\varpi}(j_k) = 0\right) \varpi_l^{j_k} = \tilde{b}_{ll}^{j_k},\ k = \overline{1, n_{\eta_\iota r_\iota}},\ l = \overline{1, n_c},$$

$$\tilde{\varpi}(j_k) = 1,\ k = \overline{1, n_{\eta_\iota r_\iota}}.$$

Рассмотренные выше модели тендера вычислительных работ являются развитием базовых моделей проведения торгов агентами, описанных в [72, 83, 362].

4.6. Мультиагентный алгоритм планирования вычислений и распределения заданий

Проведение торгов агентами распределения ресурсов базируется на применении алгоритма, являющегося модификацией распределенного древесного волнового алгоритма, предложенного в [312]. В модифицированном алгоритме процедура выбора координатора совмещена с передачей предложений агентов. Начало работы алгоритма инициируется пользовательским агентом. Агенты, которым соответствуют листовые вершины дерева, запускают прямую волну. Итоги торгов подводит корневой агент, который также определяет координатора виртуального сообщества. Координатор регулирует выполнение схемы решения задачи. Агенты взаимодействуют посредством посылки сообщений через коммуникационные каналы и используют дополнительные служебные сообщения (например, подтверждения о доставке сообщений). Алгоритм функционирует в фоновом режиме: агенты продолжают решение своих текущих задач при взаимодействии между собой путем обмена сообщениями. Ниже приведены основные этапы мультиагентного алгоритма А.5 планирования вычислений и распределения ресурсов.

А.5.1. *Планирование вычислений*. На основе процедурной или непроцедурной постановки задачи, сформулированной на модели $\mathbb{M}$, агент пользователя выполняет построение схемы s решения задачи с помощью алгоритмов А.1 или А.2, рассмотренных во второй главе диссертации.

А.5.2. *Формирование виртуального сообщества агентов*. Объединение агентов в виртуальное сообщество осуществляется агентом пользователя на основе знаний о соответствии классов заданий и ресурсов ГРВС. В виртуальное сообщество включаются агенты, представляющие ресурсы, на которых могут быть запущены модули, реализующие операции схемы s. Проверку соответствия классов заданий для выполнения модулей и ресурсов, предоставляемых агентами, производит агент

классификации заданий с помощью методов и средств, рассмотренных в третьей главе.

Если для каких-то модулей нет подходящих ресурсов для их выполнения, то происходит возврат на этап 1 с целью уточнения постановки задачи, перепланирования вычислений и построения новой схемы решения задачи. В противном случае осуществляется переход к этапу 3.

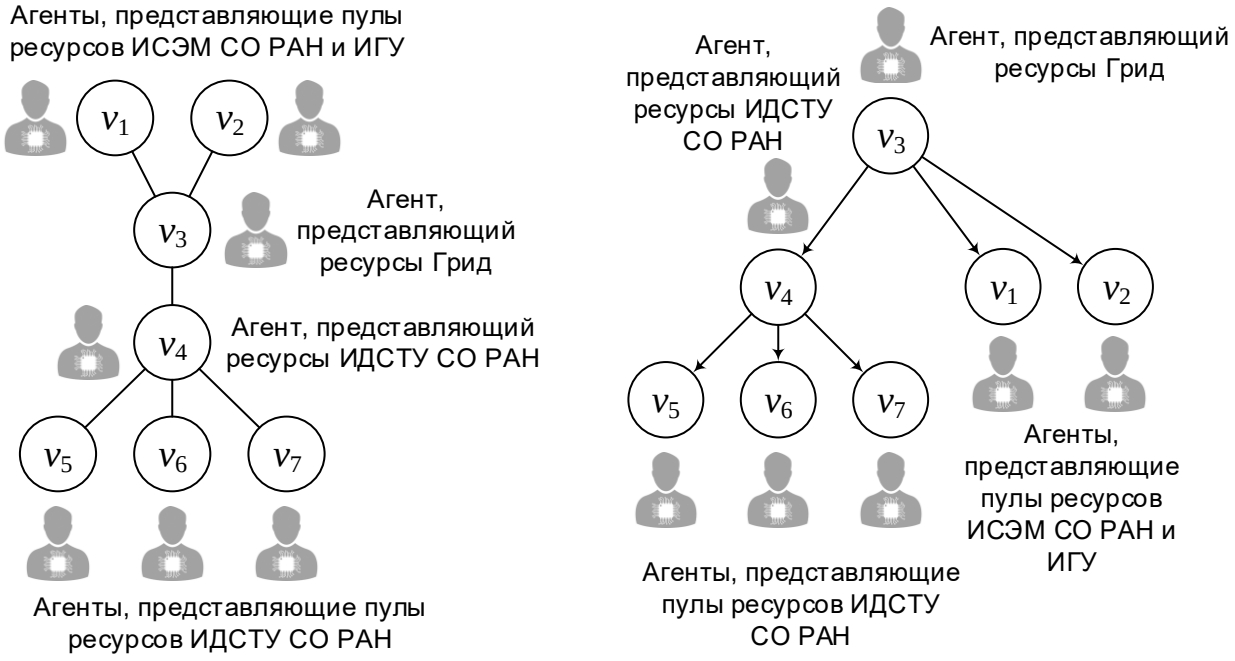
А.5.3. *Формирование топологии коммуникационной сети виртуального сообщества агентов*. Топология строится в виде ориентированного дерева $G =< V, U >$, где V – множество вершин графа, представляющих агентов, U – множество ребер графа, представляющих коммуникационные связи между этими агентами. Дерево G описывается булевой матрицей смежности $\mathbf{A}$ размерности $n_a \times n_a$, элемент которой $a_{ij} = 1$ означает, что существует дуга (v_i, v_j), n_a – число агентов виртуального сообщества. Ориентированность дерева позволяет исключить возможность коллизий при передаче сообщений между агентами. Полустепени захода и исхода вершины v_i (число дуг, входящих в вершину v_i и исходящих из нее) обозначаются соответственно как $d^+(v_i)$ и $d^-(v_i)$. Вершина v_i с полустепенью захода $d^+(v_i) = 0$ является корнем дерева. Вершины v_i с полустепенью исхода $d^-(v_i) = 0$ будем называть листовыми вершинами. Вершины v_i с полустепенью захода $d^+(v_i) = 1$ и полустепенью исхода $d^-(v_i) > 0$ будем называть промежуточными вершинами.

Обозначим через I_i^- (I_i^+) множество индексов вершин графа G, для которых есть исходящие (входящие) дуги, заходящие в вершину v_i (исходящие из вершины v_i). Индексы из множеств I_i^+ и I_i^- совпадают с индексами агентов виртуального сообщества. Построение дерева G выполняется следующим образом.

Пусть связный неориентированный граф G^0 представляет модель коммуникационной среды всех агентов виртуального сообщества, вершины которого взаимно-однозначно соответствуют данным агентам. В графе G^0 выбирается начальная вершина (корень), соответствующая одному из агентов этого сообщества, который характеризуется требуемым показателем качества его функционирования (надежностью, производительностью и т.п.). Ребра графа G^0

ориентируются от корня по направлению к дочерним вершинам. В результате формируется ориентированное дерево G, представляющее топологию коммуникационной сети данного виртуального сообщества агентов.

На рисунках 4.2 а и 4.2 б приведены примеры исходного неориентированного графа G^0 и ориентированного дерева G, построенного на его основе для виртуального сообщества агентов, представляющих ресурсы экспериментальной Грид. Показателем качества функционирования агентов выбрана надежность работы узлов, в которых они размещены. В данном примере она составляет 0.9803, 0.9753, 0.9999, 0.9998, 0.9998, 0.9998 и 0.9997 для узлов агентов, соответствующих вершинам $v_1 - v_7$. Поэтому в качестве корня выбрана вершина v_3.



а) б)

Рисунок 4.2 – Примеры графа G^0 (а) и дерева G (б)

А.5.4. *Проведение тендера вычислительных работ и выбор координатора виртуального сообщества*.

А.5.4.1. *Инициализация тендера вычислительных работ*. Пользовательский агент рассылает агентам сообщение об их включении в виртуальное сообщество. В каждое сообщение упаковываются матрица **S** размерности $k \times n$, представляющая

схему решения задачи s, матрица $\mathbf{P}$ размерности $n \times n$, граф G и фрагмент вычислительной модели $\mathbb{M}' \subseteq \mathbb{M}$, относящийся к описанию объектов схемы s. Действия пользовательского агента представлены процедурой $p1$. Вспомогательные функции, используемые процедурой $p1$, приведены в таблице 15.

Procedure $p1()$

/* действия пользовательского агента с индексом $user\_agent\_number$ */

begin

$mes = pack(\mathbf{S}, \mathbf{P}, G, \mathbb{M}')$;

for $i = \overline{1, n_a}$ **step** 1 **do**

$send(mes, user\_agent\_number, i)$;

enddo;

end.

Таблица 15 – Вспомогательные функции процедуры $p1$

Функция	Описание
$pack(d_1, d_2, \dots)$	Функция упаковки данных $d_1, d_2, \dots$ в сообщение mes.
$send(mes, i, j)$	Функция отправки сообщения mes от i-го агента j-му агенту.

А.5.4.2. *Проведение тендера.* Тендер проводится для каждого e-го уровня схемы s, $e = \overline{1, k}$. Листовые агенты k раз запускают прямую волну. Они формируют свои ставки, упаковывают их в сообщения и пересылают сформированные сообщения агентам, размещенным в родительских вершинах дерева G относительно вершин, в которых размещены листовые агенты. Действия листовых агентов реализуются с помощью процедуры $p2$. В таблице 16 приведено краткое описание вспомогательных функций, используемых в ней.

Procedure $p2(e)$

/* Действия i-го листового агента в рамках прямой волны */

begin

/* Получение сообщения mes и распаковка структуры данных $< \mathbf{S}, \mathbf{P}, G, \mathbb{M}' >$ */

$mes\_user\_agent = resive(user\_agent\_number)$;

$< \mathbf{S}, \mathbf{P}, G, \mathbb{M}' >= unpack\_structure(mes\_user\_agent)$;

/* Формирование ставок на подзадания в отдельности */

for $k = \overline{1, n_{job}}$ **step** 1 **do**

$B_i^k = create\_ind(\mathbf{S}(e,), \mathbf{P}, G, \mathbb{M}', k)$;

enddo;

/* Определение числа $n_{set} \in \overline{0, n_{max\_set}}$ ставок на наборы подзаданий */

$n_{set} = constrain\left(\mathbf{S}(e,), \mathbf{P}, G, \mathbb{M}', B_i^1, B_i^2, \dots, B_i^{n_{job}}\right)$;

/* Формирование ставок на наборы подзаданий */

for $k = \overline{1, n_{set}}$ **step** 1 **do**

$\tilde{B}_i^k = create\_set(\mathbf{S}(e,), \mathbf{P}, G, \mathbb{M}', k)$;

enddo;

/* Упаковка данных в сообщение */

/* k_i^{pr} – коэффициент надежности узла i-го агента */

if $(e = 1)$ **then**

$mes = pack\left(i, B_i^1, B_i^2, \dots, B_i^{n_{job}}, \tilde{B}_i^1, \tilde{B}_i^2, \dots, \tilde{B}_i^{n_{set}}, k_i^{pr}\right)$;

else

$mes = pack\left(i, B_i^1, B_i^2, \dots, B_i^{n_{job}}, \tilde{B}_i^1, \tilde{B}_i^2, \dots, \tilde{B}_i^{n_{set}}\right)$;

endif;

/* Отправка i-м агентом сообщения l_j-му агенту, $l_j \in I_i^+$ */

$send\left(mes, i, l_j\right)$;

end.

Таблица 16 – Вспомогательные функции процедуры $p2$

Функция	Описание
$resive(agent\_number)$	Функция получения сообщения от агента с номером $agent\_number$
$unpack\_structure(mes)$	Функция распаковки структуры данных $< \mathbf{S}, \mathbf{P}, G, \mathbb{M}' >$ из сообщения mes
$create\_ind(par_1, par_2, \dots, par_5)$	Функция формирования ставки на подзадания в отдельности, где par_1 – строка матрицы **S**, par_2 – матрица **P**, par_3 – граф G, par_4 – фрагмент $\mathbb{M}'$ вычислительной модели $\mathbb{M}$, par_5 – индекс ставки
$create\_set(par_1, par_2, \dots, par_5)$	Функция формирования ставки на набор заданий, где параметры $par_1, par_2, \dots, par_5$ имеют такое же назначение, что и для функции $create\_ind()$
$constrain(par_1, par_2, par_3, par_4, par_6, \dots)$	Функция формирования ограничения на допустимое число ставок на наборы подзаданий, где параметры $par_1, par_2, par_3, par_4$ имеют такое же назначение, что и для функции $create\_ind()$. Параметры par_6, ... представляют собой ставки на отдельные подзадания
$pack(), send()$	См. таблицу 15

Агенты, размещенные в промежуточных вершинах дерева G, формируют свои ставки и добавляют их в сообщения с помощью процедуры $p3$. Дополнительные функции, используемые процедурой $p3$, приведены в таблице 17. При $e = 1$ агенты добавляют в сообщения коэффициенты надежности узлов, в

которых они размещены. Затем сообщения рассылаются в соответствии с полустепенью захода вершин, в которых размещены данные агенты.

Procedure $p3(e)$

/* Действия i-го агента, расположенного в промежуточной вершине, в рамках прямой волны */

begin

/* Получение сообщения mes и распаковка структуры данных $< \mathbf{S}, \mathbf{P}, G, \mathbb{M}' >$ */

$mes\_user\_agent = resive(user\_agent\_number)$;

$< \mathbf{S}, \mathbf{P}, G, \mathbb{M}' >= unpack\_structure(mes)$;

/* Получение сообщений от агентов в рамках прямой волны */

for $j = \overline{1, d^{-}(v_i)}: l_j \in I_i^{-}$ **step** 1 **do**

$mes\_tmp = resive(l_j)$;

$mes = add\_mes(mes, mes\_tmp)$;

enddo;

/* Формирование ставок на подзадания в отдельности */

for $k = \overline{1, n_{job}}$ **step** 1 **do**

$B_i^k = create\_ind(\mathbf{S}(e,), \mathbf{P}, G, \mathbb{M}', k)$;

enddo;

/* Определение числа $n_{set} \in \overline{0, n_{max\_set}}$ ставок на наборы подзаданий */

$n_{set} = constrain(\mathbf{S}(e,), \mathbf{P}, G, \mathbb{M}', B_i^1, B_i^2, \dots, B_i^{n_{job}})$;

/* Формирование ставок на наборы подзаданий */

for $k = \overline{1, n_{set}}$ **step** 1 **do**

$\tilde{B}_i^k = create\_set(\mathbf{S}(e,), \mathbf{P}, G, \mathbb{M}', k)$;

enddo;

/* Упаковка данных в сообщение */

/* k_i^{pr} – коэффициент надежности узла i-го агента */

if $(e = 1)$ **then**

$$mes\_tmp = pack\left(i, B_i^1, B_i^2, \dots, B_i^{n_{job}}, \tilde{B}_i^1, \tilde{B}_i^2, \dots, \tilde{B}_i^{n_{set}}, k_i^{pr}\right);$$

else

$$mes\_tmp = pack\left(i, B_i^1, B_i^2, \dots, B_i^{n_{job}}, \tilde{B}_i^1, \tilde{B}_i^2, \dots, \tilde{B}_i^{n_{set}}\right);$$

endif;

$mes = add\_mes(mes, mes\_tmp)$;

/* Отправка i-м агентом сообщения l_j-му агенту, $l_j \in I_i^+$ */

$send\left(mes, i, l_j\right)$;

end.

Таблица 17 – Вспомогательные функции процедуры $p3$

Функция	Описание
$add\_mes(mes_1, mes_2)$	Функция добавления сообщения mes_2 к сообщению mes_1
$resive()$, $unpack\_structure()$, $create\_ind()$, $create\_set()$, $constrain()$	См. таблицу 16
$pack()$, $send()$	См. таблицу 15

Когда прямая волна доходит до агента, размещенного в корне дерева G, он проверяет наличие ставок агентов для всех подзаданий. Если для какого-либо подзадания нет ставок, удовлетворяющих заданным критериям, то происходит возврат на этап 1 с целью уточнения постановки задачи, перепланирования вычислений и построения новой схемы решения задачи. В противном случае проведение тендера продолжается, и агент принимает решение о распределении вычислительной нагрузки на основе заданной администратором ГРВС модели тендера вычислительных работ – одной из моделей, рассмотренных в разделе 4.5.

При $e = 1$ данный агент назначает агента с максимальным коэффициентом надежности его узла координатором виртуального сообщества (при использовании надежности узла, в котором работает агент, в качестве показателя функционирования агента при выборе корневой вершины дерева G агент,

размещенный в корне, совпадает с координатором виртуального сообщества). Агент-координатор осуществляет функции контроля в процессе выполнения задания. В случае отказа какого-либо агента он при необходимости может принять на себя выполнение доли вычислительной работы отказавшего агента, обеспечивая тем самым надежность функционирования виртуального сообщества [361, 369].

Далее в рамках обратной волны агент, размещенный в корневой вершине, рассылает результаты проведения тендера агентам, размещенным в дочерних вершинах относительно корневой вершины. Его действия представлены процедурой $p4$. Дополнительные функции, используемые процедурой $p4$, приведены в таблице 18.

Procedure $p4(e)$

/* Действия i-го агента, расположенного в корне дерева G */

begin

/* Получение сообщения mes и распаковка структуры данных $< \mathbf{S}, \mathbf{P}, G, \mathbb{M}' >$ */

$mes\_user\_agent = resive(user\_agent\_number)$;

$< \mathbf{S}, \mathbf{P}, G, \mathbb{M}' >= unpack\_structure(mes)$;

/* Получение сообщений от агентов в рамках прямой волны */

for $j = \overline{1, d^-(v_i)}: l_j \in I_i^-$ **step** 1 **do**

$mes\_tmp = resive(l_j)$;

$mes = add\_mes(mes, mes\_tmp)$;

enddo;

/* Формирование ставок на подзадания в отдельности */

for $k = \overline{1, n_{job}}$ **step** 1 **do**

$B_i^k = create\_ind(\mathbf{S}(e,), \mathbf{P}, G, \mathbb{M}', k)$;

enddo;

/* Определение числа $n_{set} \in \overline{0, n_{max\_set}}$ ставок на наборы подзаданий */

$n_{set} = constrain(\mathbf{S}(e,), \mathbf{P}, G, \mathbb{M}', B_i^1, B_i^2, \dots, B_i^{n_{job}})$;

/* Формирование ставок на наборы подзаданий */

for $k = \overline{1, n_{set}}$ **step** 1 **do**

$\tilde{B}_i^k = create\_set(\mathbf{S}(e,), \mathbf{P}, G, \mathbb{M}', k)$;

enddo;

/* Упаковка данных в сообщение */

/* k_i^{pr} – коэффициент надежности узла i-го агента */

if $(e = 1)$ **then**

$mes\_tmp = pack\left(i, B_i^1, B_i^2, \dots, B_i^{n_{job}}, \tilde{B}_i^1, \tilde{B}_i^2, \dots, \tilde{B}_i^{n_{set}}, k_i^{pr}\right)$;

else

$mes\_tmp = pack\left(i, B_i^1, B_i^2, \dots, B_i^{n_{job}}, \tilde{B}_i^1, \tilde{B}_i^2, \dots, \tilde{B}_i^{n_{set}}\right)$;

endif;

$mes = add\_mes(mes, mes\_tmp)$;

/* Определение победителей торгов и назначение координатора */

$tender\_results = tender(\mathbf{S}(e,), \mathbf{P}, G, \mathbb{M}', mes)$;

$coordinator\_id = coordinator(mes)$;

/* Формирование и отправка сообщения агентам в рамках обратной волны */

if $(e = 1)$ **then**

$mes = pack(tender\_results, coordinator\_id)$;

else

$mes = pack(tender\_results)$;

endif;

for $j = \overline{1, d^-(v_i)}: l_j \in I_i^-$ **step** 1 **do**

$send\left(mes, i, l_j\right)$;

enddo;

end.

В рамках обратной волны каждый агент, размещенный в промежуточной вершине, извлекает из полученного сообщения результаты его участия в тендере, а само сообщение пересылает агентам в соответствии с полустепенью исхода

соответствующей ему вершины. Действия агента представлены процедурой $p5$. Дополнительные функции, используемые процедурой $p5$, приведены в таблице 19.

Таблица 18 – Вспомогательные функции процедуры $p4$

Функция	Описание
$tender(par_1, par_2, par_3, par_4, mes)$	Функция определяет результаты проведения тендера вычислительных работ, где параметры $par_1, par_2, par_3, par_4$ имеют такое же назначение, что и для функции $create\_ind()$ в таблице 15. Данные о ставках агентов содержатся в сообщении mes
$coordinator(mes)$	Функция определяет индекс координатора виртуального сообщества агентов. Данные о коэффициентах надежности узлов агентов содержатся в сообщении mes
$resive()$, $unpack\_structure()$, $create\_ind()$, $create\_set()$, $constrain()$	См. таблицу 16
$pack()$, $send()$	См. таблицу 15
$add\_mes(mes_1, mes_2)$	См. таблицу 17

Procedure $p5(e)$

/* Действия i-го агента, расположенного в промежуточной вершине, в рамках обратной волны */

begin

/* Получение сообщения mes от агента $l_j \in I_i^+$ */

$mes = resive(l_j)$;

if $(e = 1)$ **then**

$tender\_results = unpack\_tender\_results(mes)$;

$coordinator\_id = unpack\_coordinator\_id(mes)$;

else

$tender\_results = unpack\_tender\_results(mes)$;

endif;

/* Отправка сообщения агентам в рамках обратной волны */

for $j = \overline{1, d^{-}(v_i)}: l_j \in I_i^{-}$ **step** 1 **do**

$send(mes, i, l_j)$;

enddo;

end.

Таблица 19 – Вспомогательные функции процедуры $p5$

Функция	Описание
$unpack\_tender\_results(mes)$	Функция распаковывает результаты тендера вычислительных работ, содержащиеся в сообщении mes
$unpack\_coordinator\_id(mes)$	Функция распаковывает индекс агента координатора виртуального сообществ, содержащийся в сообщении mes
$resive()$, $unpack\_structure()$, $create\_ind()$, $create\_set()$, $constrain()$	См. таблицу 16
$pack()$, $send()$	См. таблицу 15
$add\_mes(mes_1, mes_2)$	См. таблицу 17
$tender()$, $coordinator()$	См. таблицу 18

Каждый листовой агент извлекает из полученного сообщения результаты его участия в тендере с помощью процедуры $p6$. Вспомогательные функции, используемые процедурой $p6$, представлены в таблицах 16 и 19. Если $e < k$, то он

запускает очередную прямую волну для уровня $e = e + 1$ схемы s с помощью процедуры $p2$.

Procedure $p6(e)$

/* Действия i-го агента, расположенного в промежуточной вершине, в рамках обратной волны */

begin

/* Получение сообщения mes от агента $l_j \in I_i^+$ */

$mes = resive(l_j)$;

if $(e = 1)$ **then**

$tender\_results = unpack\_tender\_results(mes)$;

$coordinator\_id = unpack\_coordinator\_id(mes)$;

else

$tender\_results = unpack\_tender\_results(mes)$;

endif;

end.

А.5.5. *Процесс вычислений*. Выполнение модулей схемы s в назначенных агентами ресурсах осуществляется в порядке обслуживания очереди заданий СУПЗ, установленной в узлах агентов, в асинхронном режиме по готовности данных.

Замечание. Выбор одного из волновых алгоритмов для реализации взаимодействия агентов в процессе проведения тендера вычислительных работ обусловлен рядом важных свойств, присущих алгоритмам этого класса [312]: завершение работы за конечное число шагов, обеспечение принятия определенного решения и его причинно-следственная обусловленность всеми участниками процесса. Оценки сложности известных распределенных волновых алгоритмов по числу обменов сообщениями приведены в таблице 20. Параметр k показывает число обменов сообщениями в алгоритме обхода, на котором базируется алгоритм угасания. На основе сравнительного анализа приведенных оценок можно сделать

вывод о преимуществе древесного алгоритма по числу обменов сообщениями в сравнении с другими указанными алгоритмами.

Таблица 20 – Оценки сложности алгоритмов по числу обменов сообщениями

Алгоритм	Топология коммуникационной среды агентов	Оценка сложности алгоритма по числу обменов сообщениями
Древесный алгоритм	Древесная	$2n_a - 2$
Алгоритм Ченя–Робертса	Кольцевая	$O(n_a \log n_a)$
Алгоритм Петерсона / Долева–Клейва–Роде		$1.5n_a \log n_a$
Алгоритм угасания	Заранее неизвестная топология	kn_a
Алгоритм «задиры»		$O(n_a^2)$

Основные этапы алгоритма в той или иной степени детализации представлены в [83, 325, 362, 363].

4.7. Схема мультиагентного управления

Структурная схема мультиагентного управления вычислениями, предложенная в диссертации [84, 367, 390], представлена на рисунке 4.3. На этой схеме в качестве объекта управления выступает ГРВС. Внешними возмущениями для объекта управления являются поток $flow_1$ заданий пользователей среды и поток $flow_2$ локальных пользователей кластеров. Результаты распределения $dstr_1$ и $dstr_2$ этих потоков по кластерам являются соответственно управляющими воздействиями МАС и локальных пользователей на объект управления. Задающим воздействием для объекта управления является вектор параметров административных политик $plcs_1$. Дополнительно администраторы узлов ГРВС определяют намерения $intents$ агентов распределения ресурсов по выполнению заданий разных классов. Агенты распределения ресурсов осуществляют перехват

заданий потока $flow_1$ с целью более детальной настройки требований к вычислительной системе, содержащихся в этих заданиях. Поток $flow_1$ преобразуется в поток $flow_1^*$. Распределение $dstr_2$ потока $flow_2$ задается локальными пользователями.

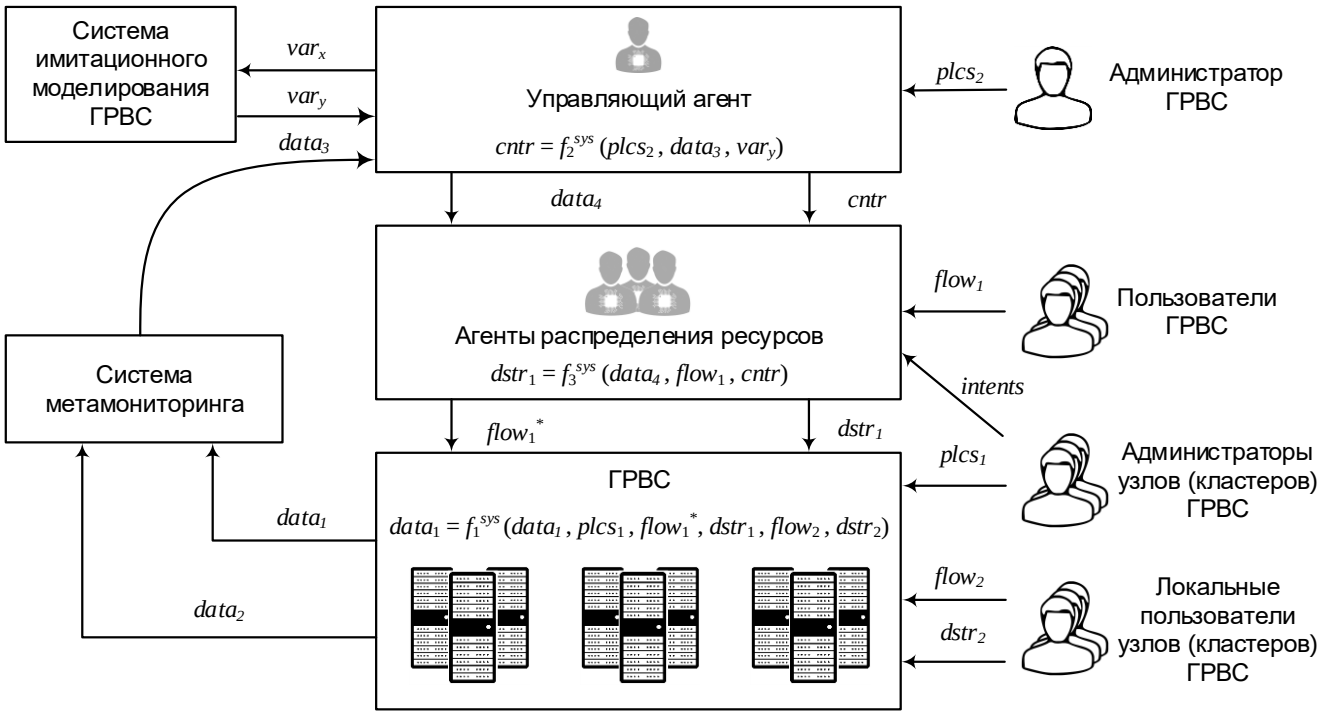


Рисунок 4.3 – Структурная схема мультиагентного управления вычислениями

Сведения о характеристиках узлов ГРВС собираются системой метамониторинга [358] с помощью контрольно-измерительных приборов в виде файловой структуры данных $data_1$. Сведения о текущих показателях объемов работ в узлах среды (выполняющихся и стоящих в очередях) также собираются комплексом метамониторинга в виде файловой структуры данных $data_2$.

Предполагается, что между компонентами структуры $data_1$ с одной стороны и задающим воздействием для объекта управления, потоками заданий, их распределениями и характеристиками узлов с другой стороны существует некоторая абстрактная связь $data_2 = f_1^{sys}(data_1, plcs_1, flow_1^*, dstr_1, flow_2, dstr_2)$. Для различных компонентов структуры $data_2$ эта связь может быть представлена функциональным, статистическим, неоднозначным или иным отображением.

Связи f_2^{sys} и f_3^{sys} имеют ту же природу, что и рассмотренная связь f_1^{sys}.

В общем случае собранные сведения передаются управляющему агенту в виде вектора $data_3$ агрегированных показателей работы объекта управления по его запросу. В частном случае, когда ГРВС разделена на сегменты, в каждом из них может работать свой агент, управляющий агентами распределения ресурсов, которые установлены в узлах данного сегмента.

Запросы управляющего агента к системе мониторинга посылаются с некоторым периодом дискретности T_1. Величина T_1 выбирается таким образом, чтобы не перегружать коммуникационную среду ГРВС сбором информации и в тоже время с необходимой точностью фиксировать моменты приближения показателей функционирования объекта управления к их предельным значениям. Часть сведений, представленных вектором $data_3$ и актуальных для агентов распределения ресурсов, немедленно передается этим агентам в виде вектора $data_4$.

Задающим воздействием для управляющего агента является вектор $plcs_2$ параметров административных политик ГРВС. На основе информации, представленной векторами $data_3$ и $plcs_2$, в заданные моменты времени управляющий агент прогнозирует на определенный промежуток времени динамику показателей качества функционирования объекта управления с помощью системы имитационного моделирования среды [327, 338, 358].

Результаты моделирования используются для формирования вектора управляющих воздействий $cntr$ на алгоритмы работы агентов распределения ресурсов путем их параметрической настройки. Элементами вектора $cntr$ являются следующие параметры алгоритма работы агента распределения ресурсов: ограничения загрузки компонентов узла; бонусы за удовлетворение данным ограничениям; штрафы за их превышение; приоритеты классов; степень желания выполнять задания определенных классов. Процесс имитационного моделирования инициируется управляющим агентом с некоторым периодом дискретности $T_2 > T_1$. Входные и наблюдаемые переменные имитационной модели

ГРВС представлены векторами var_x и var_y. После того, как вектор $cntr$ сформирован, он передается каждому агенту распределения ресурсов, включенному в виртуальное сообщество.

Таким образом, автономные агенты МАС, имеющие свои цели и намерения, кооперируются для обработки общего задания и в то же время конкурируют в процессе распределения его подзаданий. Эмерджентность в работе системы проявляется в том, что в результате проведения тендера они коллективно достигают рационального распределения ресурсов путем локальных взаимодействий между собой, принимая сложное решение, непосильное отдельно взятым агентам. Классификация заданий и управляющие воздействия на алгоритмы работы агентов распределения ресурсов обеспечивают их адаптацию к изменяющимся условиям функционирования ГРВС. Взаимозаменяемость агентов (например, способность агента-координатора выполнять функции отказавшего агента виртуального сообщеста) обеспечивает надежность работы МАС.

4.8. Экспериментальный анализ

Оценка функционирования МАС при обработке потоков заданий (схем решений задач) для трех предложенных моделей тендера проведена в экспериментальной ГРВС, включающей два пула ресурсов (таблица 21).

Таблица 21 – Характеристики ресурсов ГРВС

Пул	Ресурсы	Число ядер
1	8 ВМ со следующими характеристиками: 1 процессор Intel Xeon E5506 (1 ядро, 2.13 Ггц, 4 Мбайт кэш L3, 2 Гбайт RAM DDR3-800, 4 FLOP/cycle).	8
2	10 ВМ со следующими характеристиками: 1 процессор AMD Opteron 6276 (8 ядер, 2.3 Ггц, 16 Мбайт кэш L3, 32 Гбайт RAM DDR3-1600, 4 FLOP/cycle).	80

В роли схем решения задач выступают типовые рабочие процессы Montage

[22], CyberShake [143], Epigenomics [207], LIGO (Laser Interferometer Gravitational Wave Observatory) [5] и SIPHT (sRNA Identification Protocol using High-throughput Technology) [138], объединенные в рамках одного РППП. Предпочтение данных рабочих процессов обусловлено тем, что они наиболее близко отражают характерные особенности схем решения практических задач, которые рассматриваются ниже в главах 5 и 6.

Рабочие процессы в РППП включают от 4 до 13 операций. Графическое представление рабочих процессов, отражающее распределение их операций по уровням, включено в Приложение Е. В таблице 22 для каждого уровня рабочих процессов приведены общее число операций / число параллельных операций / максимальное число комбинаций операций. Максимальное число комбинаций операций дано без учета выполнения экземпляров одной и той же операции.

Таблица 22 – Размещение операций рабочих процессов по уровням

Рабочий процесс	Номер уровня рабочего процесса								
	1	2	3	4	5	6	7	8	9
Montage	1/1/1	1/1/1	1/0/1	1/0/1	1/1/1	1/0/1	1/0/1	1/0/1	1/0/1
CyberShake	1/1/1	1/1/1	1/0/1	1/1/1	1/0/1	–	–	–	–
Epigenomics	1/0/1	1/1/1	1/1/1	1/1/1	1/1/1	1/0/1	1/0/1	1/0/1	–
LIGO	1/1/1	1/1/1	1/0/1	1/1/1	1/1/1	1/0/1	–	–	–
SIPHT	5/1/32	2/0/4	1/0/1	4/0/16	1/0/1	–	–	–	–

Примеры сгенерированных заданий для каждого из вышеперечисленных рабочих процессов в форматах, требуемых метапланировщиками Condor DAGMan и GridWay представлены в Приложении Ж. МАС использует формат описания рабочего процесса, используемый Condor DAGMan.

С целью сбора статистики, необходимой для экспериментального анализа, каждый поток схем решения задач был предварительно выполнен на одном узле пулов 1 и 2, а также в пуле 1. Далее эти потоки выполнены в ГРВС, включающей оба пула, под управлением трех метапланировщиков: Condor DAGMan, GridWay и

МАС, предложенной в рамках диссертационного исследования. В случае применения МАС все потоки выполнены для каждой из трех моделей тендера вычислительных работ. Каждый поток включает 30 рабочих процессов. Характеристики потоков рабочих процессов приведены в Приложении 3. Для сравнительного анализа обработки потоков схем решения задач отобраны следующие критерии: стоимость и время выполнения потока, ускорение вычислений, эффективность использования ресурсов, средняя загрузка процессора и среднеквадратическое отклонение от нее, характеризующее степень балансировки загрузки ресурсов. Первые три критерия представляют требования пользователя. Остальные критерии отражают предпочтения владельцев ресурсов.

МАС показывает преимущество при использовании модели 1 по всем критериям пользователя для каждого потока рабочих процессов в сравнении с Condor DAGMan и GridWay. МАС обеспечивает сокращение времени выполнения потоков рабочих процессов Montage, CyberShake, Epigenomics, LIGO и SIPHT (рисунок 4.4) по сравнению с Condor DAGMan (GridWay). Такое сокращение времени обусловлено следующими факторами:

- применением агентами системы классификации заданий, обеспечивающей распределение заданий, которые относятся к классу заданий с большим (меньшим) временем счета, на более (менее) производительные узлы;
- назначением подзаданий в ходе проведения тендера на ресурсы, на которых выполнялись взаимосвязанные с ними подзадания, что позволяет минимизировать время на передачу данных между ними;
- долевым распределением вычислительной нагрузки агентами при выполнении параллельных операций путем группировки заданий и сокращения накладных расходов на обработку отдельных заданий;
- уменьшением накладных расходов на передачу файлов данных между выполняемыми модулями (агенты МАС передают данные напрямую друг другу, а Condor DAGMan и GridWay – через централизованное хранилище);
- снижением накладных расходов на обработку очереди заданий (агенты сразу распределяют подзадания рабочего процесса по ресурсам, где они выполняются по готовности данных, тогда как Condor DAGMan и GridWay

помещают их в общую очередь, которая обрабатывается с определенной периодичностью и включает, в частности, проверку статуса выполнения всех подзаданий, от которых зависит стоящее в очереди подзадание).

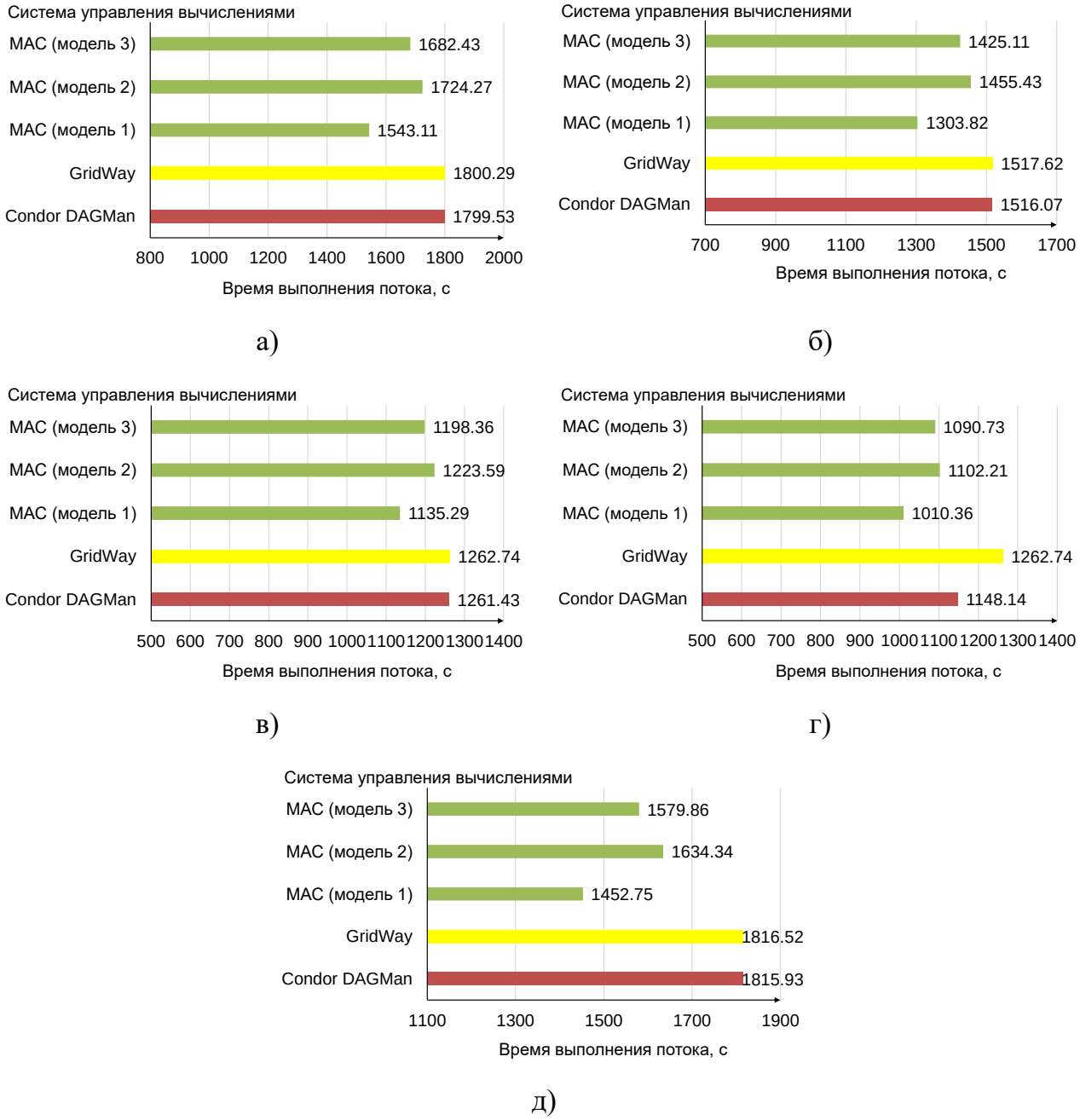
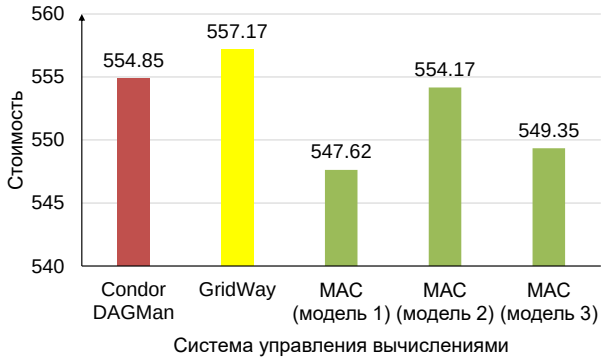


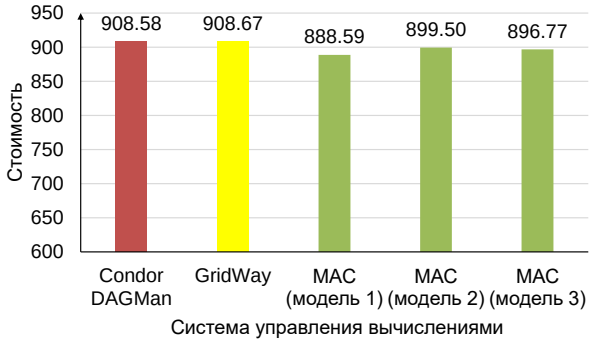
Рисунок 4.4 – Время выполнения потоков рабочих процессов Montage (а), CyberShake (б), Epigenomics (в), LIGO (г) и SIPHT (д)

Как показывает рисунок 4.4, время выполнения потоков под управлением MAC с использованием моделей 2 и 3 также меньше времени выполнения потоков

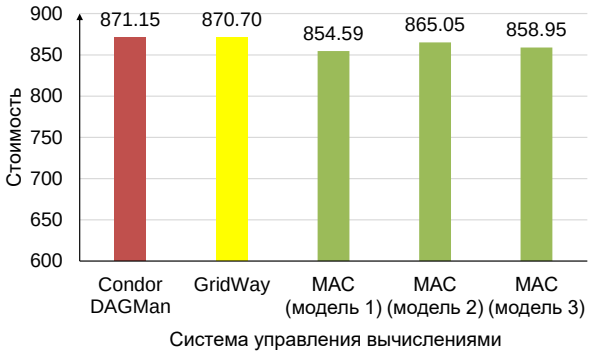
под управлением Condor DAGMan и GridWay.



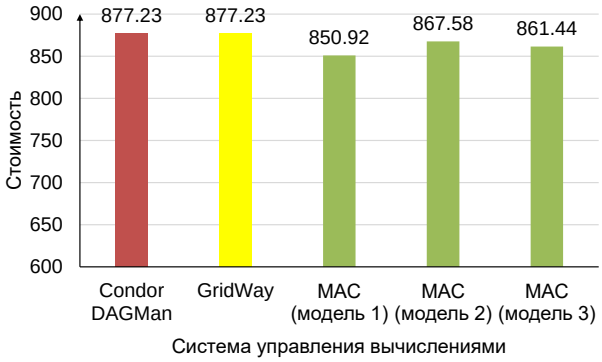
а)



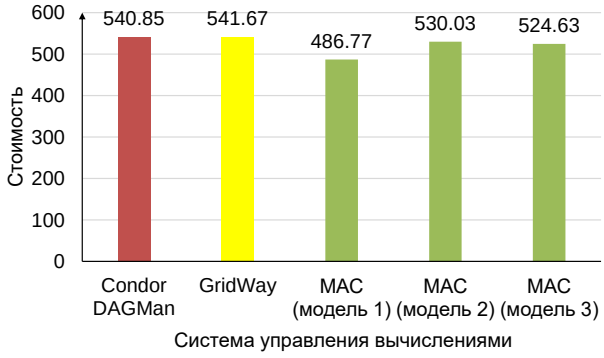
б)



в)



г)



д)

Рисунок 4.5 – Стоимость процессорного времени при выполнении потоков рабочих процессов Montage (а), CyberShake (б), Epigenomics (в), LIGO (г) и SIPHT (д)

Так как предполагается, что стоимость выполнения одной и той же вычислительной работы на более производительном ресурсе не может быть выше стоимости ее выполнения на менее производительном ресурсе (см. раздел 4.4), то

при сокращении времени решения задачи стоимость вычислений не увеличивается. Таким образом, преимущество MAC по времени выполнения потоков заданий обуславливает ее преимущество и по стоимости процессорного времени (рисунок 4.5).

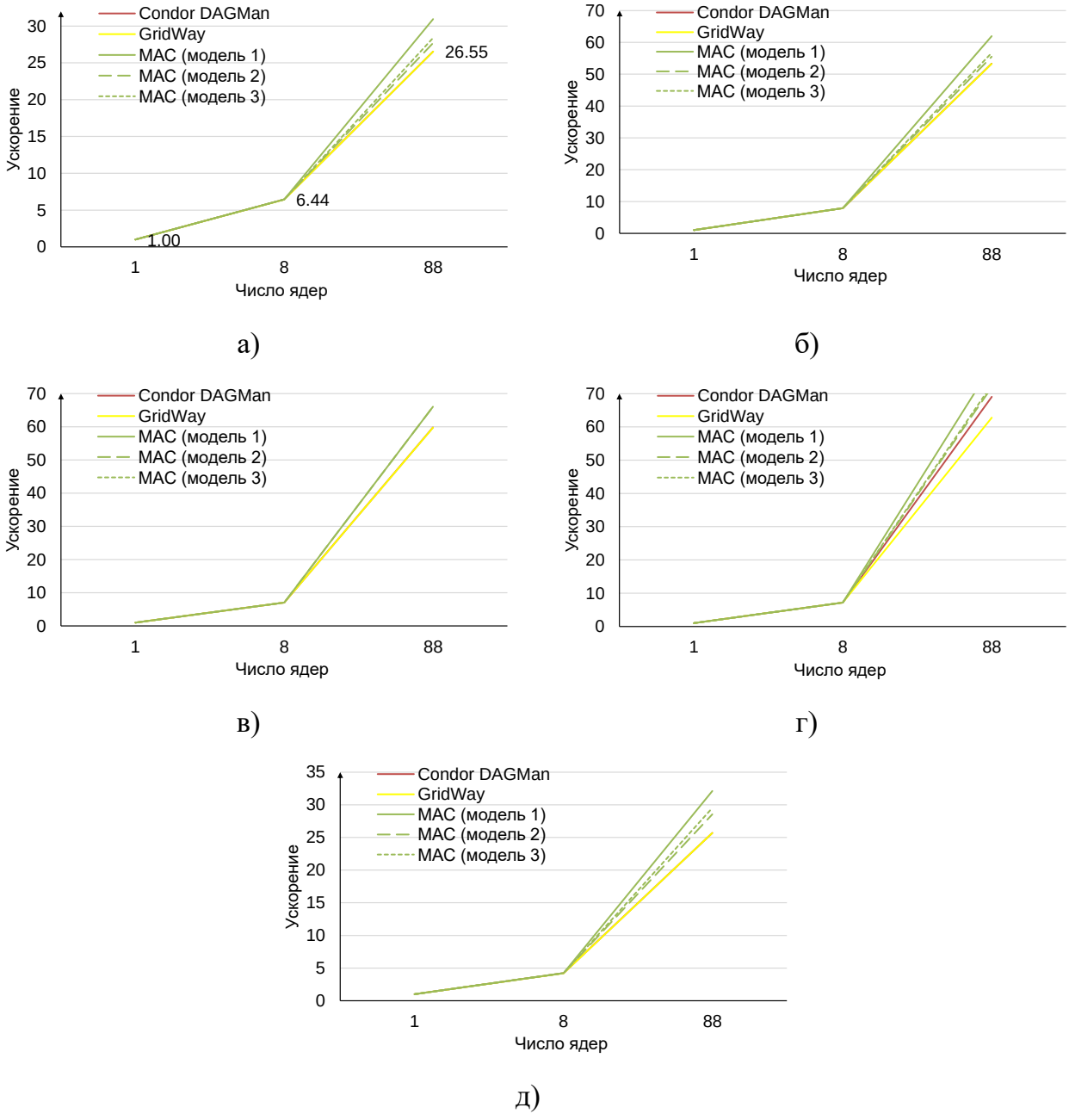


Рисунок 4.6 – Ускорение вычислений при выполнении потоков рабочих процессов Montage (а), CyberShake (б), Epigenomics (в), LIGO (г) и SIPHT (д)

Сокращение времени выполнения потоков напрямую повышает ускорение

вычислений (рисунок 4.6) и эффективность использования ресурсов (рисунок 4.7). При использовании всех трех моделей MAC эти показатели улучшены по сравнению с Condor DAGMan и GridWay. Ускорение вычислений, приведенное на рисунке 4.6, рассчитано относительно выполнения потоков заданий на одном ядре ВМ пула 2.

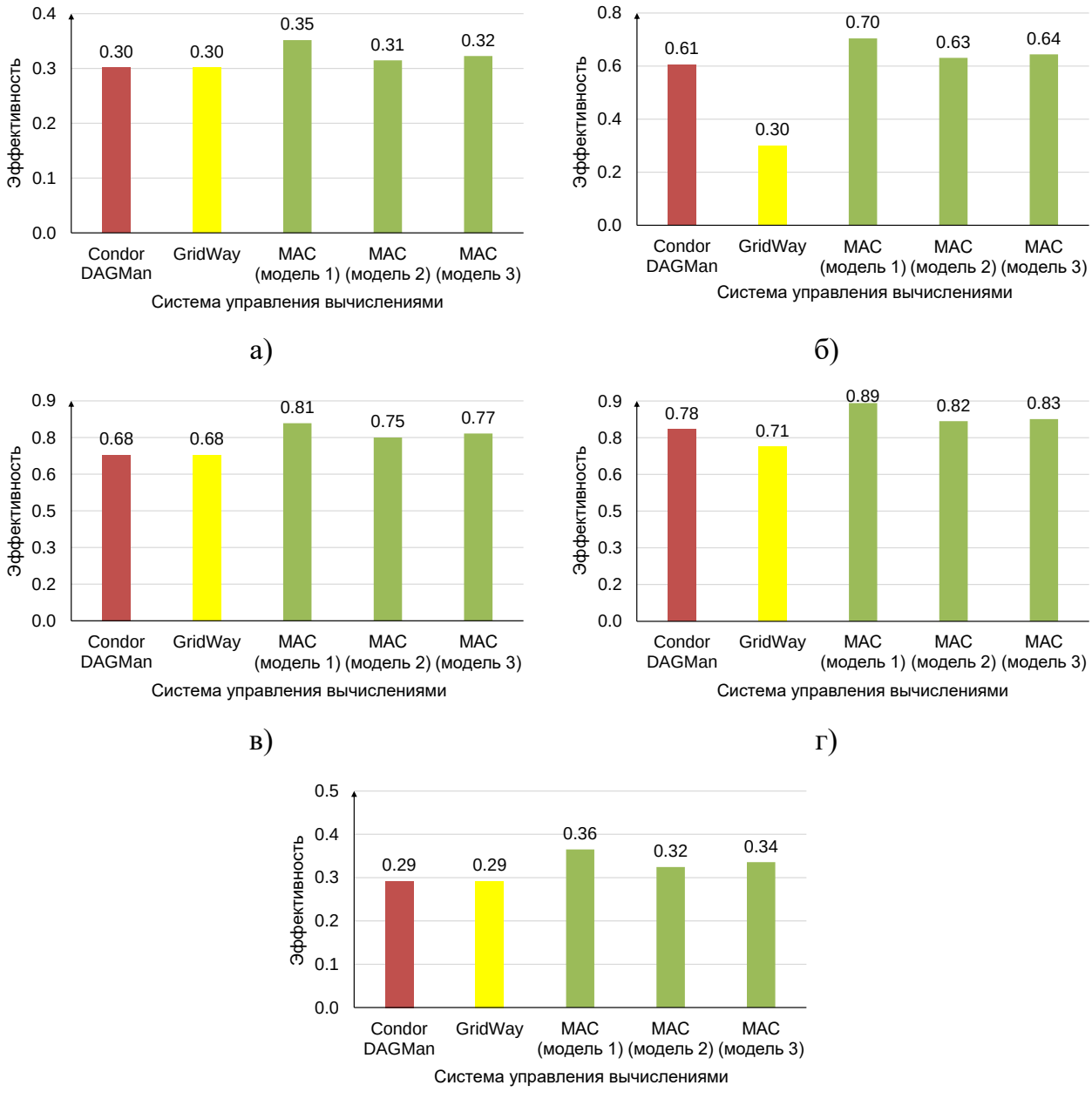


Рисунок 4.7 – Эффективность использования ресурсов при выполнении потоков рабочих процессов Montage (а), CyberShake (б), Epigenomics (в), LIGO (г) и SIPHT (д)

Результаты, приведенные на рисунке 4.8, демонстрируют близкие показатели средней загрузки процессора, полученной при выполнении всех пяти потоков рабочих процессов под управлением CondorDAGMan, GridWay и MAC, использующей модели 1 и 3. Наилучшие значения средней загрузки процессора для всех этих потоков достигнуты MAC на основе модели 2, ориентированной на удовлетворение предпочтений их владельцев.

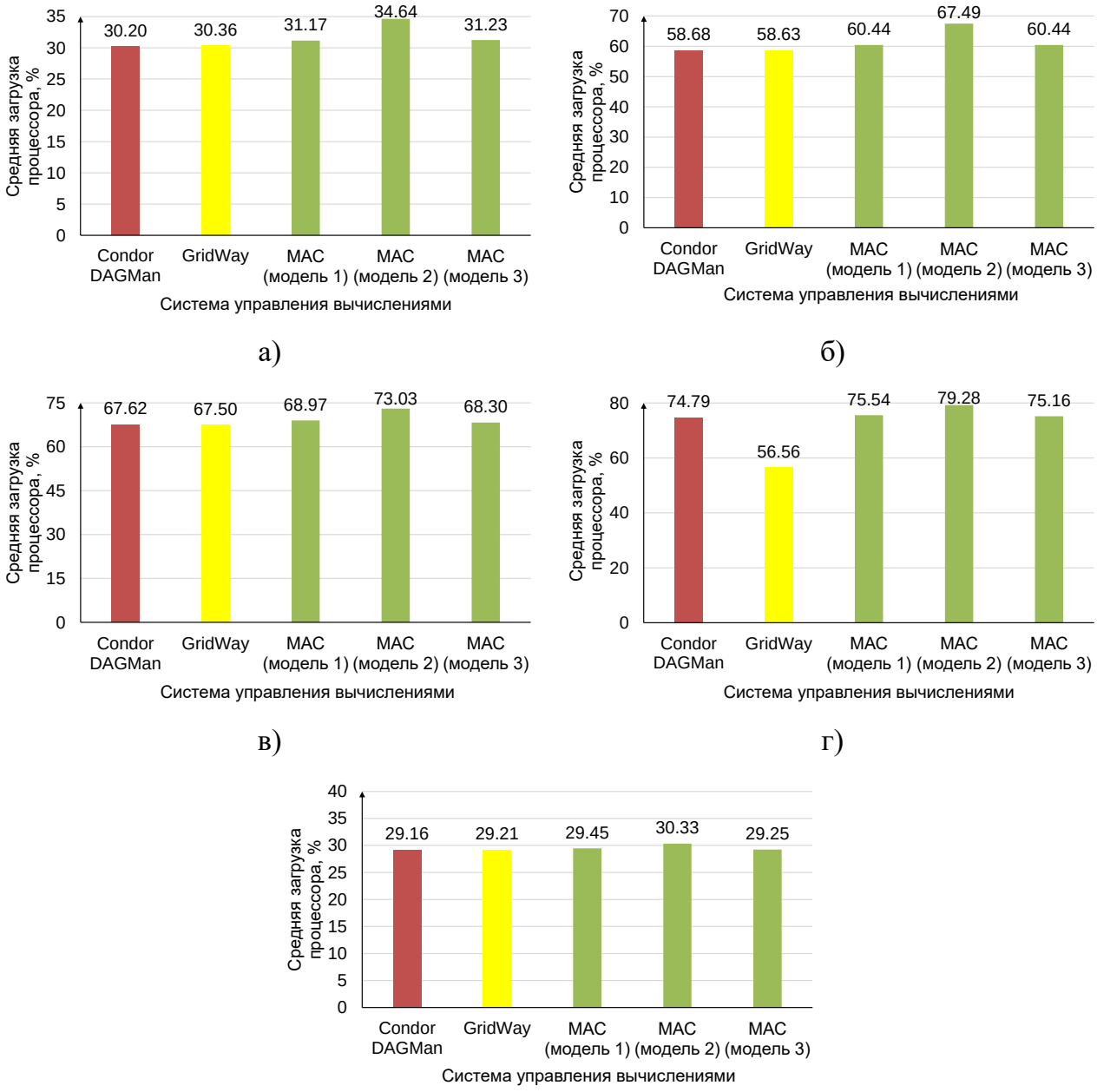


Рисунок 4.8 – Средняя загрузка процессора при выполнении потоков рабочих процессов Montage (а), CyberShake (б), Epigenomics (в), LIGO (г) и SIPHT (д)

Среднеквадратическое отклонение от средней загрузки процессора, полученное MAC при использовании моделей 1 и 3, в большинстве случаев сопоставимо со среднеквадратическим отклонением, соответствующим CondorDAGMan, и, как правило, меньше среднеквадратического отклонения, достигнутого при применении GridWay (рисунок 4.9).

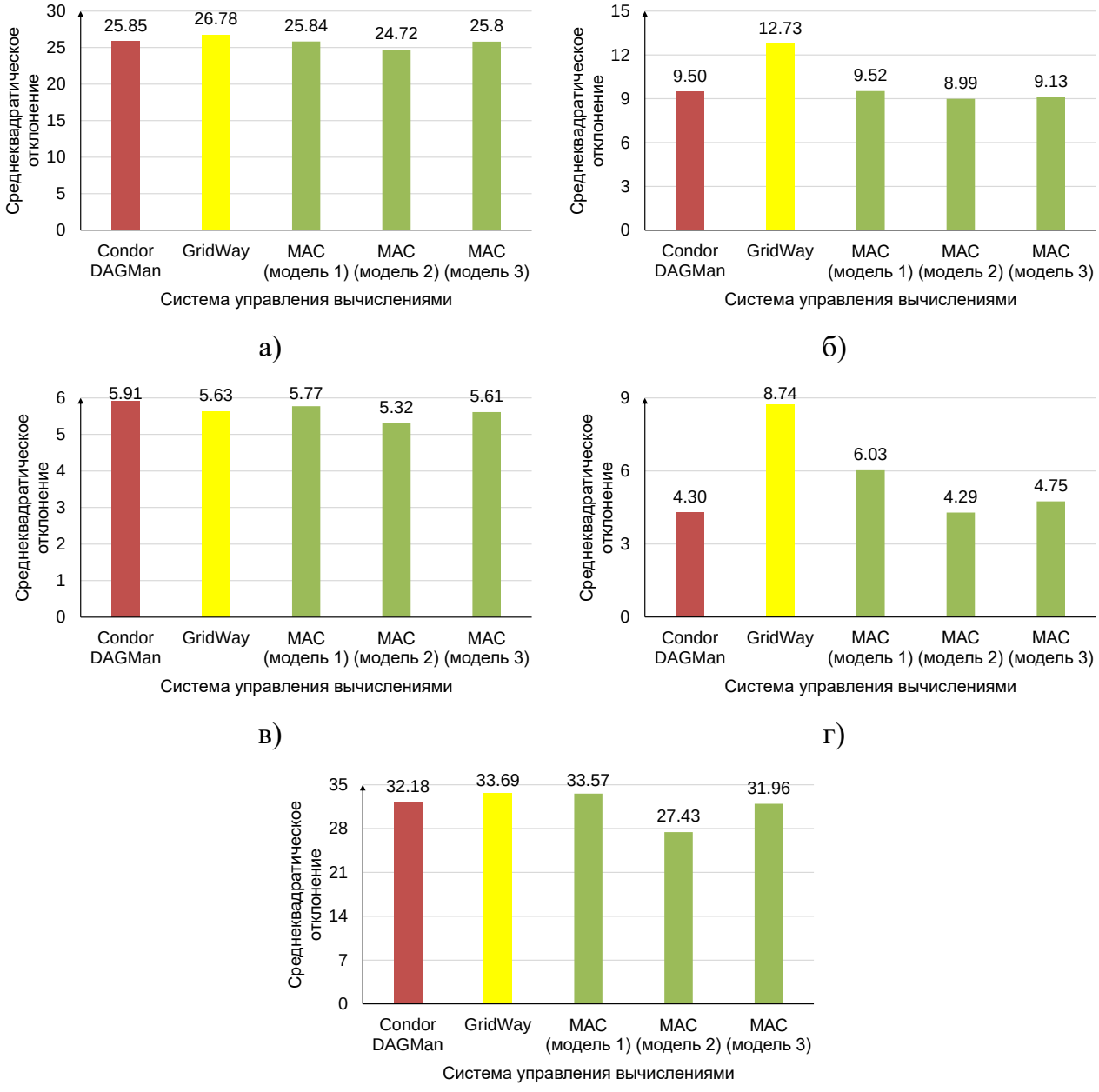


Рисунок 4.9 – Среднеквадратическое отклонение от средней загрузки процессора при выполнении потоков рабочих процессов Montage (а), CyberShake (б), Epigenomics (в), LIGO (г) и SIPHT (д)

Однако среднеквадратическое отклонение, соответствующее модели 2 тендера вычислительных работ, во всех случаях меньше соответствующих показателей как для CondorDAGMan, так и для GridWay. Этот факт подчеркивает хорошую балансировку загрузки ресурсов при использовании модели 2, которая, как было отмечено выше, как раз и ориентирована на удовлетворение предпочтений их владельцев.

Результаты, приведенные на рисунках 4.4-4.9, показывают, что использование модели 3 позволяет улучшить большую часть рассмотренных выше показателей по сравнению с CondorDAGMan и GridWay в случае неупорядоченных критериев.

Таким образом, преимущества предложенного мультиагентного управления по сравнению с традиционными средствами управления распределенными вычислениями в среде с разнородными ресурсами очевидны. Каждая из трех моделей тендера позволяет достичь требуемого уровня обслуживания с учетом способа упорядочения критериев.

4.9. Выводы

Четвертая глава посвящена решению проблем организации мультиагентного управления в ГРВС.

В четвертой главе получены следующие основные результаты:

- проведен сравнительный анализ известных мультиагентных средств, используемых для управления распределенными вычислениями;
- сформулирована постановка задачи мультиагентного управления заданиями распределенными вычислениями в ГРВС;
- разработаны модели и алгоритмы для поддержки такого управления;
- предложен подход к распределению ресурсов агентами на основе проведения тендера вычислительных работ;
- выполнены экспериментальные исследования по оценке разработанных моделей, алгоритмов и методов мультиагентного управления по сравнению с

известными на основе полунатурного моделирования процессов обработки потоков типовых рабочих процессов.

Результаты исследований, представленные в данной главе, опубликованы в [72, 77, 80, 83, 84, 86, 325–327, 333, 334, 338, 354, 358–364, 366, 367, 369, 370, 372, 377, 390].

Глава 5. Технология разработки и применения научных приложений в гетерогенных распределенных вычислительных средах

Анализ тенденций развития технологий высокопроизводительных распределенных вычислений и систем как в России, так и за рубежом [189, 241, 400] позволяет сделать следующие выводы. Процесс решения крупномасштабных задач в ГРВС, которые интегрируют ресурсы вычислительных центров общего доступа, грид-системы и облачные инфраструктуры, создает новые проблемы как для научных приложений, так и систем управления вычислениями. Эти проблемы связаны с существующими различиями в моделях облачных и грид-вычислений [146], необходимостью их совместного использования, а также конфликтами между предпочтениями владельцев ресурсов и критериями качества решения задач пользователей среды [205].

Мультиагентный подход позволяет значительно смягчить вышеупомянутые различия и конфликты посредством взаимодействия агентов, представляющих ресурсы центров и облаков, а также их владельцев и пользователей. В то же время скоординированные действия агентов могут существенно улучшить качество управления распределенными вычислениями, особенно при использовании рыночных механизмов регулирования спроса и предложения ресурсов [186].

Другим направлением для улучшения качества распределенных вычислений является проблемная ориентация систем управления [308]. Его важность обусловлена необходимостью интегрированного использования гетерогенных ресурсов среды в процессе решения общих задач с учетом их специфики, соответствия предпочтений владельцев ресурсов и критериев решения задач, а также автоматизации поддержки принятия решений в системах управления [169, 188]. Таким образом, встает вопрос об интегрированном использовании знаний научных приложений и систем управления вычислениями, в том числе мультиагентных систем.

Качество функционирования агентов напрямую зависит от знаний, которые они используют [196]. В известных средствах мультиагентного управления

вычислениями [141] процессы выявления и применения знаний агентами остаются актуальной проблемой и обоснованно требуют их дальнейшего развития [60].

В пятой главе решается актуальная проблема, связанная с разработкой научных приложений (пакетов прикладных программ) для решения крупномасштабных задач в ГРВС, которые могут включать различные инфраструктуры (кластеры, грид-системы, облака) и обеспечивать их интегрированное использование.

Предлагается подход к разработке приложений для таких сред. Он основан на интеграции концептуального и модульного программирования. Для реализации данного подхода под руководством автора диссертации разработаны три инструментальных комплекса.

Организация грид-вычислений осуществляется с помощью инструментального комплекса DISCENT [345]. Он предназначен для организации кластерных грид-систем.

Разработка приложений реализуется с помощью инструментальных комплексов DISCOMP [373] и Orlando Tools [392], которые используются в качестве базовых сред параллельного программирования в ЦКП ИСКЦ.

По сравнению с известными инструментами, используемыми для разработки и выполнения распределенных приложений, средства данных инструментальных комплексов [325] обеспечивают выполнение заданий приложений в интегрированной среде ВМ, которая включает как выделенные, так и невыделенные ресурсы, с мультиагентным управлением, рассмотренным в главе 4.

Приводятся примеры организации и применения ГРВС с различной вычислительной инфраструктурой.

Эксперименты по решению ряда крупномасштабных практических задач показывают преимущества разработанных приложений при их решении в среде, которая поддерживает гибридную вычислительную модель, включающую облачные и грид-вычисления.

5.1. Инструментальный комплекс DISCENT

В настоящее время грид-технология по-прежнему остается одним из важных направлений создания вычислительных сред для поддержки крупномасштабных научных исследований [175]. Эта технология базируется на интеграции разнообразных географически распределенных информационно-вычислительных и коммуникационных ресурсов и их совместном использовании в процессе вычислений.

Как правило, вычислительная Грид предоставляет стандартный набор услуг и ПО для интегрированного использования информационно-вычислительных ресурсов и обеспечивает организационно-структурную безопасность в процессе управления доступом к удаленным ресурсам в процессе их использования, а также передачи данных. Использование стандартизированных интерфейсов компонентов Грид позволяет пользователям и их приложениям осуществлять выбор программных и аппаратных средств, применяемых в вычислительной среде.

Инфраструктура Грид может включать разнообразные аппаратные средства выполнения распределенных вычислений и обработки больших данных (суперкомпьютеры, кластеры, отдельные персональные компьютеры, системы хранения данных и др.), объединенные телекоммуникационной средой, а также системное ПО, предназначенное для управления процессом выполнения заданий пользователей в рамках такой инфраструктуры. Широко распространенной формой организации распределенных вычислений является кластерная Грид, которая включает кластеры в качестве своих основных узлов.

Инструментальный комплекс DISCENT предназначен для организации и применения Грид такого вида [345, 382]. Его архитектура включает три основных компонента (рисунок 5.1): веб-интерфейс, конструктор среды, базу знаний и систему управления распределенными вычислениями.

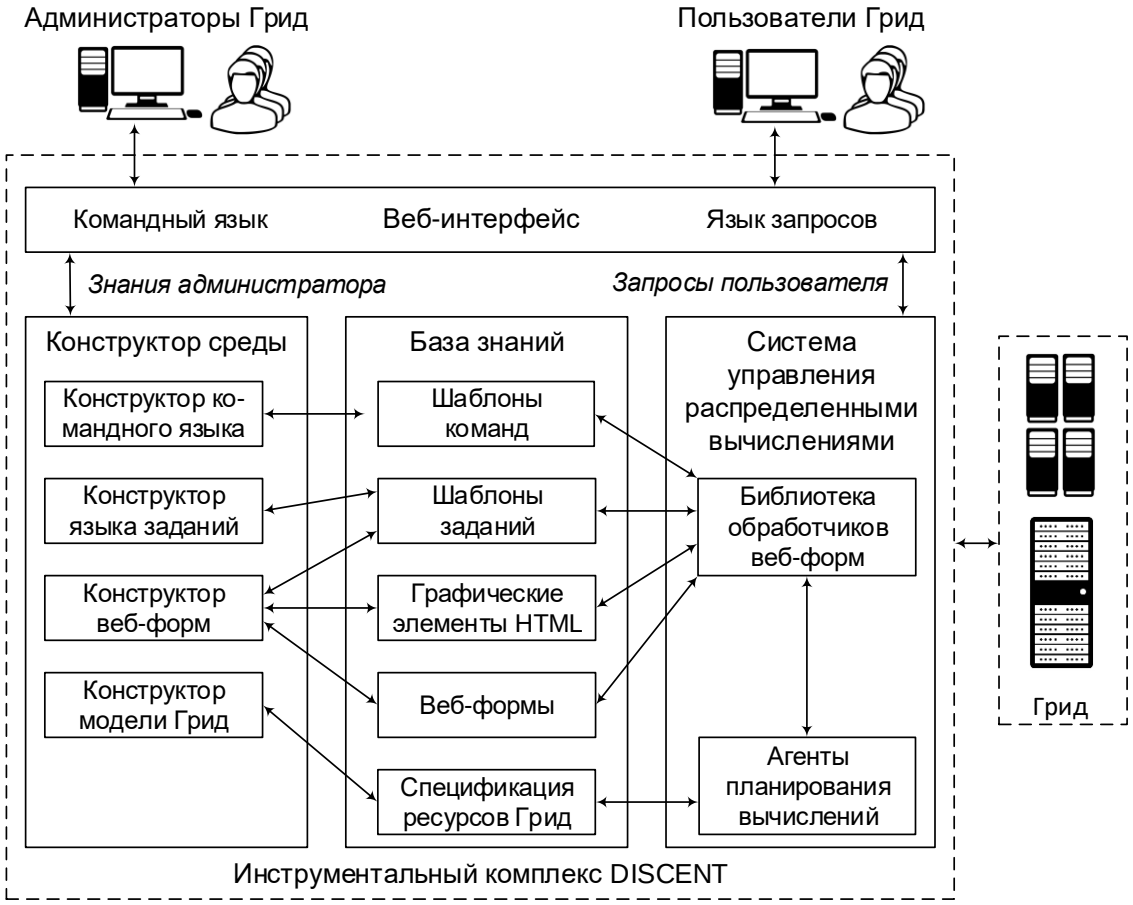


Рисунок 5.1 – Архитектура инструментального комплекса DISCENT

Веб-интерфейс обеспечивает доступ администраторов Грид и ее пользователей к компонентам данного инструментального комплекса, конфигурирование среды и регистрацию пользователей. Он поддерживает два входных языка: командный язык и язык запросов. Командный язык представляет собой совокупность команд, предназначенных для управления пользовательскими заданиями в узлах Грид. Язык заданий предназначен для спецификации заданий. Он обеспечивает взаимодействие пользователей с системой управления распределенными вычислениями.

Конструктор среды обеспечивает ее администратору следующие системные возможности:

– формирование командного языка путем создания шаблонов команд для различных СУПЗ;

- определение языка заданий на основе разработки шаблонов спецификаций заданий для различных СУПЗ;
- построение веб-форм для заполнения спецификаций заданий с использованием в интерактивном режиме набора графических элементов языка HyperText Markup Language (HTML) – полей, списков и т.п., добавляемых в шаблоны веб-форм в соответствии с параметрами данных спецификаций;
- описание и обработку информации о вычислительных ресурсах Грид.

База знаний хранит шаблоны команд, заданий и веб-форм, а также описания графических элементов HTML, которые могут быть использованы при построении этих форм. Она также включает информацию о ресурсах Грид, правилах и квотах их использования. Для описания данной информации применяется АМ ГРВС.

Система управления распределенными вычислениями обеспечивает обработку заданий пользователя и распределение ресурсов Грид для их выполнения. Основными составляющими данной системы являются библиотека обработчиков веб-форм и агенты планирования вычислений.

Библиотека обработчиков веб-форм предназначена для выполнения следующих функций: вывода пользователям требуемых веб-форм; обработки поступивших из этих форм данных; спецификации заданий; предоставления пользователю информации о состоянии заданий и результатах счета; добавления, удаления и приостановления заданий пользователей.

Данная библиотека состоит из набора скриптов (обработчиков веб-форм) на языке Hypertext Preprocessor (PHP), каждый из которых реализует алгоритмы для работы с одной или несколькими веб-формами на стороне веб-сервера. Имя обработчика веб-формы содержится на ней в специальном скрытом поле. Обработчики веб-форм имеют доступ к шаблонам команд, хранящимся в базе данных. В шаблонах команд для каждой операции управления заданием, размещенной на веб-форме, определяется команда СУПЗ, с помощью которой данная операция будет выполнена в Грид.

Подсистема планирования вычислений служит для обработки заданий,

назначения им ресурсов и постановки этих заданий в очереди СУПЗ назначенных ресурсов. Данная подсистема реализована с использованием методов и средств мультиагентного управления, рассмотренных в главе 4.

Приложения и сервисы. В инструментальном комплексе DISCENT грид-сервисы приложений могут быть реализованы с использованием протокола Simple Object Access Protocol (SOAP) [213]. Данный протокол был выбран, исходя из его надежности и безопасности, а также отсутствия каких-либо ограничений по сравнению со стилем Representational State Transfer (REST) [215] или стандартом Web Processing Service (WPS) [88] при организации грид-сервисов и взаимодействия с ними.

Для того чтобы создать грид-сервис на основе приложения, требуется сформировать описание сервиса на языке Web Services Description Language (WSDL) [127]. WSDL-описание грид-сервиса должно содержать унифицированный идентификатор ресурса, в котором размещен сервис, интерфейс взаимодействия клиента с сервисом, типы используемых данных, функции, предоставляемые сервисом, а также другую информацию, необходимую для взаимодействия клиента с этим сервисом. Формат WSDL-описания грид-сервиса включает: корневой элемент <definition>, в котором определяются пространства имен, используемых в данном описании; элемент <types>, включающий типы данных, используемые грид-сервисом; элемент <message>, описывающий сообщения, которые обрабатывает грид-сервис; элемент <portType>, содержащий описание операций, выполняемых грид-сервисом; список элементов <operation>, каждый из которых именует один из методов, выполняемых грид-сервисом, и описывает его входные и выходные сообщения; элементы <input>, <output> и <fault>, описывающие входные и выходные сообщения, а также сообщения об ошибках; элемент <binding>, определяющий для каждого сообщения конкретизацию сетевого протокола, способы кодировки и упаковки послания; элемент <service>, определяющий местонахождение сервиса и включающий один или несколько вложенных элементов <port>, каждый из которых задает адрес операции грид-сервиса.

Таким образом, при реализации сервис-оринтированного подхода к организации распределенных вычислений для приложений пользователей требуется решить две основные задачи: получить описание сервиса для приложения на языке WSDL; конвертировать пользовательский запрос к сервису в вычислительное задание для СУПЗ, установленных в узлах ГРВС.

Будем считать, что приложения, размещенные в узлах ГРВС, представлены в виде исполняемых файлов, реализованы для работы в пакетном режиме, а их конфигурационные параметры настроены для запуска этих приложений в соответствующих узлах. Зачастую в таком виде бывает представлено унаследованное ПО – программные системы или комплексы, не в полной мере соответствующие характеристикам имеющейся распределенной вычислительной среды, но до сих пор используемые ввиду значительных финансовых, организационных, технических и прочих затруднений, связанных с их модернизацией или заменой.

Для каждого приложения генерируется программная оболочка, реализующая шаблоны вызовов типовых команд СУПЗ, применяемых в вычислительной среде. Как правило, шаблон команды включает имя этой команды, а также списки ее опций и параметров, необходимых для работы с приложением. Схема конвертирования пользовательских запросов в вычислительные задания представлена на рисунке 5.2.

Администратор среды с помощью веб-интерфейса модуля оформления приложений в виде сервисов добавляет описания приложений, размещенных в узлах вычислительной среды, в единый WSDL-файл. Этот файл описывает все операции, предоставляемые сервис-ориентированной средой. Для каждого приложения в форме добавления приложения заполняются следующие данные: имя и описания операции, входные/выходные параметры (имена, описание, типы данных), команда для запуска приложения при выполнении конкретной операции. Для каждого приложения может быть определено несколько операций. Таким образом, составляется шаблон, с помощью которого запрос пользователя будет преобразован в команду вызова удаленного приложения в конкретном узле среды.

Шаблоны сохраняются в базе данных. При обращении к грид-сервису запрос на выполнение операции конвертируются в SOAP-сообщение и поступает на SOAP-сервер, который вызывает программную оболочку. В процессе своего выполнения программная оболочка в свою очередь извлекает нужный шаблон вызова команды СУПЗ из базы данных, конвертирует поступивший запрос и данные в задание для СУПЗ. Далее планировщик СУПЗ отправляет это задание на выполнение в среду.

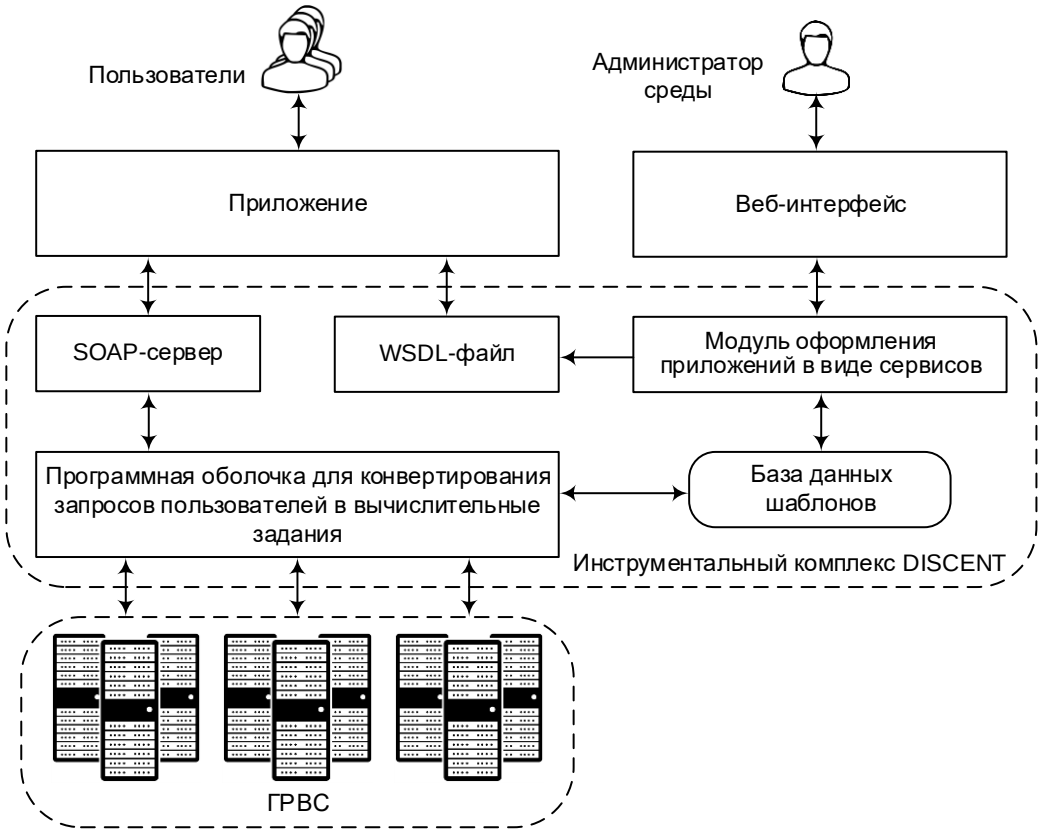


Рисунок 5.2 – Схема конвертирования пользовательских запросов к сервис-ориентированной среде в вычислительные задания

Рассмотрим пример программной оболочки для работы с приложением peispack.exe, реализующим поливариантную схему вычисления собственных значений произвольной плотной матрицы с использованием алгоритмов, представленных в работе [321]. Основные операции сервиса для этого приложения приведены на рисунке 5.3.

```
<portType name='MyServicePortType'>
    <operation name='RunPeispackExe'>
        <documentation>Usage peispack.exe [input.file].</documentation>
        <input message='tns:RunPeispackExeRequest'/>
        <output message='tns:RunPeispackExeResponse'/>
    </operation>
    <operation name='GetTasksList'>
        <documentation>Get task list</documentation>
        <input message='tns:GetTasksListRequest'/>
        <output message='tns:GetTasksListResponse'/>
    </operation>
    <operation name='GetTaskStatus'>
        <documentation>Get task status. Use Task ID</documentation>
        <input message='tns:GetTaskStatusRequest'/>
        <output message='tns:GetTaskStatusResponse'/>
    </operation>
    <operation name='GetTaskResult'>
        <documentation>Get task result. Use Task ID</documentation>
        <input message='tns:GetTaskResultRequest'/>
        <output message='tns:GetTaskResultResponse'/>
    </operation>
</portType>
```

Рисунок 5.3 – Фрагмент WSDL-файла, описывающий операции грид-сервиса

Приложение получает на вход файл с входными данными, результаты счета также выводятся в файл. Операция *RunPeispackExe* предназначена для запуска приложения peispack.exe. Данная операция в формате WSDL представлена двумя секциями (рисунок 5.3): секция *RunPeispackExeRequest* описывает параметры *file_input*, определяющий файл с входными данными, и *meta_scheduler*, задающий имя планировщика заданий (в текущей реализации в качестве планировщика заданий могут использоваться соответствующие утилиты систем Condor и PBS, а также планировщик заданий инструментального комплекса DISCENT); секция *RunPeispackExeResponse* описывает параметр *output*, определяющий файл с результатами счета.

Операция *RunPeispackExe* извлекает шаблон для запуска приложения peispack.exe (рисунок 5.4) из базы данных и конвертирует указанные пользователем данные в задание, которое передается планировщику заданий системы Condor. Данный планировщик выполняет планирование загрузки узлов распределенной вычислительной среды и осуществляет размещение и запуск полученного

приложения в одном из них. Программная оболочка включает ряд дополнительных операций для работы с приложением: получения списка запущенных пользователем заданий (*GetTasksList*), их состояний выполнения (*GetTaskStatus*) и результов счета (*GetTaskResult*).

```
<message name='RunPeispackExeRequest'>
    <part name='file_input'                                    type='xsd:file'/>
    <part name='meta_scheduler'                                type='xsd:string'/>
</message>
<message name='RunPeispackExeResponse'>
    <part name='output' type='xsd:file'/>
</message>

<message name='GetTasksListResponse'>
    <part name='output_list' type='tns:stringArray'/>
</message>

<message name='GetTaskStatusRequest'>
    <part name='task_id_get_task_status' type='xsd:string'/>
</message>
<message name='GetTaskStatusResponse'>
    <part name='output_get_task_status' type='xsd:string'/>
</message>

<message name='GetTaskResultRequest'>
    <part name='task_id_get_task_result' type='xsd:string'/>
</message>
<message name='GetTaskResultResponse'>
    <part name='output_get_task_result' type='xsd:stringArray'/>
</message>
```

Рисунок 5.4 – Фрагмент WSDL-файла, описывающий параметры операций грид-сервиса

Представленный на рисунке 5.5 шаблон для запуска приложения использует следующие переменные: $*ServiceName* – имя сервиса; $*ServiceOperationName* – имя операции, $*InParam1*, …, $*InParamN* – входные параметры, $*InParamType1*, …, $*InParamTypeN* – типы входных параметров, $*OutInParam1*, …, $*OutParamN* – выходные параметры, $*OutInParamType1*, …, $*OutParamTypeN* – типы выходных параметров, $*Service_description* – описание операции, $*Domain* – домен, в котором размещен сервис.

```
<?xml version ='1.0' encoding ='UTF-8' ?>
<definitions name='{$ServiceName}'
 targetNamespace='http://example.org/{$ServiceName}'
 xmlns:tns=' http://example.org/{$ServiceName} '
 xmlns:soap='http://schemas.xmlsoap.org/wsdl/soap/'
 xmlns:xsd='http://www.w3.org/2001/XMLSchema'
 xmlns:soapenc='http://schemas.xmlsoap.org/soap/encoding/'
 xmlns:wsdl='http://schemas.xmlsoap.org/wsdl/'
 xmlns='http://schemas.xmlsoap.org/wsdl/'>

<message name='{$ServiceOperationName}Request'>
    <part name='{$InParam1}' type='{$InParamType1}'/>
    <part name='{$InParam2}' type='{$InParamType2}'/>
    <part name='{$InParamN}' type='{$InParamTypeN}'/>
</message>
<message name='{$ServiceOperationName}Response'>
    <part name='{$OutParam1}' type='{$OutParamType1}'/>
    <part name='{$OutParam2}' type='{$OutParamType2}'/>
    <part name='{$OutParamN}' type='{$OutParamTypeN}'/>
</message>

<portType name='{$ServiceName}PortType'>
    <operation name='{$ServiceOperationName}'>
        <documentation>{$Service_description}</documentation>
        <input message='tns:{$ServiceOperationName}Request'/>
        <output message='tns:{$ServiceOperationName}Response'/>
    </operation>
</portType>
<binding name='{$ServiceName}Binding' type='tns:{$ServiceName}PortType'>
    <soap:binding style='rpc'
    transport='http://schemas.xmlsoap.org/soap/http'/>
    <operation name='{$ServiceOperationName}'>
        <soap:operation soapAction='urn:xmethods-delayed-quotes#{$ServiceOperationName}'/>
        <input>
            <soap:body use='encoded' namespace='urn:xmethods-delayed-quotes'
            encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
        </input>
        <output>
            <soap:body use='encoded' namespace='urn:xmethods-delayed-quotes'
            encodingStyle='http://schemas.xmlsoap.org/soap/encoding/'/>
        </output>
    </operation>
</binding>

<service name='{$ServiceName}'>
    <port name='{$ServiceName}Port' binding='{$ServiceName}Binding'>
        <soap:address location='http://{$Domain}/{$ServiceName}/{$ServiceName}.php'/>
    </port>
</service>
</definitions>
```

Рисунок 5.5 – Шаблон для запуска приложения

Описание грид-сервисов может быть использовано в дальнейшем в инструментариях, обеспечивающих организацию веб-ориентированного доступа к грид-сервисам и поддерживающих формат WSDL. В случае необходимости использования услуг грид-сервисов в удаленных пользовательских приложениях

пользователь может скачать WSDL-файл с описанием грид-сервисов и реализовать функции анализа этого файла и вызова нужных операций грид-сервисов в своем приложении.

Экспериментальная Грид [334, 345, 365, 382]. Инструментальный комплекс DISCENT использован для организации и применения экспериментальной Грид ИДСТУ СО РАН с мультиагентным управлением заданиями. Общие принципы функционирования Грид отражены на рисунке 5.6. В зависимости от требований решаемых задач конфигурации кластеров ПК (кластеров, организованных на базе персональных компьютеров) и ресурсов ЦКП ИСКЦ, подключаемых к Грид, изменялись в процессе ее эксплуатации.

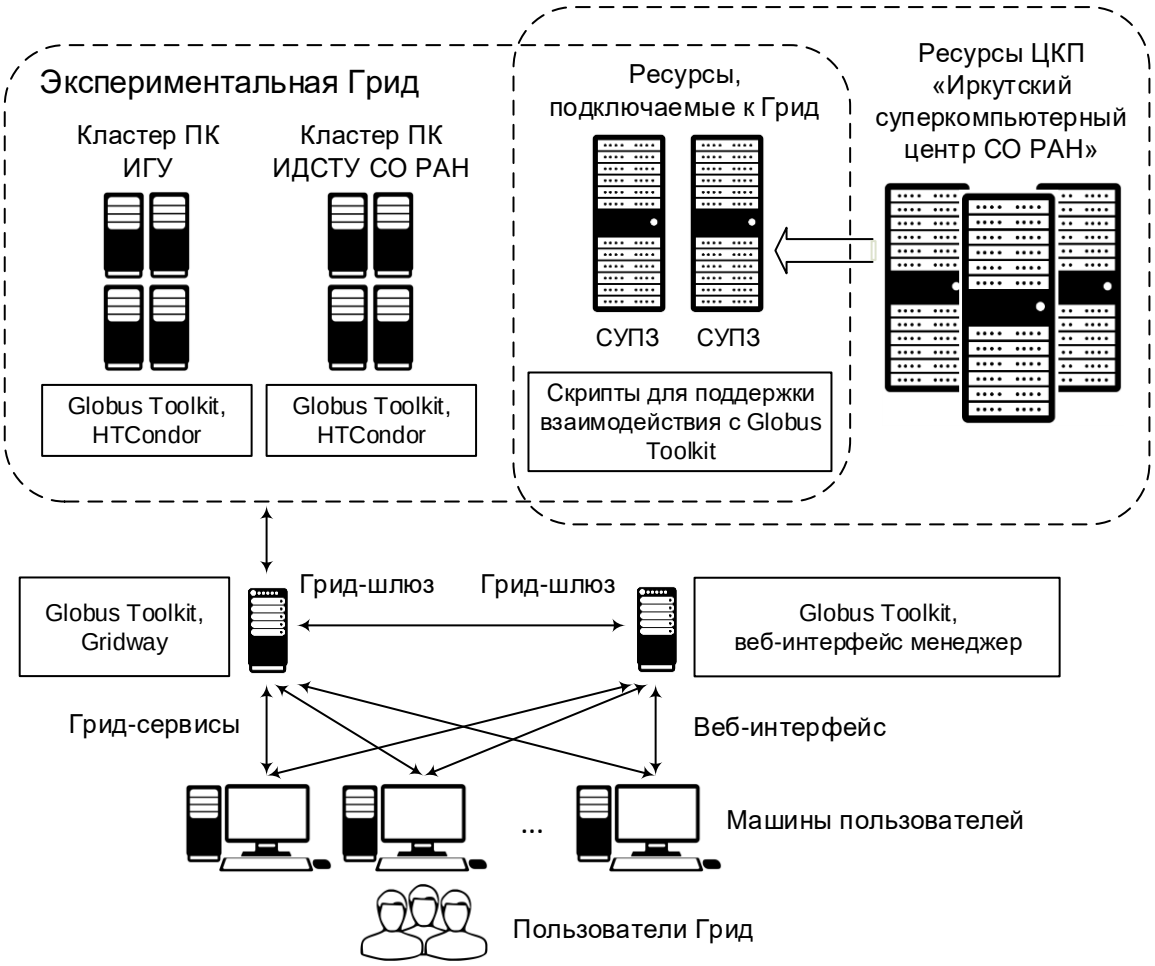


Рисунок 5.6 – Экспериментальная Грид

Кластер ПК объединяет невыделенные машины (персональные компьютеры

и серверы) организации в единую вычислительную систему с целью их совместного использования для решения общих задач (рисунок 5.7). Его узлы, как правило, используются как для решения общих задач в рамках кластера, так и для выполнения заданий своих владельцев. Коммуникационная среда такого кластера представлена имеющейся локальной сетью.

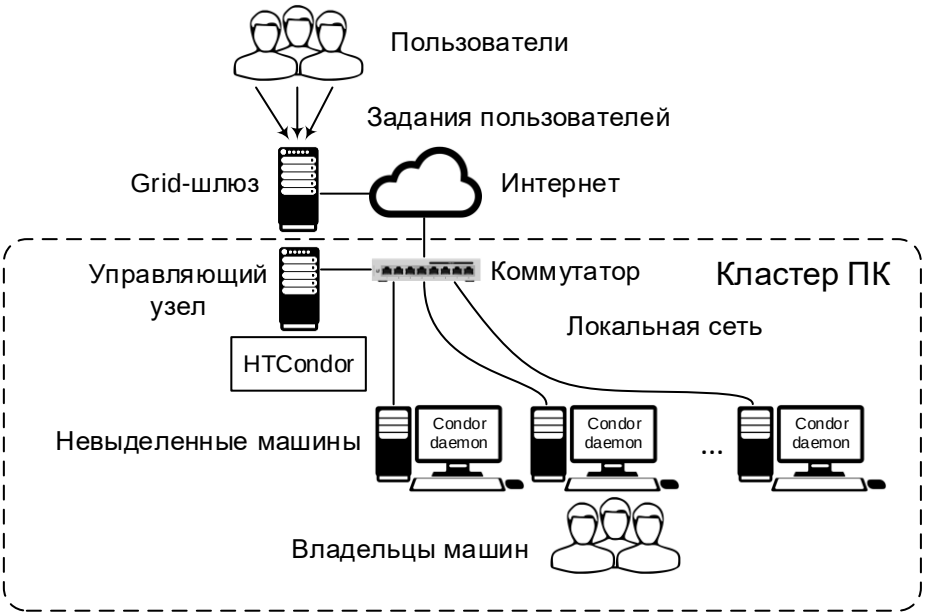


Рисунок 5.7 – Кластер ПК

Кластеры ПК хорошо подходят для выполнения многовариантных расчетов. Выполнение большого числа независимых заданий обеспечивает хорошую загрузку таких дополнительных ресурсов и сокращение общего времени решения задач.

В то же время качество выполнения параллельной программы может быть существенно снижено из-за накладных расходов на взаимодействие ее процессов, запущенных в разных узлах кластера, а также дополнительной вычислительной нагрузки, производимой заданиями владельцев узлов. Низкая надежность вычислительных процессов не позволяет в полной мере обеспечить гарантированное выполнение взаимосвязанных заданий. Таким образом, применение кластеров ПК в составе Грид требует детального учета характеристик узлов.

Кластер ПК Иркутского государственного университета [387] создан и внедрен в учебный процесс Международного института экономики и лингвистики Иркутского государственного университета под руководством и при непосредственном участии автора диссертации с использованием следующих результатов интеллектуальной деятельности [323, 330, 331, 344, 376, 385, 386]. Автор диссертационной работы является соавтором всех вышеперечисленных свидетельств об официальной регистрации программ.

Создание данного кластера является примером успешного внедрения результатов научных исследований в области параллельных и распределенных вычислений в учебный процесс вузов Сибирского региона и их использования при решении ресурсоемких задач исследования систем массового обслуживания [322].

Пакет Globus Toolkit выбран в качестве базового связующего ПО для построения и поддержки функционирования Грид. Управление на уровне Грид реализовано с помощью метапланировщика GridWay. На кластерах установлены различные СУПЗ.

В инструментальном комплексе DISCENT разработан набор системных модулей, реализованных в виде скриптов на языке PHP, для поддержки взаимодействия различных СУПЗ с пакетом Globus Toolkit.

МАС инструментального комплекса DISCENT использована в качестве интеллектной надстройки к метапланировщику GridWay. Она обеспечивает мультиагентное управление потоками заданий разных типов.

Пользователи Грид передают задания со своих машин на Грид-шлюз (вычислительный узел, служащий точкой входа в среду) с помощью Грид-сервисов непосредственно метапланировщику GridWay или используют МАС для предварительного назначения ресурсов с целью сужения области возможных вариантов распределения заданий на основе мультиагентного управления и дальнейшей конкретизации этих заданий. Конкретизированные задания передаются МАС метапланировщику GridWay.

Экспериментальный анализ мультиагентного управления в Грид [365, 370]. С целью анализа использования МАС для управления заданиями в

экспериментальной Грид совместно с метапланировщиком GridWay и различными СУПЗ проведено полунатурное моделирование процессов формирования и обработки потоков заданий разных типов в данной среде.

В рамках полунатурного моделирования генерируются потоки заданий разных типов, осуществляется их распределение и выполнение в реальной Грид. Сгенерированные потоки заданий характеризуются следующими свойствами: неоднородностью (задания соответствуют разным типам задач и отличаются друг от друга по своей специфике); отсутствием обратной связи (число заданий, поступивших за один промежуток времени, не зависит от числа заданий, поступивших за другой промежуток времени); неординарностью (возможно поступление двух и более заданий в один и тот же момент времени); стационарностью (число событий, поступивших за определенный промежуток времени, зависит от длины этого промежутка и не зависит от момента его начала).

Для формирования потока заданий и отправки их на выполнение в Грид используется специальный генератор. В качестве приложений в заданиях используются программы-имитаторы, выполняющие в Грид реальную загрузку вычислительных ресурсов и обмен заданными объемами данных.

Управление сгенерированными потоками заданий в процессе полунатурного моделирования осуществляется двумя способами: с помощью метапланировщика GridWay и его совместного использования с МАС (GridWay+MAC). Во втором случае МАС выступает в качестве интеллектной надстройки к метапланировщику GridWay, которая осуществляет предварительное назначение ресурсов и сужает область возможных вариантов распределения заданий.

Для сравнения этих двух способов управления выбраны следующие показатели качества выполнения потоков заданий в Грид: число заданий с нулевым временем c_0 ожидания, среднее время t_q ожидания задания в очереди, среднее число c_q заданий в очереди, среднее время t_g пребывания задания в Грид, средний коэффициент k_g полезного использования ресурсов Грид, число c_f заданий в потоке, общее время t_f решения заданий потока.

Проведены следующие основные эксперименты:

1) выполнение потока заданий, включающего задания с параллельными программами, многовариантные задания, взаимосвязанные задания, задания по выполнению нетиражируемого ПО и их комбинации;
2) выполнение потока заданий, включающего задания с параллельными программами, многовариантные задания, взаимосвязанные задания;
3) выполнение потока заданий, включающего задания с параллельными программами;
4) выполнение потока заданий, включающего многовариантные задания.

Результаты вычислительных экспериментов приведены в таблице 23. Они показывают устойчивое преимущество второго способа распределения заданий (GridWay+MAC) по всем выбранным показателям функционирования ресурсов Грид.

Таблица 23 – Результаты вычислительных экспериментов в Грид

№ эксперимента	Грид	c_f	Система управления	c_0	t_q	c_q	t_g	k_g	t_f
1	2 кластера (168 ядер)	18688	GridWay	49060	87260	7360.31	123920	93.34	4917080
			GridWay+ MAC	52620	80100	7261.05	118440	94.13	4716820
2	3 кластера (184 ядер)	9344	GridWay	24340	44020	3760.76	62580	92.41	2486420
			GridWay+ MAC	26680	39740	3619.87	58460	95.12	2336520
3	2 кластера (168 ядер)	311	GridWay	35556	56480	4783.95	77662	94.74	3195068
			GridWay+ MAC	40150	50371	4619.28	71035	95.17	2800819
4	3 кластера (184 ядер)	157	GridWay	17980	28240	2380.47	39220	93.65	1585160
			GridWay+ MAC	19920	25540	2283.10	35420	96.97	1459720

Улучшение показателей выполнения потока заданий при использовании MAC по сравнению с метапланировщиком GridWay для четырех экспериментов приведено на рисунке 5.8.

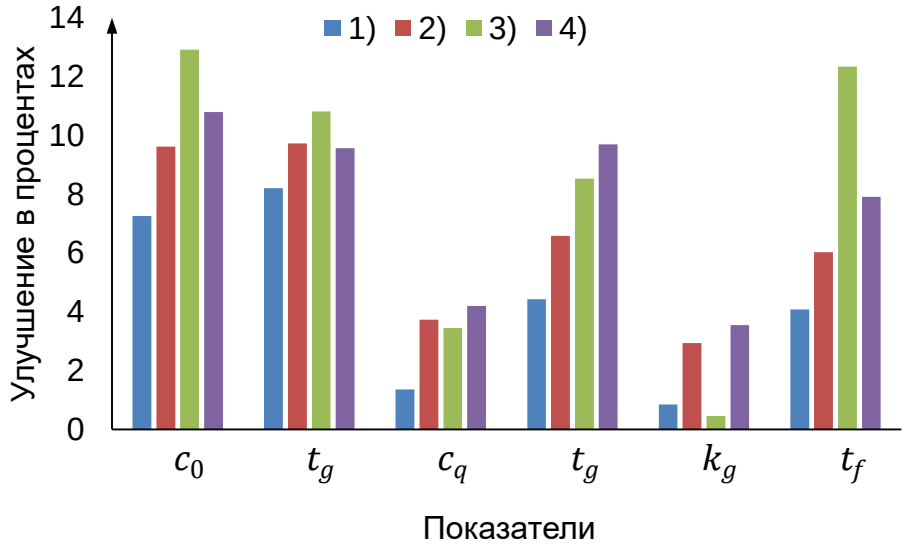


Рисунок 5.8 – Улучшение показателей выполнения потока заданий при использовании MAC

На рисунке 5.9 показано среднеквадратическое отклонение σ от среднего коэффициента k_g полезного использования ресурсов Грид. Данные результаты показывают, что применение MAC позволяет улучшить существенно балансировку загрузки ресурсов.

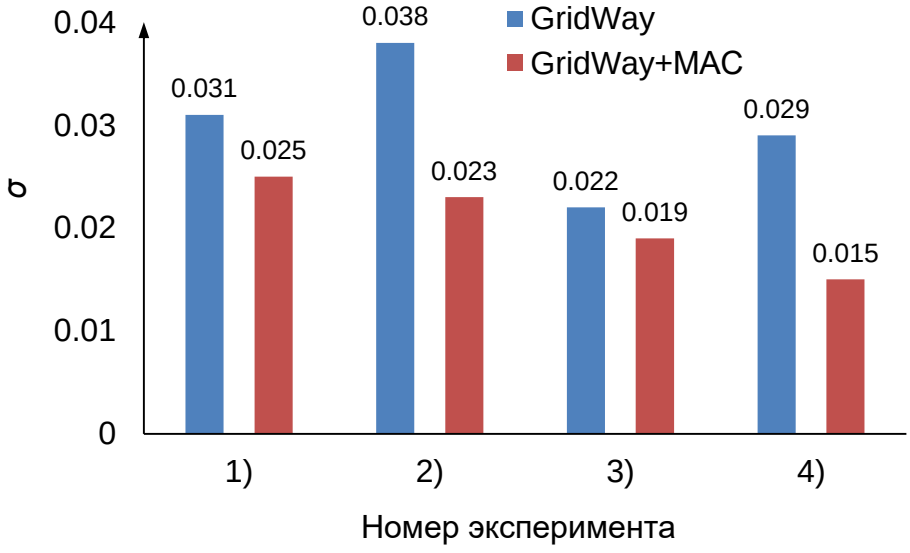


Рисунок 5.9 – Среднеквадратическое отклонение σ от k_g

Развитие системного ПО инструментального комплекса DISCENT. На основе системного ПО DISCENT разработаны микро-сервисные средства

поддержки имитационного моделирования работы природосберегающего оборудования инфраструктурных объектов в ГРВС [71]. В качестве такого оборудования могут выступать различные компоненты систем энергоснабжения [71, 357]. Схема организации вычислений для моделей на языке GPSS на основе предложенного микро-сервисного подхода показана на рисунке 5.10.

Рисунок 5.10 – Схема организации вычислений на основе предложенного микро-сервисного подхода

Все операции, связанные с подготовкой данных, выполнением GPSS-моделей и анализом полученных результатов, выполняются микро-сервисами. Микро-сервисы создаются на основе предопределенного набора родительских сервисов, реализующих базовые операции для организации и выполнения вычислений в гетерогенной среде, путем наследования, что обеспечивает автоматизацию и сокращение сроков разработки имитационных моделей исследуемых объектов. Доставка и настройка прикладного ПО на ресурсы, а также конфигурирование программно-аппаратных средств осуществляется агентами [63]. В таблице 24 приведены средства разработанного веб-интерфейса, а также родительских сервисов.

Таблица 24 – Компоненты веб-интерфейса и родительских сервисов

Компонент	Средства реализации	Операции
Веб-интерфейс	HTML, Bootstrap 4	– Обеспечение доступа к API; – создание нового эксперимента; – выбор модели и определение значений входных данных, изменяемых в допустимых диапазонах с определенным шагом; – выбор вычислительных ресурсов и прогон модели на основе многовариантных расчетов; – доступ к полученным результатам имитационного моделирования; – анализ этих результатов.
Родительский сервис обработки спецификации модели	Node.js	– Обработка спецификации модели; – динамическая настройка и создание веб-форм в соответствии со спецификацией; – извлечение необходимых данных из ретроспективной БД и преобразование их в целевой формат (Excel, CSV и т.д.).
Родительский сервис для обработки заданий	Node.js, GPSS, BAT-scripts	– Генерация вариантов входных данных с использованием их заданных диапазонов и шагов изменения значений; – генерация заданий; – отправка запроса на запуск сгенерированных заданий в СУПЗ; – выполнение BAT-скриптов, запускающих GPSS в пакетном режиме; – сбор отчетов GPSS (результатов моделирования) и размещение их в вычислительной БД.
Родительский сервис системы мониторинга	Nagios, Python	– Обработка, сбор и передача данных, полученных с помощью измерительной техники при мониторинге сложных технических объектов.
Родительский сервис по анализу результатов имитационного моделирования	Node.js	– Извлечение полученных значений наблюдаемых переменных из расчетной БД для каждого варианта входных данных; – преобразование извлеченных данных в целевой формат для многокритериального анализа; – применение методов многокритериального выбора оптимальных вариантов к преобразованным данным.

Для обеспечения взаимодействия с микро-сервисами разработан API на основе REST-подхода. В отличие от сервисов SOAP, ответы службы REST на запросы в формате JSON более компактные. Они также могут быть представлены в любом формате, включая XML, используемый службами SOAP.

Дополнительно разработана специальная спецификация модели в формате JSON [63]. Этот формат позволяет как описывать новую модель, так и

поддерживать сохранение, передачу и установку значений параметров для каждого варианта входных данных в рамках каждого эксперимента. Разработанная спецификация описывает следующие концептуальные объекты: параметры модели, методы их генерации, их источники и требования к ресурсам. На рисунке 5.11 представлен фрагмент спецификации, который включает ключи описания объектов без значений, а также комментарии к ним.

```
{
   "id": "",  // Уникальный идентификатор модели
   "name": "",  // Название экземпляра модели
   "model": "",  // Название исходной модели
   "title": "",  // Краткое описание экземпляра модели
   "description": "",  // Описание экземпляра модели
   "resources": [  // Перечень вычислительных ресурсов
      {
         "id": "",  // Уникальный идентификатор ресурса
         "name": "", // Название ресурса
         "address": ""” // IP-адрес ресурса
      },
      … ],
   "parameters": {  // Перечень параметров модели
      "inputFile": "",  // Путь до внешнего модуля (файла) модели
      "dataPath": "" ,  // Рабочий каталоги модели
      "input": [  // Перечень входных параметров
         {
            "type": "",  // Тип параметра (string, number, path)
            "name": "",  // Имя параметра в модели
            "gpssName": "",  // Имя параметра в GPSS
            "title": "",  // Краткое описание параметра
            "value": "",  // Значение параметра
         },
       … ],
      "output": [  // Перечень выходных параметров
         {
            "type": "",  // Тип параметра (string, number, path)
            "name": "",  // Имя параметра в модели
            "gpssName": "",  // Имя параметра в GPSS
            "title": "",  // Краткое описание параметра
            "value": "",  // Значение параметра
         },
        … ],
…
},
   "commands": {  // Консольные команды
      "start": ""  // Строка запуска вычислений в пакетном режиме
      …
   }
```

Рисунок 5.11 – Фрагмент спецификации

Сервис обработки заданий передает их СУПЗ, которая запускает необходимое число ВМ на ресурсах и управляет их выполнением. Экземпляры модели GPSS с соответствующими вариантами данных передаются по SSH для запуска на ВМ, работающих под управлением ОС Windows.

В отличие от системы моделирования общего назначения GPSS данные средства поддерживают ряд специальных возможностей, отсутствующих или характеризующихся их частичным предоставлением в вышеупомянутой системе. В числе таких возможностей: поддержка научных рабочих процессов (workflow), обеспечение использования мультиоблачной среды, многовариантные расчеты с последующим многокритериальным анализом результатов моделирования и другие функциональные особенности. В тоже время, данные средства поддерживают использование готовых GPSS-моделей и запуск системы GPSS в пакетном режиме.

5.2. Технологические аспекты создания пакетов прикладных программ в гетерогенной распределенной вычислительной среде

Развитие распределенных вычислительных систем и их активное применение в процессе решения сложных задач обуславливает необходимость их декомпозиции на взаимосвязанные подзадачи, выполняемые в разных узлах. Подзадачи должны обладать достаточной вычислительной сложностью для обеспечения загрузки ресурсов и снижения накладных расходов на обмены между процессами их решения.

Для реализации такой схемы решения сложных задач хорошо подходит модульное программирование, представляющее собой интеграцию двух базовых направлений программирования – процедурного и сборочного.

При процедурном программировании предполагается, что каждая подзадача, полученная в результате декомпозиции сложной задачи, является относительно независимой, допускает самостоятельное решение и реализуется отдельной прикладной программой (модулем) [317, 393]. В этом случае процесс декомпозиции сложной задачи на подзадачи и их программная реализация в виде

модулей называется модуляризацией [252]. Модуль является первичной частью ПО, выделенной по тем или иным причинам [283]. В качестве модулей могут выступать фрагменты исходных текстов программ, текстов исходных данных, текстов процедур, применяемых к программным материалам, а также описания атрибутов этих фрагментов – спецификации языка программирования, входных и выходных переменных, их форматов, накладываемых на них ограничений, параметров настройки, требований к вычислительной системе, прав доступа и других характеристик. Объектные или загрузочные модули, библиотеки объектных модулей и другие вторичные программные компоненты формируются путем применения типовых процедур обработки (трансляции, компиляции и других операций) к выделенной первичной части ПО. Запросы к базам данных также могут рассматриваться в качестве модулей.

Сборочное программирование основано на применении многократно используемых программ [394]. Создание ПО осуществляется путем подбора, настройки и организации совместного применения совокупности многократно используемых модулей, размещенных в общедоступных хранилищах или сформированных в виде библиотек программ [281]. Задача модуляризации при построении библиотеки заключается в формировании общеупотребительного набора модулей (например, подпрограмм или классов), позволяющих решать характерные типовые подзадачи.

Пакет прикладных программ является средством автоматизации модульного программирования в некоторой предметной области при решении определенного класса задач – совокупности прикладных проблем, характеризуемых большим числом частичных подзадач, решаемых на основе общих алгоритмов и данных [260].

В рамках традиционного подхода [252] ППП – это комплекс взаимосвязанных прикладных программ и средств системного обеспечения (программных и языковых), предназначенный для автоматизации решения определенного класса задач. В структуре ППП выделяют три основных компонента: функциональное наполнение, язык заданий и системное наполнение

(рисунок 5.12).

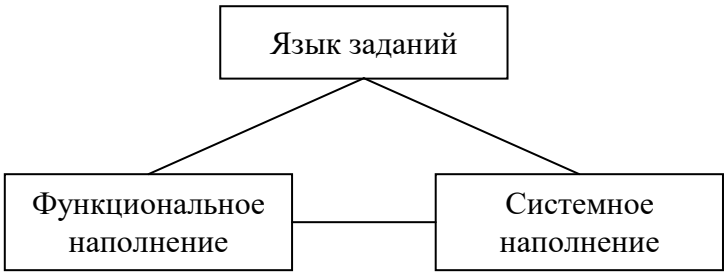


Рис. 5.12 – Структура ППП

Функциональное наполнение отражает специфику предметной области пакета и представляет собой совокупность прикладных программ (модулей).

Языковые средства обеспечивают описание предметной области решаемых задач; конструирование схем (планов) решения задач, в которых указывается порядок выполнения и взаимодействия модулей при решении определенной задачи; управление процессами генерации расчетной программы по схеме решения задачи; формирование заданий вычислительной системе пользователя по исполнению расчетных программ. Системное наполнение представляет собой совокупность программ, обеспечивающих формирование и выполнение заданий, а также взаимодействие пользователя с ППП.

Развитие технологий параллельного программирования привело к существенным изменениям архитектуры современных ППП (рисунок 5.13), разрабатываемых для работы в ГРВС. Такая среда состоит из вычислительных узлов, объединенных коммуникационной сетью.

Вычислительный узел представляет собой программно-аппаратный ресурс, включающий модуль оперативной памяти, один или несколько процессоров, жесткий диск и системное ПО (совокупность программных средств, поддерживающих функционирование узла в ГРВС). Один из вычислительных узлов назначается главным узлом и наделяется функциями управления заданиями пользователей и ресурсами ГРВС. Как следствие, возникают новые требования к

средствам описания и обработки схем решения задач со стороны пользователей ППП.

Рисунок 5.13 – Структура современного ППП

В общем случае архитектура ППП должна включать вычислительные ресурсы, в которых будут выполняться модули пакета; связующее ПО для поддержки взаимодействия между функциональным и системным наполнением ППП и вычислительными ресурсами; средства виртуализации компонентов ППП, обеспечивающие открытый, прозрачный доступ пользователей к этим компонентам.

Можно выделить два основных подхода к организации функционального наполнения пакета и решению прикладных задач в ГРВС. В рамках первого подхода все вычислительные модули (прикладные программы пакета) находятся в одном узле ГРВС. По схеме решения задачи производится синтез целевой параллельной программы с последующей ее компиляцией и запуском на распределенной вычислительной установке. При втором подходе вычислительные модули размещаются в разных узлах ГРВС. Схема решения задачи выполняется в

режиме интерпретации.

Появление развитого базового ПО [398], реализующего организацию расчетов в ГРВС, является основой для массовых параллельных и распределенных приложений. Однако анализ методов и средств организации такого рода вычислений выявил ряд проблем научно-технического характера: разработка распределенных и параллельных приложений выполняется зачастую «точечно», в привязке к узкому классу задач из конкретной предметной области; создаваемые приложения плохо интегрируются из-за использования различного связующего ПО, форматов данных и протоколов их передачи, а также использования разных моделей программирования приложений, планирования вычислительных процессов и загрузки вычислительных ресурсов; круг потенциальных пользователей таких приложений достаточно ограничен, что обусловлено сложностью их освоения и применения прикладными специалистами для решения своих задач.

В частности, обеспечение масштабирования потоков заданий, порождаемых ППП в ГРВС, является в настоящее время одним из фундаментальных и практически важных направлений исследований по организации проблемно-ориентированных ГРВС. Необходимость создания масштабируемых ППП возникает при выполнении многовариантных расчетов, решении широкого спектра переборных задач и многих других.

Предполагается, что масштабируемое приложение включает набор прикладных программ для параллельного решения задачи с помощью различных вычислительных единиц (например, ядер) разнородных узлов ГРВС и порождает комбинированный поток, объединяющий задания для этих прикладных программ. При этом вычислительная нагрузка, связанная с решением задачи, распределяется между вычислительными единицами разнородных узлов ГРВС, а время выполнения заданий комбинированного потока уменьшается обратно пропорционально числу используемых вычислительных единиц с учетом их производительности в составе конкретного узла ГРВС. Создание системы управления комбинированными потоками заданий для ППП является

нетривиальной и весьма актуальной проблемой.

В течение длительного времени в качестве связующего ПО для выполнения распределенных приложений в Грид используются хорошо известные инструменты GT, HTCondor, Berkeley Open Infrastructure for Network Computing (BOINC) или X-COM. В то же время активно разрабатываются и применяются с той же целью системы управления рабочими процессами. В их числе Askalon, Karajan, Kepler, Pegasus, Taverna, Triana и другие системы [197]. Зачастую эти системы функционируют совместно со связующим ПО.

Однако возможности вышеупомянутых инструментов и систем часто ограничивают потенциальные возможности современных научных приложений в рамках вычислительной модели Грид [201].

В то же время по мере совершенствования аппаратного обеспечения (появления и широкого распространения многоядерных процессоров, а также существенного увеличения емкости устройств хранения данных), активно развиваются облачные вычисления. Облако представляет собой совокупность доступных виртуализованных ресурсов (вычислительных инфраструктур, систем хранения данных, приложений, сервисов и других программно-аппаратных компонентов), предоставляемых по запросам пользователей. Как правило, облачные ресурсы выделяются и перераспределяются динамически, в зависимости от изменяющейся вычислительной нагрузки. Облачный провайдер обеспечивает требуемый уровень качества обслуживания приложений пользователей.

Облачные вычисления являются результатом развития грид-вычислений. Их модели взаимно дополняют друг друга. Обе модели предназначены для обеспечения доступа к мощным вычислительным ресурсам и большим данным, а также их интенсивного использования.

Облачные вычисления используют виртуализацию для обеспечения однородного интерфейса с динамически масштабируемыми базовыми ресурсами с тем, чтобы уровень виртуализации скрывал их физическую гетерогенность и географическую распределенность, а также отказы программно-аппаратных средств. Однако развертывание данных и приложений на облачной платформе, как

правило, ограничивает пользователя только одним провайдером или требует дополнительных затрат для повторения этого процесса в другом облаке. Использование концепции Gird позволяет объединить различные облачные платформы путем использования стандартизированных интерфейсов для доступа к распределенным ресурсам и их контроля.

Грид-вычисления могут получить дополнительные преимущества от интеграции с облачными вычислениями за счет использования новых коммерчески доступных ресурсов хранения и вычислительных ресурсов, а также путем развертывания облачных технологий на ресурсах Грид для улучшения управления ими и повышения надежности этих ресурсов за счет виртуализации.

Таким образом, направление исследований, связанное с интеграцией моделей облачных и грид-вычислений, становится чрезвычайно актуальным.

Такая интеграция приводит к необходимости решения задач унификации интерфейса компонентов среды, адаптации приложений и их масштабируемости, обеспечения качества обслуживания, смягчения неопределенности различных видов, интеллектуализации системы управления ресурсами, мониторинга гетерогенных ресурсов и т.д. За последние десять лет получено много научных и практических результатов для решения вышеуказанных проблем.

В [175] предлагается подход к конвергенции облачных и грид-технологий в сети следующего поколения посредством разработки стандартов для их интерфейсов и окружений, поддерживающих множество реализаций архитектурных компонентов.

Рассмотрена способность гибридной вычислительной платформы к адаптации при изменении требований заданий к ресурсам и качества их обслуживания [121].

Представлена гибридная архитектура высокопроизводительной вычислительной инфраструктуры, которая обеспечивает предсказуемое выполнение научных приложений и масштабируемость при изменении числа доступных ресурсов с разными характеристиками, владельцами, политиками и географическими местоположениями [147].

Реализована облачная платформа приложений Aneka, предоставляющая ресурсы различных инфраструктур, включая облака, кластеры и добровольные вычисления в Грид, в рамках проведения крупномасштабных научных исследований [208].

Предложен подход к интеграции облачных и грид-ресурсов с использованием системы управления базой данных для развертывания изображений ВМ в облачных средах по запросам приложений, выполняемых в Грид [146].

В диссертационной работе разработка приложений для проведения научных и прикладных исследований осуществляется с помощью двух инструментальных комплексов Orlando Tools и DISCOMP. Эти комплексы обеспечивают построение предметно-ориентированных вычислительных сред, в которых интегрируются различные вычислительные инфраструктуры, поддерживающие как облачные, так и грид-вычисления.

5.3. Инструментальные средства создания и применения распределенных пакетов прикладных программ

Как правило, РППП ориентируются на класс задач, характеризующихся следующими свойствами:

- решение задачи требует проведения расчетов с использованием больших объемов вычислительных ресурсов (процессорного времени, оперативной памяти, дискового пространства и других характеристик);
- возможна декомпозиция общей сложной задачи на более простые (с вычислительной точки зрения) подзадачи;
- процесс решения общей задачи подразумевает распределенное решение набора ее взаимосвязанных подзадач;
- не предполагается интенсивного взаимодействия между параллельными вычислительными процессами; задача допускает декомпозицию данных на блоки и независимую параллельную обработку этих блоков;

– решение задачи выполняется, как правило, по одной слабо меняющейся схеме, требующей динамического управления процессом вычислений.

К такому классу относятся многие переборные задачи, включая задачи с многовариантными расчетами.

Инструментальный комплекс DISCOMP [373]. Функциональное наполнение РППП, разрабатываемых с помощью инструментального комплекса DISCOMP, составляют модули, представленные в виде исполняемых в пакетном режиме программ, реализованных на различных языках программирования (например, C, Fortran, Pascal). Модули размещаются в разных узлах ГРВС. В каждом узле может быть установлено несколько модулей.

Допустимо включение унаследованного ПО и приложений, размещенных в специализированных узлах среды и представленных сервисами, в состав функционального наполнения РППП. Для использования унаследованного ПО и сервисов приложений в пакете создаются специальные модули-оболочки, в которых реализуется их вызов (рисунок 5.14). Эти модули осуществляют специализацию функций вызываемого ПО путем их параметрической настройки.

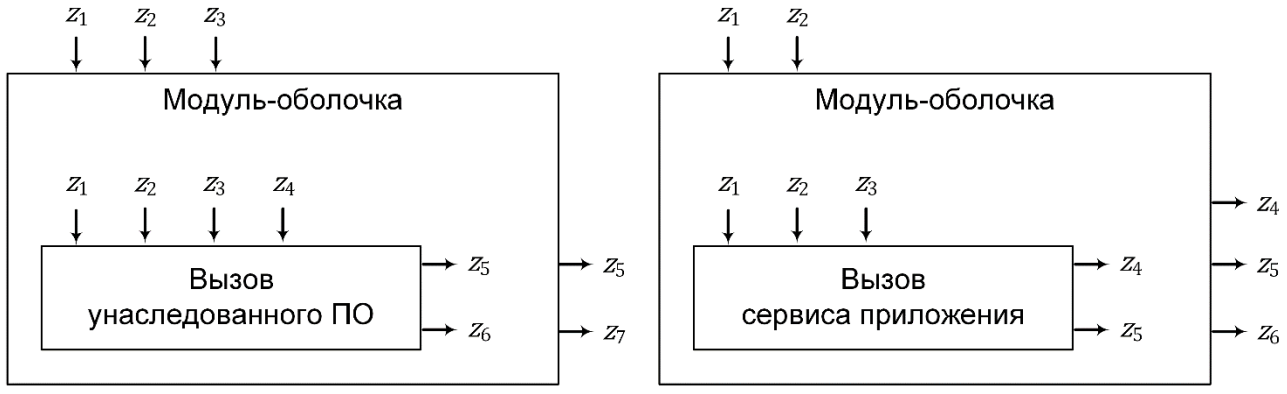


Рисунок 5.14 – Модули-оболочки для вызова унаследованного ПО (а) и сервиса приложения (б)

Обмен данными между модулями пакета осуществляется через файлы.

Системная часть РППП включает мультиагентные компоненты

инструментального комплекса DISCOMP для управления распределенными вычислительными процессами и обменом данных. Эти компоненты являются платформонезависимыми и могут функционировать под управлением различных ОС, например, Microsoft (MS) Windows, Linux и Macintosh Operating System (Mac OS). Набор компонентов, включаемых в системную часть пакета, определяется его разработчиком. Модуль-оболочка для вызова сервиса приложения реализована с использованием возможностей инструментария Everest [194].

Управление вычислениями реализуется системой агентов. Решение задачи выполняется, как правило, по одной слабо меняющейся схеме, требующей динамического управления процессом вычислений. Схема решения задачи строится в параллельно-ярусной форме на основе процедурной постановки задачи.

Выполнение плана решения задачи осуществляется в режиме интерпретации. Предварительная классификация заданий специальным агентом позволяет рационально назначать ресурсы модулям. Инструментальный комплекс DISCOMP поддерживает виртуализацию ресурсов ГРВС.

Если программно-аппаратные ресурсы какого-либо узла среды не соответствуют требованиям модуля схемы решения задачи, то для выполнения такого модуля может быть запущена ВМ. ВМ, запущенные в разных узлах среды, которые находятся под управлением различных систем (СУПЗ, метапланировщиков, гипервизоров и др.), объединяются в единую виртуальную среду РППП. Управление заданиями в ВМ, запущенных перечисленными выше системами, осуществляется агентами инструментального комплекса DISCOMP.

Вычислительная модель и схемы решения задач описываются в текстовом виде с использованием расширения языка XML. Описание включает следующие основные секции: спецификации параметров, модулей и постановок задач. На рисунке 5.15 представлены фрагменты текста описания модуля и его параметров (рисунок 5.15 а), а также постановки задачи (рисунок 5.15 б), включающей этот модуль.

```
<!-- секция спецификаций параметров -->
<parameters>
   <param name='list' type='filelist'
          filepattern='list_%1.txt'/>
   <param name='result' type='filelist'
          filepattern='result_%1.txt'/>
</parameters>

<!-- секция спецификаций модулей -->
<modules>
   <module name='solver'>
      <commands os='Linux'>
         <start>./solver</start>
         <stop>./stop.sh</stop>
      </commands>
      <parameters>
         <input>
            <param name='list'/>
         </input>
         <output>
            <param name='result'/>
         </output>
      </parameters>
   </module>
</modules>
```

а)

```
<scheme>
  <stage>
     <listmodule name='solver'
         onFinish='checkResult($el_num)'/>
  </stage>
</scheme>

<control><![CDATA[
  function checkResult (el_num) {
     // получить значение выходного элемента
     // параметра-списка result с порядковым
     // номером el_num
     var res = DiscompAPI.getLPV('result', el_num);
     // если значение элемента содержит подстроку
     // "RESULT FOUND", то остановить вычисления на
     // текущем ярусе и перейти к следующему
     if ( res.match(/RESULT FOUND/gi) ) {
        DiscompAPI.stopStageModules ();
     }
  }
]]></control>
```

б)

Рисунок 5.15 – Описание модуля (а) и схемы решения задачи (б) в инструментальном комплексе DISCOMP

Элемент parameters описывает параметры предметной области, которые используются в процессе решения задач пакета. Каждый параметр снабжен набором атрибутов: уникальным идентификатором, концептуальным именем, типом и соответствующими типу атрибутами. К допустимым типам параметров относятся файл и список файлов. Дополнительным атрибутом параметра типа файл является имя файла на диске для этого параметра. Дополнительными атрибутами параметра типа список файлов являются число элементов списка и шаблон имен файлов на диске для этих элементов.

Элемент modules содержит спецификации, определяющие назначение модулей пакета, атрибуты (идентификатор, имя модуля), команды запуска, а также множество входных и выходных параметров этих модулей.

Элемент scheme описывает схемы решения задач предметной области пакета. Схемы задаются в ярусно-параллельной форме. Модули, расположенные на одном ярусе схемы, могут выполняться параллельно. Модуль, расположенный на i+1-м ярусе схемы, может быть запущен на выполнение, если выполнены все модули на i-м ярусе.

Подход к управлению вычислительным процессом в инструментальном комплексе DISCOMP базируется на механизмах обработки событий, возникающих в процессе выполнения схемы решения задачи. Он включает две фазы:

– проверка условий применения обработки событий;
– применение операций, созданных пользователем пакета и предназначенных для обработки события и выбора требуемого управления.

Языковые средства обеспечивают задание условий обработки следующих событий:

– запуск и завершение выполнения модуля или группы модулей;
– вычисление значений управляющих параметров, мониторинг которых производится с заданной интенсивностью;
– приближение характеристик ресурсов узла среды к их критическим значениям;
– отказ ресурсов узла и другие действия, совершенные в процессе вычислений.

Реализация обработчиков событий разработчиком РППП задается в виде функций на языке программирования JavaScript [295] и включается в схему решения задачи. Такой обработчик реализуется в виде набора операторов на языке JavaScript. Язык JavaScript расширен программным интерфейсом DiscompAPI, позволяющий пользователю взаимодействовать с исполнительной подсистемой инструментального комплекса. Интерфейс DiscompAPI предоставляет следующие функциональные возможности: получение/изменение значений параметров, остановку/запуск процесса выполнения модуля (всех модулей на ярусе), передачу управления на нужный ярус.

Корректность обработки событий средствами DiscompAPI обеспечивается ограничениями целостности, накладываемыми в процессе вычислений на отношения между объектами предметной области (параметрами, модулями и значениями параметров). Пример задания условия завершения модуля и операции для обработки данного события приведен на рисунке 5.15 б.

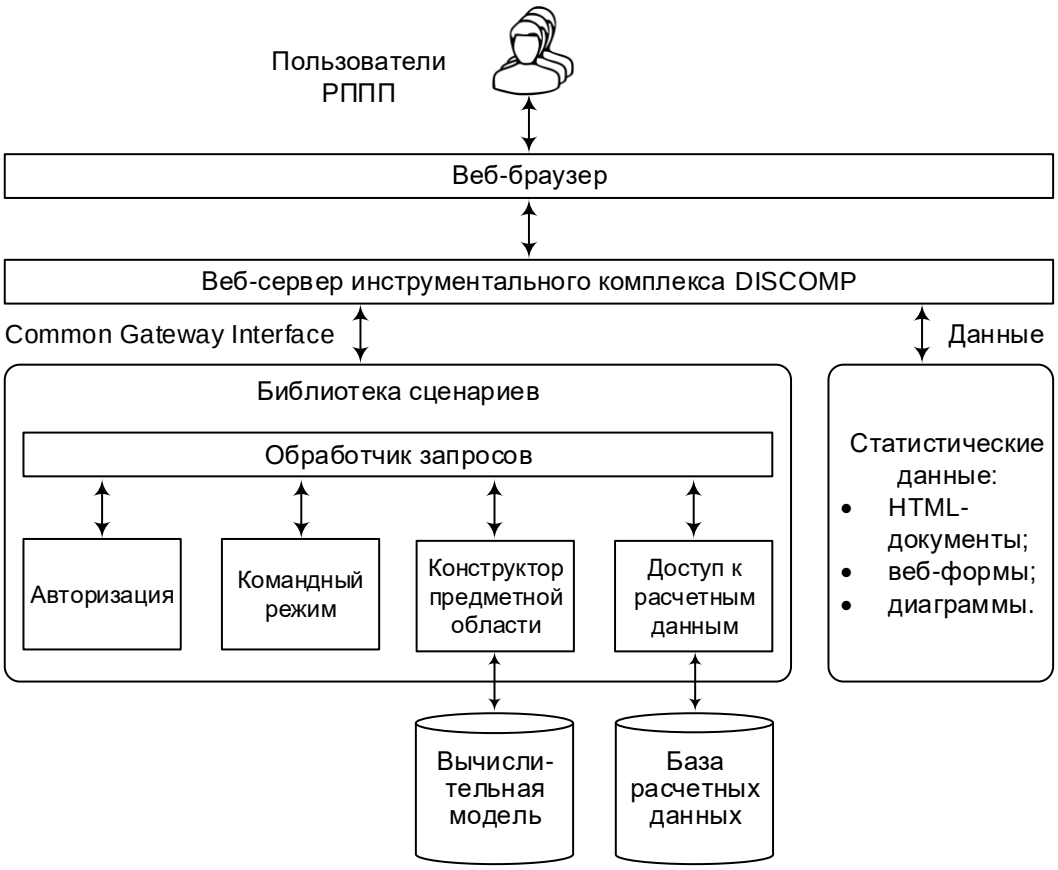


Рисунок 5.16 – Архитектура веб-интерфейса DISCOMP

Интерактивный пользовательский веб-интерфейс DISCOMP поддерживает режимы работы со следующими тремя категориями пользователей РППП: разработчик, администратор и конечный пользователь пакета (рисунок 5.16). Эти режимы регламентируют разграничение правил доступа пользователей разных категорий к компонентам РППП.

Разработчик пакета описывает объекты предметной области с помощью специализированного конструктора. Создание вычислительного модуля включает следующие этапы: присвоение уникального имени, создание директории для размещения файлов модуля, загрузку исполняемой программы (включая системные библиотеки и сопутствующие файлы, необходимые для запуска на любом вычислительном узле) и формирование файла спецификации. Спецификация модуля включает описания всех входных и выходных параметров модуля, команд для запуска (остановки) исполняемой программы, ограничений на время выполнения, а также системных требований к вычислительным узлам, на которых будет производиться выполнение данного модуля. Когда все параметры и

модули пакета описаны, разработчик формирует процедурные постановки задач и определяет перечень управляющих конструкций, которые будут обработаны интерпретатором схемы решения задачи.

Администратор обладает возможностью создания и декларирования прав конечных пользователей пакета, определения лимитов на использование ими вычислительных ресурсов, выбора приоритетов заданий, управления вычислительными ресурсами распределенной среды и выполнения прочих системных функций.

Конечному пользователю предоставляется возможность работы с готовым к исполнению РППП, в том числе задание значений входных параметров, запуск и остановка процесса вычислений, использование механизма контрольных точек (запуск задания с его последнего сохраненного состояния), просмотр статуса выполнения вычислительного процесса, анализ статистики и работа с результатами вычислений. Процесс запуска задания включает следующие этапы: выбор РППП из списка доступных, выбор процедурной постановки задачи, означивание входных параметров и постановка сформированного задания в очередь СУПЗ для его последующего выполнения. После запуска задания пользователь имеет возможность контролировать ход его выполнения с помощью графического представления схемы решения задачи.

Инструментальный комплекс Orlando Tools [74, 368]. Данный комплекс включает более широкие возможности для представления в пакете специфики предметной области и решаемых задач. Помимо алгоритмических знаний и постановок задач вычислительная модель Orlando Tools позволяет также описывать схемные и продукционные знания.

Инструментальный комплекс Orlando Tools предоставляет пользователям как текстовый, так и графический языки описания предметной области РППП и постановок задач. В текстовом виде описание осуществляется с использованием специализированного расширения языка XML [392]. Элементы входного языка описания модели предметной области РППП в инструментальном комплексе Orlando Tools приведены на рисунке 5.17.

```
<module>
  <name>Имя_модуля</name>
  <parameters>Входные_параметры>
    Выходные_параметры</parameters>
  <signature>Команда_запуска</signature>
  <cores>Число_ядер</cores>
  <walltime>Время_останова</walltime>
  <run_mode>Режим_запуска</run_mode>
</module>
```

а)

```
<operation>
  <name>Имя_операции</name>
  <parameters>Входные_параметры>
    Выходные_параметры</parameters>
  <run_condition>Условие_запуска</run_condition>
  <while_flag>Признак_повторения</while_flag>
  <module_name>Имя_модуля</module_name>
  <split_condition>Условие_расщепления</split_condition>
  <task_name>Имя_задачи</task_name>
</operation>
```

б)

```
<parameter>
  <name>Имя_параметра</name>
  <extention>Расширение</extention>
  <list>Число_элементов</list>
</parameter>
```

в)

```
<task>
  <name>Имя задачи</name>
  <parametrs>Входные_параметры>
                    Выходные_параметры</parametrs>
  <operations>Список_операций</operations>
</task>
```

г)

Рисунок 5.17 – Элементы входного языка описания модели предметной области РППП в инструментальном комплексе Orlando Tools: module (а), parameter (б), operaton (в) и task (г)

Элемент module предназначен для спецификации объектов алгоритмического слоя знаний. Данная спецификация включает информацию об исполняемом программном модуле, входных и выходных параметрах модуля, а также инструкции для запуска и выполнения модуля. Вложенные элементы name, parameters, signature, cores, walltime и run_mode содержат соответственно следующие данные (риунок 5.17 а): идентификатор модуля (Имя_модуля); списки формальных входных и выходных параметров модуля (Входные_параметры > Выходные_параметры); команду запуска модуля (Команда_запуска), включающую имя исполняемой программы и опции ее запуска, в том числе регулярные выражения, определяющие структуры данных для параметров модуля и способы их использования; максимально допустимое время выполнения модуля в часах (Время_останова); режим (Режим_запуска), определяющий запускается MPI-программа или нет.

Функциональное наполнение ППП, реализующее алгоритмический слой знаний, включает библиотеку модулей для решения задач в узлах с различными вычислительными характеристиками, библиотеку модулей для препроцессорной и постпроцессорной обработки данных, библиотеку модулей, реализующих предметно-ориентированные процессы декомпозиции задач на подзадачи. Здесь и

ниже список объектов (параметров или операций) представляет собой список идентификаторов объектов, разделенных запятыми.

Элемент parameter используется для спецификации объектов схемного уровня знаний – параметров предметной области ППП. Множество параметров включает два вида параметров: базовый параметр и параметр-список. В вычислительной системе базовый параметр представлен произвольным файлом данных неопределенной структуры. Параметр-список формируется на основе некоторого базового параметра, предназначается для представления множества вариантов значений (файлов данных) этого базового параметра и их параллельной обработки.

Элемент operation служит для спецификации объектов схемного уровня знаний – операций предметной области ППП. Вложенные элементы name, parameters, run_condition, while_flag, module_name, split_condition и task_name содержат соответственно следующие данные (риунок 5.17 б): идентификатор операции (Имя_операции); списки входных и выходных параметров операции (Входные_параметры>Выходные_параметры); логический предикат (Условие_запуска), определяющий необходимость запуска операции, все значения входных параметров которой известны; признак повторного запуска операции (Признак_повторения); идентификатор модуля, реализующего операцию (Имя_модуля); логический предикат (Условие_расщепления), определяющий необходимость динамической декомпозиции подзадачи, решаемой с помощью операции; новая постановка задачи для динамической декомпозиции (Имя_задачи).

Элемент task служит для спецификации постановки задачи, формулируемой в процедурной или непроцедурной форме. Процедурная форма постановки задачи определяется списком операций, которые нужно выполнить для решения задачи. В случае непроцедурной постановки задачи список операций определяется автоматически путем планирования на модели предметной области ППП на основе списков заданных и искомых параметров. Вложенные элементы name, parameters и operations содержат соответственно следующие данные (рисунок 5.17 г): идентификатор постановки задачи (Имя_задачи); списки входных и выходных

параметров постановки задачи (Входные_параметры> Выходные_параметры); список операций (Список_операций), необходимых для решения задачи. Схема может включать управляющие конструкции ветвления, цикла и рекурсии.

Графическое описание предметной области РППП и постановок задач автоматически переводится во внутреннее представление в терминах языка XML. Схема решения задачи выполняется в режиме интерпретации. На рисунке 5.18 приведен пример графического представления схемы решения задачи в пакете Градиент для исследования многоэкстремальных функций методом мультистарта [350]. Проверка корректности и целостности описания предметной области РППП выполняется на этапах его трансляции с входного языка во внутреннее состояние, которое хранится в базе знаний пакета.

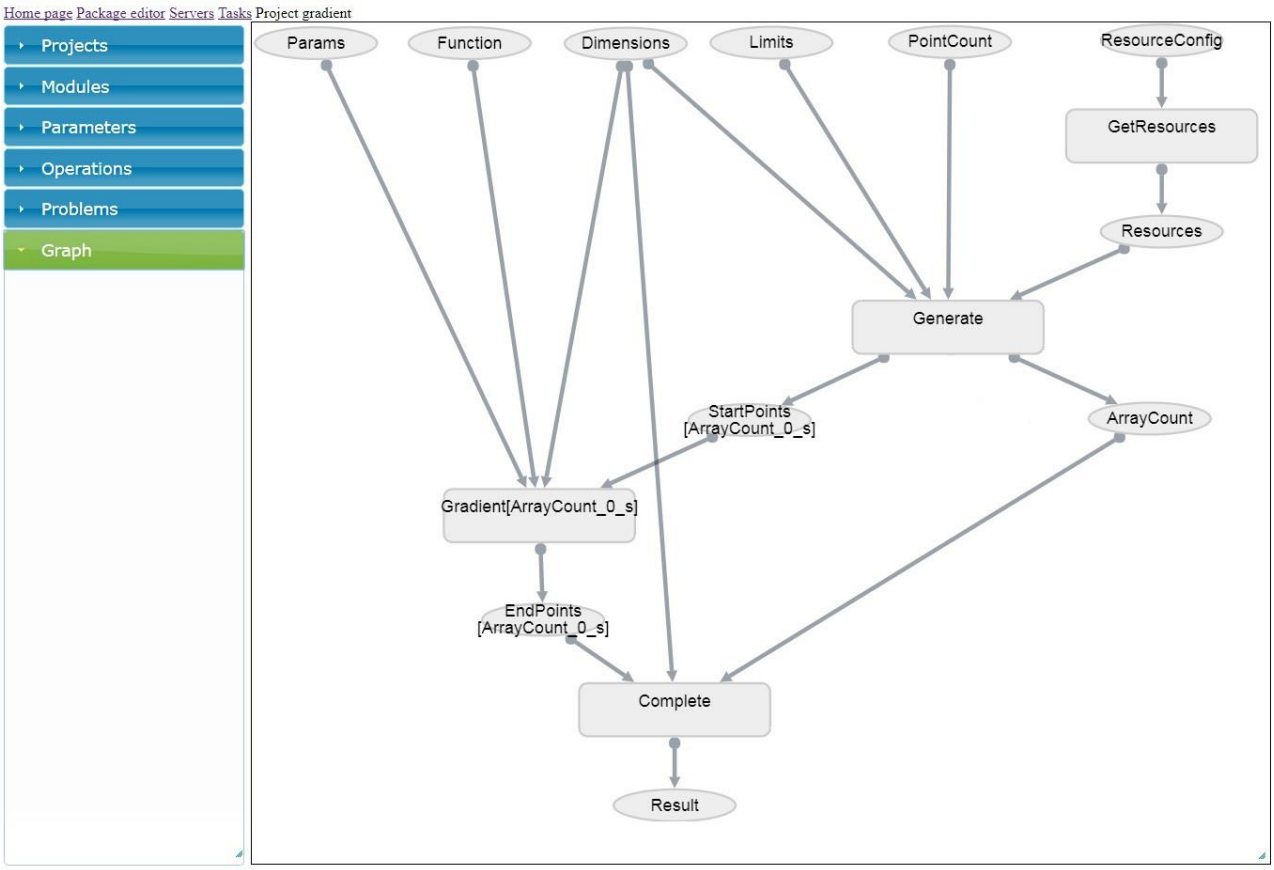


Рисунок 5.18 – Графическое представление схемы решения задачи

Графическое описание предметной области РППП и постановок задач автоматически переводится во внутреннее представление в терминах языка XML.

Схема решения задачи выполняется в режиме интерпретации. На рисунке 5.18 приведен пример графического представления схемы решения задачи в пакете Градиент для исследования многоэкстремальных функций методом мультистарта [350]. Проверка корректности и целостности описания предметной области РППП выполняется на этапах его трансляции с входного языка во внутреннее состояние, которое хранится в базе знаний пакета.

Исполнительная подсистема инструментального комплекса Orlando Tools включает набор интерпретаторов схем решения задач и планировщиков вычислений. Интерпретатор производит обработку управляющих конструкций и выполнение операций схем решения задач. Планировщик осуществляет декомпозицию схем решения задач на подсхемы с целью оптимизации распределения вычислительной и коммуникационной нагрузки на ресурсы ГРВС. Декомпозиция может быть выполнена как в статическом режиме до начала вычислений, так и динамически в процессе вычислений. Исходные данные и результаты вычислений хранятся в специальной базе расчетных данных.

По одному интерпретатору и планировщику размещается во всех управляющих узлах кластеров. В рамках децентрализованной архитектуры исполнительной подсистемы интерпретаторы и планировщики могут поддерживать локальные взаимодействия друг с другом с целью перераспределения вычислительной нагрузки. В целом исполнительная подсистема обеспечивает реализацию многоуровневого параллелизма в процессе решения задачи. Сервис-ориентированный доступ к компонентам Orlando Tools реализован с использованием стиля REST.

Поддержка приложений для работы с пространственно-временными данными. В настоящее время разработка и применение передовых цифровых технологий для мониторинга охраняемых природных территорий является чрезвычайно важной задачей. В частности, в ИДСТУ СО РАН разрабатывается сервис-ориентированная информационно-аналитическая среда (ИАС) [244], предназначенная для сбора, оцифровки, хранения и анализа пространственно-временных данных о различных аспектах жизненного цикла таких территорий.

Основным компонентом такой ИАС является геопортал, предоставляющий пользователям программные инструменты и WPS-сервисы для работы с данными. Применение WPS-сервисов упрощает обработку пространственно-временных и тематических данных, а также позволяет расширять функциональные возможности ИАС путем добавления новых WPS-сервисов.

Стандарт WPS-сервисов разработан консорциумом Open Geospatial Consortium (OGC) [38]. Он широко используется в геоинформационных приложениях [53, 153]. Данный стандарт обеспечивает их описание, обнаружение, вызов по запросу пользователя, контроль выполнения и взаимодействие с другими сервисами. Однако в случае, когда требуется высокопроизводительная обработка больших объемов геоданных, разработчики и пользователи WPS-сервисов сталкиваются с необходимостью использования параллельных или распределенных вычислительных систем. Таким образом, у них возникают нетривиальные проблемы, связанные с разработкой некоторого связующего ПО или адаптации сервисов к существующему, а также учетом всех аспектов системной архитектуры, особенностей интерконнекта, административных политик предоставления ресурсов, планирования вычислений и распределения ресурсов, взаимодействия с СУПЗ, установленными в узлах среды, и др. [198].

Поэтому в диссертации предложен подход к автоматизации выполнения ресурсоемких WPS-сервисов в ГРВС. Для их реализации разрабатывается научное приложение на основе workflow, которое выполняется под управлением мультиагентной системы. Агенты представляют собой разнородные ресурсы среды и распределяют вычислительную нагрузку между собой. Пользователь оформляет модули приложения в виде РППП, используя Orlando Tools. С помощью подсистемы непрерывной интеграции данного инструментального комплекса [64] они размещаются в узлах ГРВС. Данная подсистема обеспечивает также возможность дальнейшего сопровождения прикладного ПО в процессе модификации модулей, включая контроль их версий, отладку и тестирование в узлах ГРВС с разными вычислительными характеристиками. Схемы решения задач (workflow) приложения публикуются в виде WPS-сервисов.

В рамках Orlando Tools разработана новая подсистема, обеспечивающая автоматизацию создания, опубликования и применения WPS-сервисов. Отдельные модули и схемы решения задач, сформированные с помощью встроенного планировщика, автоматически публикуются (регистрируются) в виде асинхронных WPS-сервисов Orlando Tools на геопортале. Схемы решения задач могут включать вызовы других WPS-сервисов, обеспечивая возможность работы с композициями сервисов. В Orlando Tools поддерживается возможность обмена файлами между WPS-сервисами в качестве их параметров, включая обмен данными с системой хранения данных (СХД) ИСКЦ. Схема работы с WPS-сервисами приведена на рисунке 5.19.

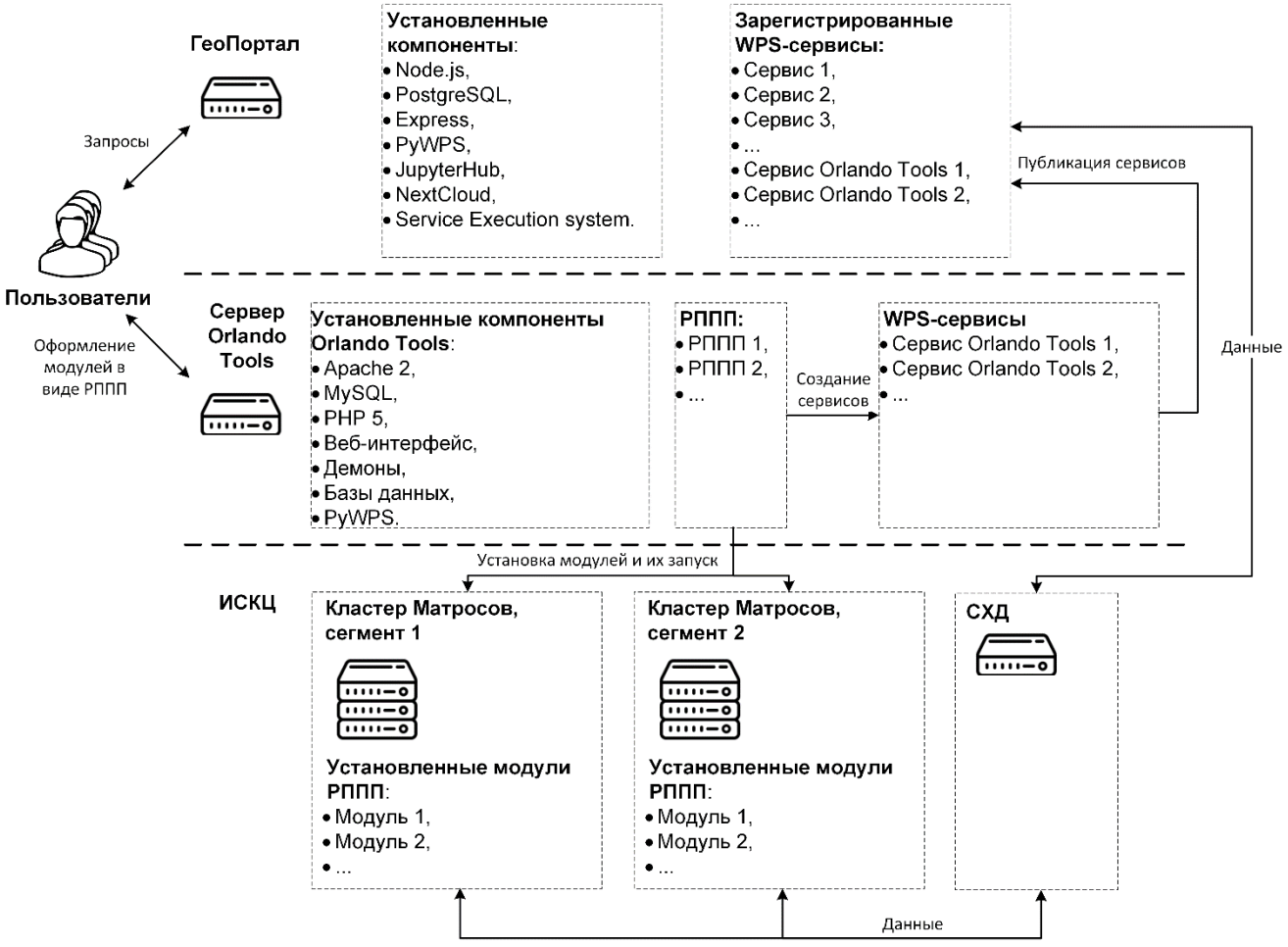


Рисунок 5.19 – Схема работы с WPS-сервисами

Алгоритм А.6. работы асинхронного WPS-сервиса Orlando Tools, опубликованного на геопортале, включает следующие этапы:

А.6.1. WPS-сервис получает запрос от пользователя.

А.6.2. WPS-сервис осуществляет проверку входных параметров, содержащихся в запросе. Если параметры корректны, то выполняется переход на следующий этап. Иначе он возвращает пользователю сообщение об ошибке, и его работа завершается.

А.6.3. WPS-сервис формирует XML-файл со статусом выполнения запроса и указывает URL XML-файла, в который будут помещены результаты выполнения запроса, или путь к СХД в случае реализации обмена данными между модулями с помощью текстовых файлов.

А.6.4. WPS-сервис генерирует вычислительное задание для Orlando Tools, вызывает диспетчера вычислительных работ этого инструментального комплекса, передает ему сгенерированное задание, а также информацию о входных параметрах запроса с помощью специализированного API, и завершает свою работу.

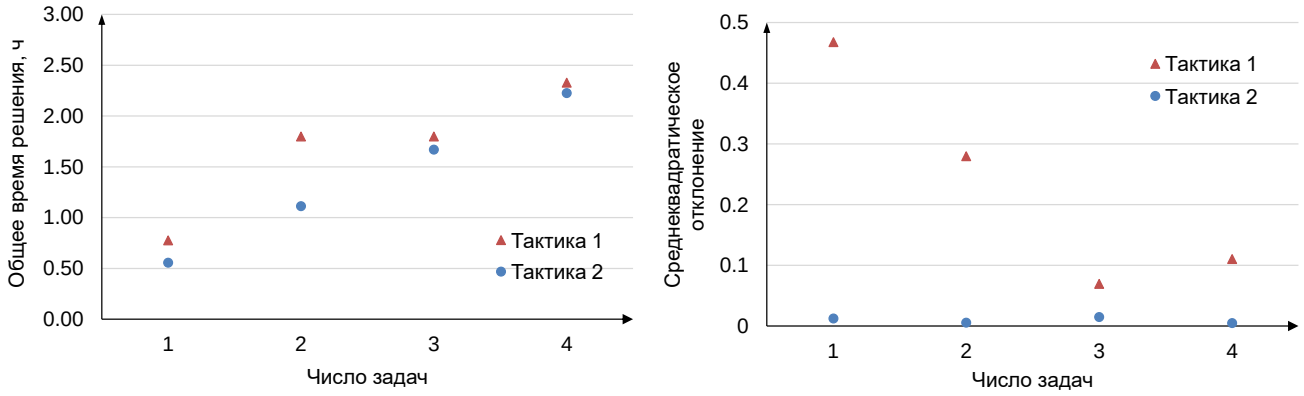
Получив управление, диспетчер Orlando Tools производит декомпозицию вычислительной работы по ресурсам, отправку подзаданий в очереди СУПЗ, установленных на этих ресурсах, передачу входных/выходных параметров в процессе выполнения модулей подзаданий и проверку статусов их выполнения. С заданной периодичностью он обновляет статус выполнения запроса в соответствующем XML-файле, сформированном WPS-сервисом. Процесс обновления выполняется вплоть до успешного завершения процесса решения задачи или обнаружения сбоя вычислительного процесса.

Таким образом, Orlando Tools выступает в роли связующего ПО и выполняет все необходимые системные операции по диспетчеризации вычислительных работ, передаче данных между вычислительными узлами, взаимодействию с СУПЗ, установленными на них, мониторингу состояния ресурсов, проверке статуса выполнения заданий и т. п. WPS-сервисы Orlando Tools выступают в качестве интерфейса между пользователем и ГРВС. Тем самым они расширяют функциональные возможности геопортала и обеспечивают возможность высокопроизводительной обработки данных.

Рассмотренный выше подход к созданию, опубликованию и применению WPS-сервисов апробирован в рамках проекта «Фундаментальные основы, методы и технологии цифрового мониторинга и прогнозирования экологической обстановки Байкальской природной территории» для решения задачи анализа временных рядов природно-климатических показателей окружающей среды инфраструктурных объектов Байкальской природной территории [379]. Данная задача решается в рамках моделирования процессов функционирования природосберегающего оборудования инфраструктурных объектов с целью оценки сокращения выбросов CO_2 при частичной замене используемых угольных котлов в системе теплоснабжения на тепловые насосы.

Чтобы продемонстрировать преимущества предложенного подхода к реализации ресурсоемких WPS-сервисов сравниваются две тактики для решения задач на двух кластерах ИСКЦ:

– запуск пользователем серии одиночных WPS-сервисов на кластерах вручную, используя GridWay;
– запуск потока заданий на кластерах диспетчером Orlando Tools по запросу его WPS-сервиса.



а) б)

Рисунок 5.20 – Сравнение показателей для двух тактик: а) общее время решения задач; б) среднеквадратическое отклонение загрузки ресурсов от средней загрузки

Используемые доступные ресурсы – по 10 узлов на каждом кластере. Характеристики узла первого кластера: 2x16 cores CPU AMD Opteron 6276, 2.3

GHz, 64 GB RAM. Характеристики узла второго кластера: 2×18 cores CPU Intel Xeon X5670, 2.1 GHz, 128 GB RAM. На рисунке 5.20 показаны общее время решения задач в часах (а) и балансировка ресурсов (б), оцененная с помощью среднеквадратического отклонения загрузки узлов от средней загрузки, для 20 используемых узлов.

Очевидно, что показатели для второй тактики, реализуемой с помощью Orlando Tools, существенно превосходят соответствующие параметры первой тактики. Это подтверждает целесообразность применения предложенного подхода.

5.4. Технология предметно-ориентированных вычислений

В настоящее время проекты, связанные с разработкой и применением специализированных сред организации распределенных вычислений в различных предметных областях, являются чрезвычайно актуальными. Широко известными примерами таких сред являются распределенная Грид-инфраструктура для поддержки проведения экспериментов в области физики высоких энергий на ускорительном комплексе Большого адронного коллайдера [4], интегрированная среда федеративных гетерогенных ресурсов для поддержки проведения таких экспериментов в российском сегменте Грид [247], открытая веб-платформа для геномных исследований [97], облачная платформа для быстрого анализа биомедицинских данных [7], программный инструментарий для интеграции информационно-алгоритмических ресурсов в глобальной сети при решении прикладных задач [226], интеллектуальная оболочка управления параллельными вычислительными процессами в распределенной иерархической среде, включающей в себя вычислительные системы различной архитектуры [240], система моделирования критических инфраструктур электроэнергетики в распределенной вычислительной среде [163]. Как правило, схема решения задачи в таких средах тесно коррелирует с понятием workflow, используемым при описании потока работ, отдельные этапы которых реализуются в виде программных модулей.

Следует отметить, что переборные задачи, в которых поиск решения

осуществляется на основе многовариантных расчетов, составляют значительную долю больших NP-трудных задач, решаемых в подобных проектах с привлечением высокопроизводительных вычислений.

ГРВС, применяемая для организации предметно-ориентированных вычислений, предоставляет совокупность программно-аппаратных средств своему конечному пользователю для решения им определенного класса задач из некоторой предметной области, задействуя необходимые ресурсы среды. Такая среда должна обеспечивать гибкие языковые возможности и программные средства построения модели предметной области, а также ее использования в процессе формулировки задач и их решения конечным пользователем.

В диссертационной работе предложена технология предметно-ориентированных вычислений в ГРВС, интегрирующей модели облачных и грид-вычислений [66, 68, 69, 79, 325, 340, 368, 389]. Данная технология включает в себя следующую совокупность компонентов:

- методы и инструментальные средства построения АМ ГРВС;
- модели, алгоритмы, методы и программные средства оценки показателей выполнения вычислительных процессов в ГРВС, а также прогнозирования развития ее состояния;
- модели, алгоритмы, методы и программные средства поддержки принятия решений в процессе управления потоком заданий в ГРВС на основе многокритериального выбора управляющих воздействий;
- мультиагентные модели, алгоритмы, методы и программные средства управления вычислительными процессами решения научных и прикладных исследовательских задач, включая планирование вычислений и распределение ресурсов ГРВС;
- инструментальный комплекс DISCENT для организации вычислений в Грид;
- инструментальные комплексы Orlando Tools и DISCOMP для создания и применения РППП;

– программно-аппаратные средства (ПК, серверов, кластеров, облачных и грид-инфраструктур, а также системное ПО для их интеграции и управления ими), на которые ориентируются разрабатываемые РППП.

Вышеперечисленные компоненты объединяются в единую технологическую схему решения научных и прикладных исследовательских задач с заданными критериями качества их обслуживания (временем, стоимостью и надежностью процесса решения задачи). Объединение компонентов осуществляется на основе АМ ГРВС, которая обеспечивает взаимосвязанное представление знаний о среде.

Использование единой модели ГРВС в инструментальных комплексах для разработки РППП позволяет обеспечить комплексирование по данным для всех пакетов, разрабатываемых в рамках данной среды. Тем самым обеспечивается возможность использования в процессе создания нового пакета фрагментов описания предметных областей, функциональных модулей, исходных данных и результатов вычислений, имеющихся в других пакетах. Вследствие этого сокращаются сроки разработки прикладного ПО и проведения вычислительных экспериментов.

На рисунке 5.21 приведена схема интеграции вычислительных систем в рамках проблемно-ориентированной среды для решения трудных переборных задач на основе многовариантных расчетов. Интеграция осуществляется с помощью рассмотренных выше инструментальных комплексов организации вычислений в Грид и разработки РППП.

В качестве вычислительных систем могут быть использованы:

– различные ПК, серверы и базы данных;
– кластеры ПК и HPC-кластеры;
– узлы Linux (отдельные узлы с операционной системой Linux), которые могут использоваться для включения в РППП неотторжимого (расположенного в специализированных узлах) ПО;
– виртуальный кластер создается на основе выделенных ресурсов ЦКП с размещенными на них ВМ проблемно-ориентированной ГРВС, образы

которых включают необходимое прикладное ПО пакетов и средства PBS Torque для управления очередями заданий этих пакетов;

– кластер облачных ресурсов организуется с помощью ВМ проблемно-ориентированной ГРВС, которые развертываются в облачной инфраструктуре, предоставленной ее провайдером.

Рисунок 5.21 – Схема интеграции вычислительных систем

Архитектура виртуального кластера приведена на рисунке 5.22.

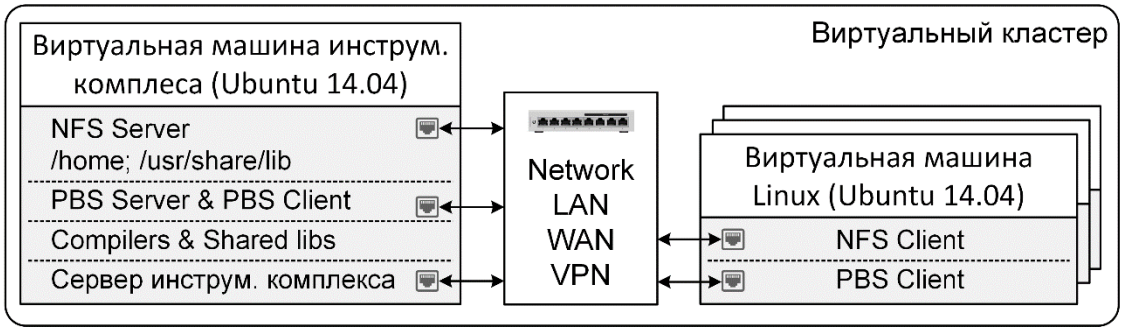


Рисунок 5.22 – Архитектура виртуального кластера

Системная часть является общей для всех РППП, разработанных в рамках конкретной ГРВС. Она представлена сервером ГРВС, поддерживающим веб-интерфейс для доступа пользователей к ее компонентам. Основу системной части составляют конструктор концептуальной модели, планировщики вычислений и интерпретаторы схем решения задач.

Библиотеки модулей РППП включают прикладное ПО для решения задач предметной области.

База знаний содержит следующие компоненты знаний концептуальной модели: информацию о прикладных программных модулях для решения задач предметной области, а также препроцессорной и постпроцессорной обработки данных; структурные свойства алгоритмов решения задач в терминах параметров и операций предметной области; условия выбора оптимальных алгоритмов решения задачи в зависимости от конфигурации ресурсов среды и их состояния; сведения о программно-аппаратной инфраструктуре (характеристики узлов, каналов связи, сетевых устройств и топологии сети, а также данные о сбоях ПО и аппаратных средств); данные о системах управления распределенными вычислениями, используемых в среде; параметры административных политик, применяемых в отношении ресурсов и пользователей.

Application Programming Interface (API) для поддержки грид- и веб-сервисов обеспечивает подключение разнообразных удаленных информационно-вычислительных ресурсов.

Расчетная база данных хранит информацию, необходимую для проведения экспериментов и результаты решения задач.

Для решения задачи пользователь должен сформулировать постановку задачи на концептуальной модели. На основе постановки задачи планировщик вычислений автоматически строит схему ее решения (абстрактную программу) и формирует задание на выполнение процесса решения в среде. Задание содержит информацию о требуемых ресурсах, исполняемых модулях, входных и выходных данных, а также другие необходимые сведения.

Определение оптимальной конфигурации среды для конкретных

экспериментов основывается на знаниях о специфике решаемой задачи, а также вычислительных характеристиках и правилах использования ресурсов системы. Эти знания отражаются в концептуальной модели.

Агенты представляют вычислительные системы, входящие в ГРВС, и управляют распределением потоков заданий РППП с учетом административных политик, принятых в каждой системе, а также классов заданий, генерируемых этими пакетами.

В зависимости от функциональных и системных требований, предъявляемых к РППП их разработчиками, данная технология может быть реализована как с помощью Orlando Tools, так и DISCOMP. Доступ к удаленным ресурсам Грид организуется с помощью сервис-ориентированных средств, являющихся развитием системного ПО инструментального комплекса DISCENT. Основные этапы создания, развития и применения ГРВС с помощью рассмотренных в данной главе инструментальных средств представлены на рисунке 5.23. Данная среда интегрировала различные программно-аппаратные средства и поддерживала разнообразные технологии распределенных вычислений.

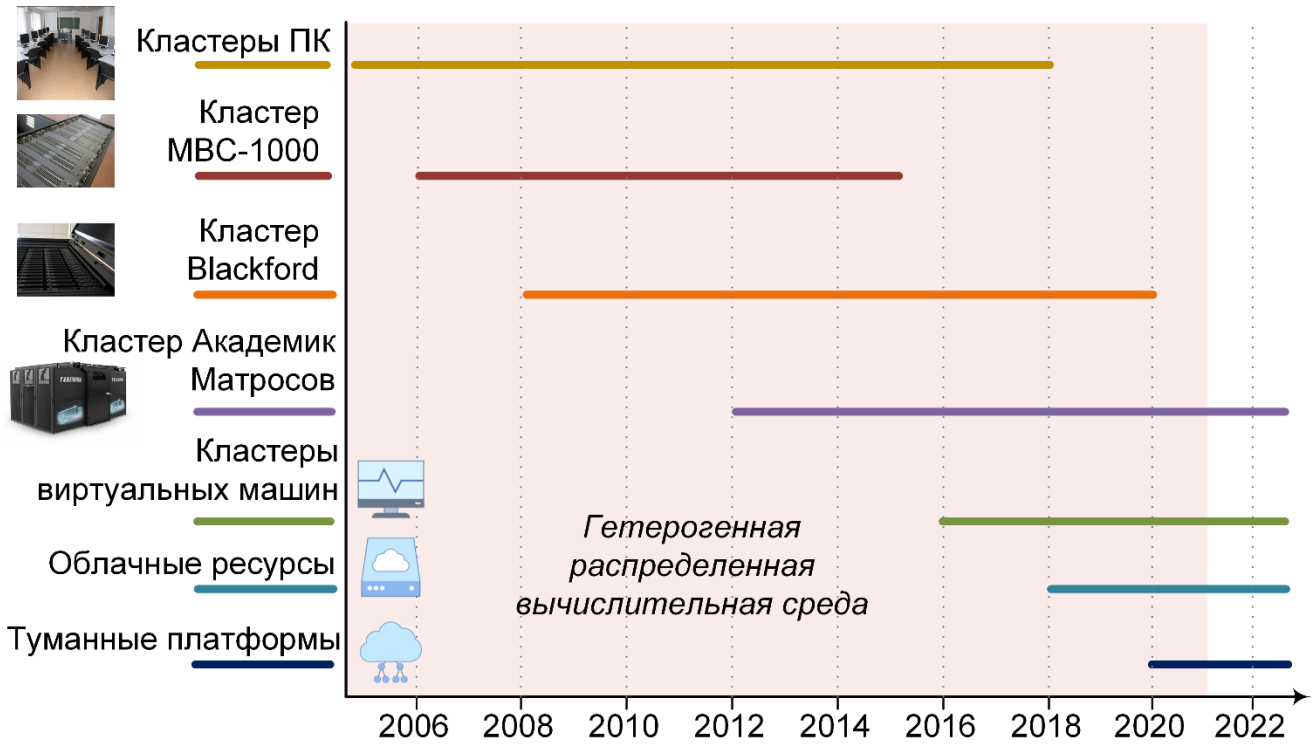


Рисунок 5.23 – Основные этапы создания, развития и применения ГРВС

5.5. Выводы

В пятой главе предложен подход к разработке РППП для решения крупномасштабных задач в ГРВС, которые могут включать вычислительные системы различного уровня (персональные компьютеры, серверы, кластеры, грид-системы и облака) и обеспечивать их интегрированное использование.

Получены следующие основные результаты:

- инструментальный комплекс DISCENT, предназначенный для организации вычислений в Грид, адаптирован к мультиагентному управлению потоками заданий приложений на основе их классификации и обеспечению сервис-ориентированного доступа к компонентам этих приложений из удаленных прикладных программ;
- выполнен анализ технологических аспектов создания распределенных пакетов прикладных программ в ГРВС, показавший актуальность интеграции моделей облачных и грид-вычислений;
- осуществлено развитие инструментальных комплексов Orlando Tools и DISCOMP, применяемых для разработки и применения распределенных пакетов прикладных программ, с целью создания в них инструментов для построения масштабируемых научных приложений в ГРВС с виртуализированными ресурсами, а также интеграции облачных и грид-вычислений;
- разработана технология построения предметно-ориентированных ГРВС, которая, в отличие от известных технологий организации распределенных вычислений, обеспечивает гибкость при выборе и настройке необходимой конфигурации вычислительной инфраструктуры в процессе разработки и применения масштабируемых распределенных пакетов прикладных программ, а также возможность подготовки и проведения экспериментов различного масштаба с использованием гибридной модели, интегрирующей модели облачных и грид-вычислений;

– получены справки об использовании программ для ЭВМ [323, 329–331, 344, 376, 385, 386] в рамках НИР, выполняемых в МИЭЛ ИГУ (см. Приложение А).

Результаты исследований, представленные в данной главе, опубликованы в [63, 64, 66, 68, 69, 71, 74, 79, 87, 325, 334, 340, 345, 347, 350, 357, 365, 368, 373, 379, 382, 387, 389, 392].

Глава 6. Применение результатов диссертационного исследования при разработке РППП

Шестая глава посвящена применению результатов диссертационного исследования при разработке РППП. Приводятся научные и прикладные задачи, которые решены на основе РППП, разработанных с помощью инструментальных комплексов DISCOMP и Orlando Tools, а также в экспериментальной Грид, созданной на базе инструментального комплекса DISCENT. Более детально рассматриваются пакеты для выявления критических элементов отраслевых систем энергетики и решения задач складской логистики.

6.1. Практическое применение инструментальных комплексов DISCOMP и Orlando Tools

Инструментальный комплекс DISCOMP успешно использован в процессе создания и применения РППП для решения в ГРВС следующих ресурсоемких научных и прикладных задач конечными пользователями среды:

- генерализация (приведение к общему виду) алгоритма CARMA [124], используемого для нахождения белковых доменов в массиве метагеномных чтений и их таксономической классификации на основе молекулярно-филогенетического анализа [298];
- рационального дизайна лекарственных средств против клещевого энцефалита и других вирусных инфекций [166, 167]:
- анализа генной экспрессии в близкородственных популяциях байкальского омуля и озерного сига [236];
- определение нуклеотидной последовательности полного генома бесшовной пеннатной диатомеи Synedra acus subsp. radians из озера Байкал [92];
- исследования направлений развития топливно-энергетического комплекса Вьетнама с позиции его энергетической безопасности [55].

Инструментальный комплекс Orlando Tools успешно использован в процессе создания и применения РППП для решения в ГРВС следующих ресурсоемких

научных и прикладных задач:

- оптимизация складской логистики с использованием согласованного семейства аналитических и имитационных моделей процессов функционирования склада [234];
- комбинаторного анализа функционирования топливно-энергетического комплекса Вьетнама [259] с учетом минимизации затрат на его эксплуатацию и развитие [56];
- оптимизация многоэкстремальных функций с помощью метода мультистарта [350].

Результаты экспериментального анализа ускорения вычислительного процесса и эффективности использования ресурсов в [55, 56, 92, 166, 167, 234, 236, 298, 350] показали масштабируемость вычислений при увеличении числа разнородных ресурсов, используемых в процессе решения задач, для всех разработанных РППП.

Кроме того, в рамках экспериментальной Грид решены следующие практические задачи: анализ надежности телекоммуникационных и вычислительных систем различного назначения [257, 279, 347]; оптимизация складской логистики [232, 235]; исследования качества функционирования предприятий технической поддержки [304] и торговли [233]).

Данная Грид использована в качестве экспериментального полигона для исследования свойств параллельных и распределенных вычислений, отработки алгоритмов решения научных и прикладных задач, отладки систем управления заданиями. Она применена в учебном процессе студентами Иркутского государственного университета и аспирантами ИДСТУ СО РАН для приобретения практических навыков параллельных и распределенных вычислений [387].

6.2. Выявление критических элементов отраслевых систем энергетики

6.2.1. Предметная область

Как правило, критическая инфраструктура определяется как совокупность взаимосвязанных физических или виртуальных объектов, нарушение

функционирования которых может иметь негативный эффект на оборону, экономику, здравоохранение и безопасность страны (см., например, [275]). При этом выделяются следующие взаимосвязи между объектами: физические, кибернетические, географические (пространственные), логические, управленческо-организационные [174]. Сами объекты, как правило, описываются технологическими и экономическими характеристиками процессов преобразования ресурсов, осуществляемых с их помощью.

Энергетическая инфраструктура включает большое число объектов, связанных с добычей, хранением, переработкой, преобразованием и транспортировкой энергоресурсов, а также конечных потребителей энергии и относится к ключевым критическим инфраструктурам [285]. Недопоставка энергоресурсов, вызванная нарушением работы объектов энергетической инфраструктуры, оказывает существенное негативное влияние на все социально-экономические сферы человеческой деятельности [217].

К основным типам возмущений, вызывающих крупные недопоставки энергоресурсов, относятся техногенные катастрофы, стихийные бедствия, террористические акты и т.п. Каждый тип возмущений характеризуется уникальной совокупностью последствий, которые могут возникнуть в результате воздействия данного типа возмущений.

Таким образом, необходимо проведение исследования живучести энергетических инфраструктур с учетом большого числа комбинаций их параметров (состояний, типов возмущений, мероприятий по повышению живучести и др.). Чем больше комбинаций возмущений и их последствий рассматривается, тем выше становится степень качества результатов исследования живучести энергетических инфраструктур. Комбинаторный характер такого анализа приводит к необходимости использования высокопроизводительных вычислений.

Анализ известных инструментов моделирования функционирования и развития энергетических инфраструктур на основе парадигм параллельных и распределенных вычислений показывает [1, 62, 223], что они позволяют изучать

только некоторые аспекты вышеупомянутых проблем [163]. Кроме того, от параллельных вычислительных систем с общей памятью к распределенным средам ускорение вычислений и эффективность использования ресурсов существенно снижаются [109]. В связи с этим возникает проблема разработки и применения предметно-ориентированной распределенной вычислительной среды для исследования живучести энергетических инфраструктур.

В зависимости от размерности параметров используемых моделей, данные задачи могут быть решены с помощью различных вычислительных систем (персональный компьютер, сервер, кластер, грид-система, облако) за относительно приемлемое время (часы, сутки), продолжительность которого обуславливается характеристиками используемых вычислительных ресурсов. Интеграция перечисленных систем в единую среду обеспечивает гибкость при выборе необходимой конфигурации вычислительной инфраструктуры, а также быстроту подготовки и проведения экспериментов различного масштаба.

6.2.2. Пакет

Общая схема подсистемы транспорта природного газа представлена на рисунке 6.1. Транспортировка газа осуществляется с помощью магистральных трубопроводов, требуемое давление в которых поддерживается узловыми компрессорными станциями. В газотранспортной сети дополнительно предусмотрены подземные хранилища газа, предназначенные для компенсации сезонных колебаний объемов его потребления [258].

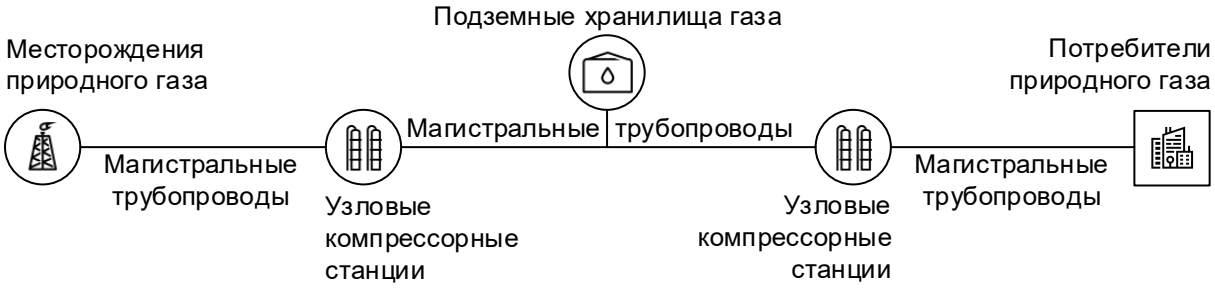


Рисунок 6.1 – Схема транспортировки газа

Основные месторождения природного газа, обеспечивающие более 90% от

общего объема его добычи в России, сосредоточены на севере Западной Сибири. В некоторых случаях расстояние до конечных потребителей превышает 2.5 тыс. км. Это обуславливает важную роль подсистемы транспорта в системе газоснабжения России [258].

Критическими элементами газотранспортной сети являются те ее объекты, нарушение штатного функционирования которых приводит к возникновению существенной недопоставки газа [258, 264]. Сложность выявления таких критических элементов значительно возрастает при необходимости учета одновременных отказов нескольких элементов, так как их синергетические последствия могут изменять степень критичности отказавших элементов.

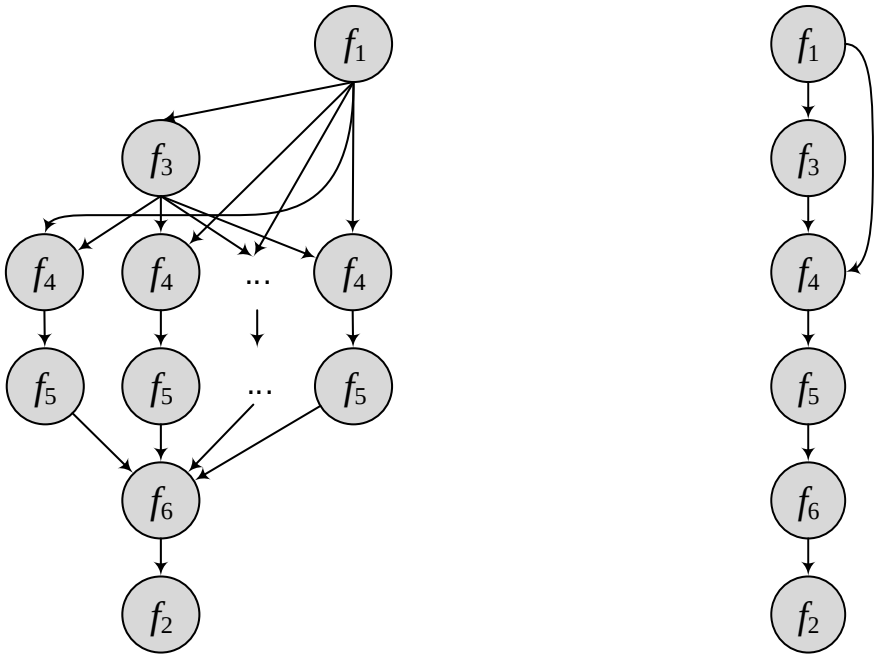
Первая версия РППП для решения данной задачи в ГРВС была разработана с помощью инструментального комплекса DISCOMP [258, 375]. Текущая версия пакета реализована в Orlando Tools [55, 67]. Схема интеграции вычислительных систем приведена на рисунке 6.2.

Схема решения задачи включает шесть операций. Операции f_1 и f_2 моделируют множества исходных и целевых параметров задачи. Операция f_3 строит модель газотранспортной сети (параметр z_4), ограничения которой схожи с (34)-(36), исходя из ее общей схемы (параметр z_1), множеств возможных чрезвычайных ситуаций (параметр z_2) и показателей штатного функционирования сети (параметр z_3). Параллельная операция f_4 формирует параметр z_5, агрегирующий варианты множеств критических элементов сети с отказами. Параллельная операция f_5 производит оценку допустимости вариантов множеств элементов с отказами, представленных элементами $z_{51}, z_{52}, \dots, z_{5n_v}$ параметра-параллельного списка z_5. Варианты множеств элементов с отказами представлены элементами $z_{61}, z_{62}, \dots, z_{6n_v}$ параметра-параллельного списка z_6, n_v – число вариантов. Операция f_6 выполняет анализ списка критических элементов (параметр z_6) с учетом недопоставки газа при их одновременных отказах. Список критических элементов (параметр z_7), упорядоченный в порядке убывания недопоставки газа, передается экспертам с целью формирования ими мероприятий, направленных на

повышение живучести системы газоснабжения России.

Рисунок 6.2 – Схема интеграции вычислительных систем

Модель схемы решения задачи в виде рабочего процесса представлена на рисунке 6.3 а. Использование параллельных операций f_4 и f_5 в Orlando Tools позволяет существенно упростить вид схемы решения задачи, используемый при распределении ресурсов (рисунок 6.3 б) агентами. Они применяют долевое распределение вычислительной нагрузки, обусловленной выполнением операций f_4 и f_5. Поэтому им не требуется рассматривать распределение ресурсов для каждого экземпляра этих операций индивидуально.



а) б)

Рисунок 6.3 – Модель схемы решения задачи в виде рабочего процесса без использования (а) и с использованием (б) долевого распределения вычислительной нагрузки, обусловленной выполнением операций f_4 и f_5

Модули m_1, m_2, m_3 и m_4 пакета, реализующие операции f_3, f_4, f_5 и f_6, разработаны в виде Windows-приложений. Модули m_1 и m_2 выполняются на невыделенных ресурсах (ПК). Модули m_2 и m_3 выполняется на выделенных виртуализированных ресурсах под управлением ОС GNU/Linux. Образы ВМ включают ОС ReactOS [172], а также модули m_2 и m_3. Данная ОС обеспечивает совместимость с Windows-приложениями.

В Приложении К приведено описание предметной области РППП в инструментальном комплексе Orlando Tools на языке XML. В приведенных спецификациях использованы содержательные имена объектов предметной области. Такое описание может быть также создано с помощью веб-интерфейса инструментального комплекса Orlando Tools (см. Приложение И, рисунки И.10-И.15) и затем автоматически конвертировано на XML.

6.2.3. Пример

Рассматриваемый в задаче граф системы газоснабжения России состоит из

378 узлов. В их числе 28 источников природного газа, 64 потребителя (регионы Российской Федерации), 24 подземных хранилища газа, 266 магистральных компрессорных станций. Граф также содержит 486 дуг, представляющих сегменты магистральных трубопроводов [258].

415 дуг отобраны как элементы с возможными отказами. Для определения критических элементов газотранспортной сети России применяется методика, представленная в [258]. Число n_k комбинаций элементов с отказами может быть оценено следующей формулой:

$$n_k = \frac{n_e!}{(n_e - n_f)!\, n_f!},$$

где n_f – число одновременно отказавших элементов, n_e – число дуг, выбранных в качестве элементов с возможными отказами. В процессе проведения экспериментов число n_f варьировалось от 2 до 3.

ГРВС, используемая в процессе решения задачи, включает выделенные кластерные ресурсы ЦКП ИСКЦ (приватное облако под управлением платформы OpenStack) и невыделенные ресурсы под управлением PBS Torque. ВМ управляются гипервизором KVM. Эксперименты проводились с использованием трех пулов ресурсов:

1) 10 узлов (невыделенные ресурсы) со следующими характеристиками – 2 процессора Intel Xeon 5345 EM64T (4 ядра, 2.3 Ггц, 8 Гбайт RAM);
2) 10 узлов (выделенные ресурсы) со следующими характеристиками: 2 процессора AMD Opteron 6276 (16 ядер, 2.3 Ггц, 64 Гбайт RAM);
3) 10 узлов (выделенные и невыделенные ресурсы) со следующими характеристиками: 2 процессора Intel Xeon CPU X5670 (18 ядер, 2.1 Ггц, 128 Гбайт RAM).

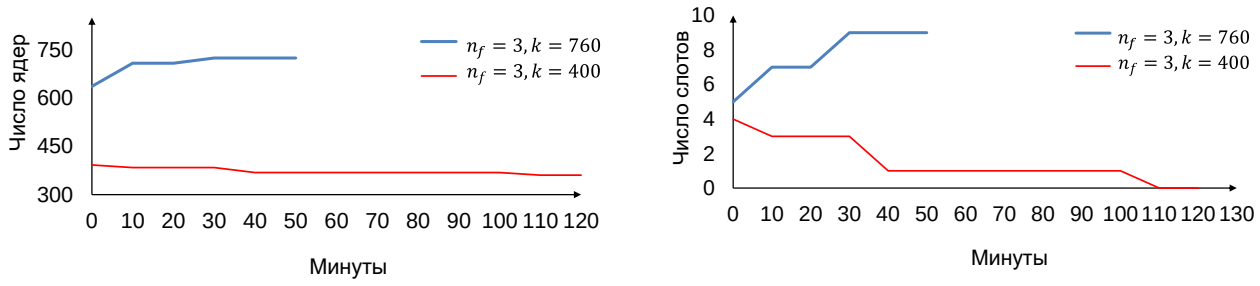
Узлы разных пулов имеют разные типы интерконнекта (1 GigE, QDR Infiniband) и жестких дисков (HDD, SSD). Все выделенные ресурсы виртуализированы. Таблица 25 показывает время $t_{k=1}$, $t_{k=400}$ и $t_{k=760}$ решения задачи для $n \in \{2,3,4\}$, где k – максимальное число доступных ядер. $t_{k=1}$, $t_{k=400}$ и

$t_{k=760}$ получены соответственно на ПК с процессором Intel Core i5-3450 (4 core, 3.10 GHz, 4 GB of RAM), в пулах 1-2 и в пулах 1-3.

Таблица 25 – Время решения задачи

n_f	n_k	$t_{k=1}$	$t_{k=400}$	$t_{k=760}$
2	85905	16	67	70
3	11826255	656640	7200	2730
4	1218104265	31536000	1555200	622080

Значения $t_{k=1}$, $t_{k=400}$ и $t_{k=760}$ для $n_f = 2$ и $n_f = 3$ получены путем реальных экспериментов с использованием Orlando Tools. Значения $t_{k=1}$, $t_{k=400}$ и $t_{k=760}$ для $n_f = 4$ получены путем оценки времени решения задач в пулах 1-3. Очевидно, что увеличение n_f влияет на быстрый рост n_k, тем самым увеличивая время решения задач во много раз. Вычисления на ПК целесообразны только при $n_f = 2$. В этом случае время решения задач в пулах сопоставимо с временем их решения на ПК из-за наличия накладных расходов, связанных с передачей данных между узлами пулов. При $n_f = 3$ время решения задач в пулах уже значительно меньше времени, показанного на ПК. Следует отметить, что при $n_f = 4$ решение задач на ПК за приемлемое время практически невозможно. Рисунки 6.4 а и 6.4 б показывают число использованных ядер и слотов в процессе решения задач при $n_f = 3, k = 400$ и $k = 760$. В целом декомпозиция потоков заданий обеспечила значительное сокращение времени решения задачи более чем на 15%.



а) б)

Рисунок 6.4 – Число использованных ядер (а) и слотов (б)

Рисунок 6.5 демонстрирует высокую среднюю загрузку процессора. Она немного меньше в пулах 1 и 3 из-за накладных расходов на виртуализацию. В то же время накладные расходы на виртуализацию в узлах всех пулов не превышают 5%. Рисунок 6.6 показывает улучшение (сокращение) времени решения задачи при $n_f = 3$ как для $k = 400$, так и для $k = 760$.

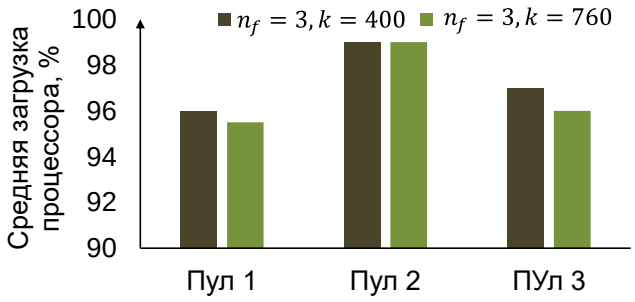


Рисунок 6.5 – Средняя загрузка процессора

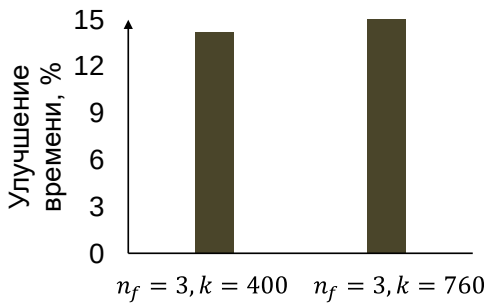


Рисунок 6.6 – Улучшение времени решения задачи

Для обеспечения выполнения задания с модулями m_2 и m_3 создан класс c_9. Он включает характеристики для заданий, которые могут выполняться как на выделенных, так и невыделенных ресурсах. Информация об этих характеристиках представлена в таблицах 26 и 27. Данная информация отражает следующие данные: список характеристик заданий; области их допустимых значений в системе классификации; единицы измерения значений характеристик; атрибуты характеристик «обязательная» или «необязательная»; специализированные области значений характеристик для класса c_9; примеры значений характеристик в спецификации задания.

В этой спецификации признаки 1-3 представляют информацию о выполняемых файлах. Характеристики 4-6 описывают запрашиваемые вычислительные ресурсы. Выбор правильного образа ВМ обеспечивается характеристиками 7 и 8. Orlando Tools указан в качестве связующего ПО, используемого в ВМ. Характеристики 9 и 10 определяют ограничения интервалов в расписании СУПЗ. Следует отметить, что характеристики 11-13 являются

необязательными, поэтому их значения могут быть не определены в спецификации задания. Применение класса c_9 позволило существенно повысить качество выполнения модулей m_2 и m_3 по сравнению с их выполнением без применения классификации заданий.

Таблица 26 – Характеристики класса c_9

Характеристика	Тип данных	Область допустимых значений	Единица измерения
1. Имя задания	String	1-255	Символ
2. Путь к исполняемому файлу	String	1-255	Символ
3. Параметры исполняемого файла	String	1-65535	Символ
4. Требуемое число узлов	Integer	1-20	Узел
5. Требуемое число ядер в узле	Integer	1-36	Ядро
6. Число ВМ	Integer	1-36	ВМ
7. Требуемая ОС	String	1-256	Символ
8. Требуемое связующее ПО в ВМ	String	1-256	Символ
9. Требуемое число свободных ядер в слотах невыделенных ресурсов	Integer	1-36	Символ
10. Требуемый минимальный интервал времени существования слота в расписании СУПЗ невыделенных ресурсов	Integer	1-1728000	Секунда
11.Максимальное время выполнения задания	Integer	1-1728000	Секунда
12. Требуемый размер оперативной памяти	Integer	8-128	Гбайт
13. Требуемый размер дискового пространства	Integer	1-2048	Гбайт

Таблица 27 – Дополнительные атрибуты характеристик класса c_9

Характеристика	Обязательная / необязательная	Специализированная область допустимых значений	Значение характеристики в спецификации
1. Имя задания	Обязательная	1-255	Corrective_solver
2. Путь к исполняемому файлу	Обязательная	1-255	~/orlando/Corrective/failsets
3. Параметры исполняемого файла	Обязательная	1-65535	-input_file input.xml -output_file output.xml
4. Требуемое число узлов	Обязательная	1	1
5. Требуемое число ядер в узле	Обязательная	32	32
6. Число ВМ	Обязательная	32	32
7. Требуемая ОС	Обязательная	ОС ReactOS	ОС ReactOS
8. Требуемое связующее ПО в ВМ	Обязательная	ORLANDO	ORLANDO
9. Требуемое число свободных ядер в слотах	Обязательная	32	32
10. Требуемый минимальный интервал времени существования слота	Обязательная	900-1728000	900
11.Максимальное время выполнения задания	Необязательная	1-1728000	–
12. Требуемый размер оперативной памяти	Необязательная	8	–
13. Требуемый размер дискового пространства	Необязательная	1	–

Решение рассмотренной выше задачи позволило экспертам из ИСЭМ СО РАН составить список критически важных объектов газотранспортной сети России и сформулировать некоторые мероприятия по повышению живучести системы газоснабжения России [258]. В частности, были предложены рекомендации по технологическому доустройству отдельных сегментов магистральных трубопроводов с целью обеспечения режима их работы, позволяющего повысить длительность кратковременного увеличения их пропускной способности в случае необходимости компенсации последствий крупных возмущений на других участках сети. Для поддержки процесса принятия решений экспертами в ГРВС организовано взаимодействие с ИАС [76], рассмотренной в пятой главе, с целью отображения результатов планируемого доустройста на электронных картах.

6.3. Пакет для решения задач складской логистики

Современный склад представляет собой сложно организованную функциональную структуру, предназначенную для обработки грузопотоков большой емкости и их распределения между потребителями. В связи с этим он имеет существенное влияние на экономические процессы в различных сферах человеческой деятельности.

Склад, как правило, является частью многих логистических цепочек, связующим звеном между снабжением, производством и сбытом. Поэтому в общем процессе движения материальных потоков от производителя к конечному потребителю необходимо учитывать наличие сети различных складов хранения и переработки продукции, трансформирующие материальные потоки.

Материальные потоки входят в склад с определенными значениями своих параметров, которые изменяются под воздействием складских процессов. К числу таких параметров относятся мощность и интенсивность потоков, их структура, вид и способ упаковки товаров, время прибытия и отправления транспортных средств, требования по грузопереработке и хранению товаров, а также другие важные характеристики. Складские процессы, воздействующие на материальные потоки и преобразующие их параметры, характеризуются различной степенью сложности их

организации и выполнения.

Недооценка роли и значения склада в управлении материальными потоками негативно влияет на организацию и показатели функционирования системы производства, распределения, физического перемещения и потребления продукции. Таким образом, качественная работа любой логистической системы зависит не только от промышленного и транспортного производства, но и от склада.

Тем самым актуализируется логистическая составляющая в управлении складом, определяющая основные технические требования к складскому комплексу, его задачи и функции как звена логистической цепи, принципы его оптимальной работы и требования к процессам обработки товаров. Склад, организованный таким образом, является логистическим складским комплексом (ЛСК), а анализ и оптимизация его функционально-организационной структуры становятся важнейшими задачами логистического складского менеджмента.

Технологии и коммуникационные системы современных ЛСК постоянно совершенствуются. Расширяется спектр и повышается уровень сложности логистических операций, выполняемых на складе. Увеличивается число альтернатив принятия управленческих решений.

Сфера влияния логистических операций существенно шире технологических операций. Выполнение логистических операций нужно рассматривать в их взаимосвязи, на всех уровнях детализации этого процесса, что обеспечивает возможность планирования материальных, финансовых и информационных потоков и их контроль с минимальными затратами.

Одним из востребованных подходов к анализу процессов функционирования ЛСК, как систем массового обслуживания, является имитационное моделирование [312], которое позволяет воспроизводить процессы физического движения во времени и пространстве потоков материальных объектов различного рода и относится к «классическому» моделированию процессов с дискретными событиями [180, 269].

Имитационное моделирование является инструментом, обеспечивающим

разработку модели системы массового обслуживания, ее исследование и оптимизацию. Оно предполагает разработку модели, ее программную реализацию и проведение многовариантных расчетов на основе метода Монте-Карло специалистом предметной области с помощью разработанной программы. В рамках этих экспериментов варьируются значения входных переменных модели исследуемой системы с целью достижения требуемых показателей качества функционирования (наблюдаемых переменных) этой системы и нахождения оптимальных параметров ее сервисов обслуживания.

Сложность и динамичность современных ЛСК требует проведения их детализированного исследования на основе применения интегрированного семейства моделей – концептуальной, аналитической и имитационной. Концептуальная модель служит для описания предметной области исследуемой системы, а аналитическая является вспомогательной и реализует математические соотношения между объектами предметной области исследуемого комплекса. Для специалиста предметной области на практике зачастую возникают значительные технические и методические трудности использования такого семейства моделей [61, 176].

В настоящее время эту проблему практически невозможно решить без использования современной высокопроизводительной вычислительной техники, методов и средств параллельных и распределенных вычислений [91, 195], а также технологий и инструментальных средств разработки, интегрирования и применения моделей ЛСК [19, 24].

В данной главе представлен РППП для решения задач складской логистики и результаты его применения в ООО «Иркутский хладокомбинат». Пакет реализован с использованием инструментальных средств, рассмотренных в пятой главе. Экспериментальный анализ результатов его применения для решения практических задач под управлением МАС показало его масштабируемость в ГРВС.

6.3.1. Логистический складской комплекс

А.М. Гаджинский определяет склад как совокупность зданий, сооружений и разнообразных устройств, предназначенных для приемки, размещения и хранения поступивших на них товаров, подготовки их к потреблению и отпуску потребителю [251]. В процессе своей работы склад может выполнять операции транспортировки товаров, их разгрузки и погрузки, упаковки, складирования, дистрибуции, грузопереработки, сбора транспортной тары, ценообразования и другие действия (рисунок 6.7).

Рисунок 6.7 – Складские операции

Как правило, складские процессы требуют выполнения совокупности операций, выполнение которых зависит от характеристик и свойств материальных, финансовых и информационных потоков (входных, внутренних и выходных). Поток случайных заявок клиентов на обслуживание, несогласованность складских операций и сбои при их выполнении, обусловленные человеческим фактором и

отказами технических устройств, существенно снижают качество работы склада. Эта проблема решается путем внедрения складской логистики, позволяющей согласовать и оптимизировать процессы на складе. В этом случае ЛСК можно определить, как склад, на котором внутрискладской технологический процесс (цепочка операций) планируется и выполняется как одно целое.

Пример информационно-логических взаимосвязей между основными складскими операциями показан на рисунке 6.8.

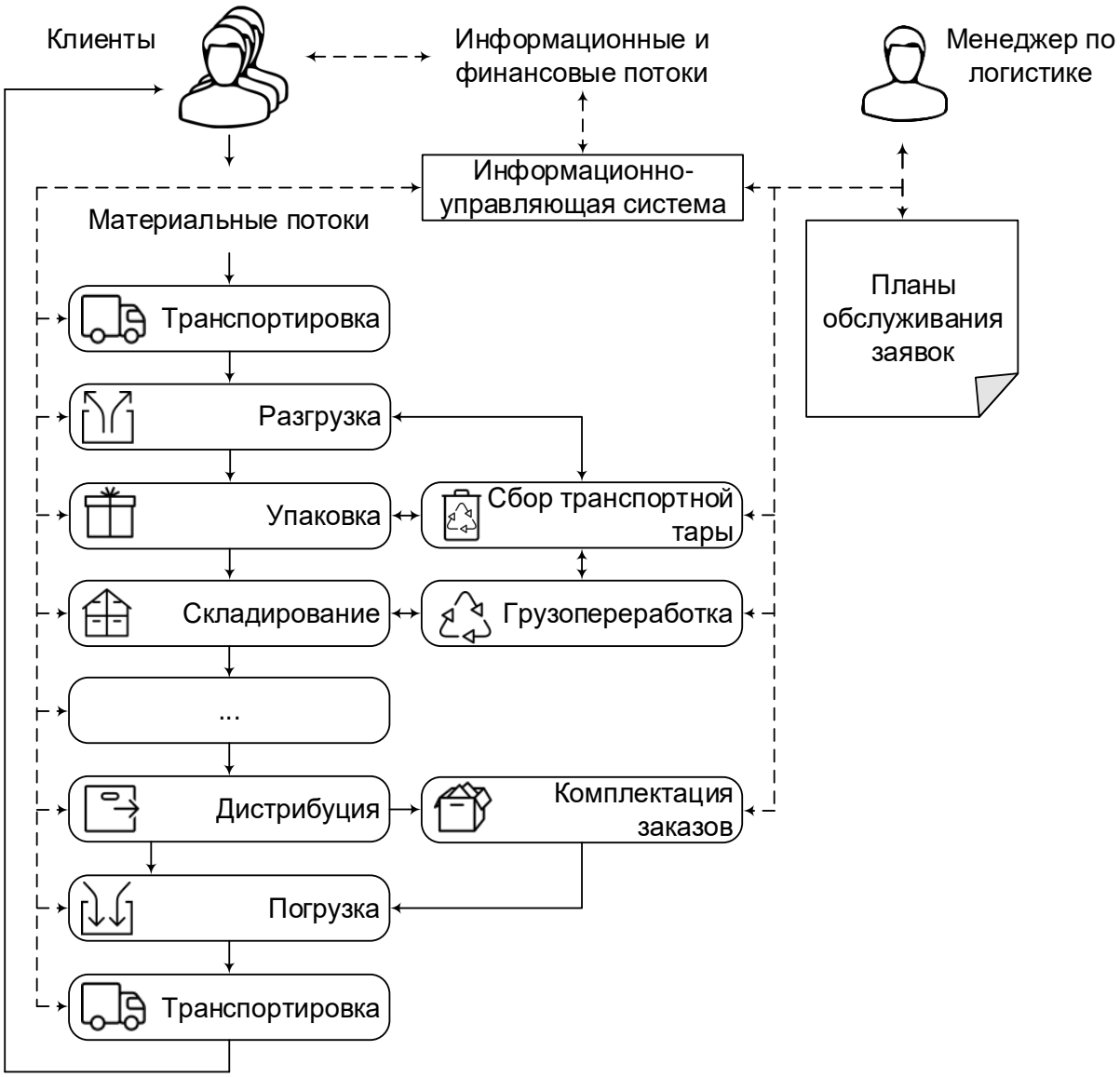


Рисунок 6.8 – Информационно-логические взаимосвязи между операциями

Материальные потоки определяют отношения между операциями. Им

сопутствуют финансовые и информационные потоки. Менеджер по логистике строит планы обслуживания заявок, используя информационно-логические взаимосвязи между операциями. Информационно-управляющая система может базироваться на интегрированном использовании компонентов систем ведения бухгалтерии, управления складом и предприятием в различном их сочетании. Такие системы имеют, как правило, узкую специализацию и не пригодны для моделирования логистических процессов. Они не позволяют исследовать технологические процессы в динамике с учетом всех необходимых деталей их реализации, а также сведений о сопутствующих событиях, явлениях и их взаимосвязях.

Возникает также необходимость поддержки принятия решений при выполнении новых логистических функций, связанных с рациональным планированием складских операций, прогнозированием отказов технических устройств, преобразованием внешних ограничений во внутренние ограничения, количественной оценкой качественных критериев, распределением ресурсов и анализом работы склада. Актуальным средством реализации процесса поддержки принятия решения является имитационное моделирование. Исследование логистических процессов является на сегодняшний день одной из основных областей применения имитационного моделирования процессов с дискретными событиями [315].

6.3.2. Постановка задачи управления процессом функционирования ЛСК

Схема управления ЛСК [78] представлена на рисунке 6.9. На данной схеме ЛСК выступает в качестве объекта управления. Функциональные возможности склада определяются имеющимися объектами $o_1, o_2, \dots, o_{n_o}$ хранения товаров, допустимыми складскими процессами $p_1, p_2, \dots, p_{n_p}$, людскими и техническими ресурсами $r_1, r_2, \dots, r_{n_c}, e_1, e_2, \dots, e_{n_e}$. Параметры $x_1, x_2, \dots, x_{n_x}$ отражают характеристики складских объектов и процессов, а также людских и технических ресурсов. Для каждого параметра x_l заданы его предельные значения x_l^{min} и x_l^{max} такие, что $x_l^{min} \le x_l \le x_l^{max}$ а также условие оптимальности критерия $x_l \rightarrow$

$\min(\max)$, $l = \overline{1, n_x}$.

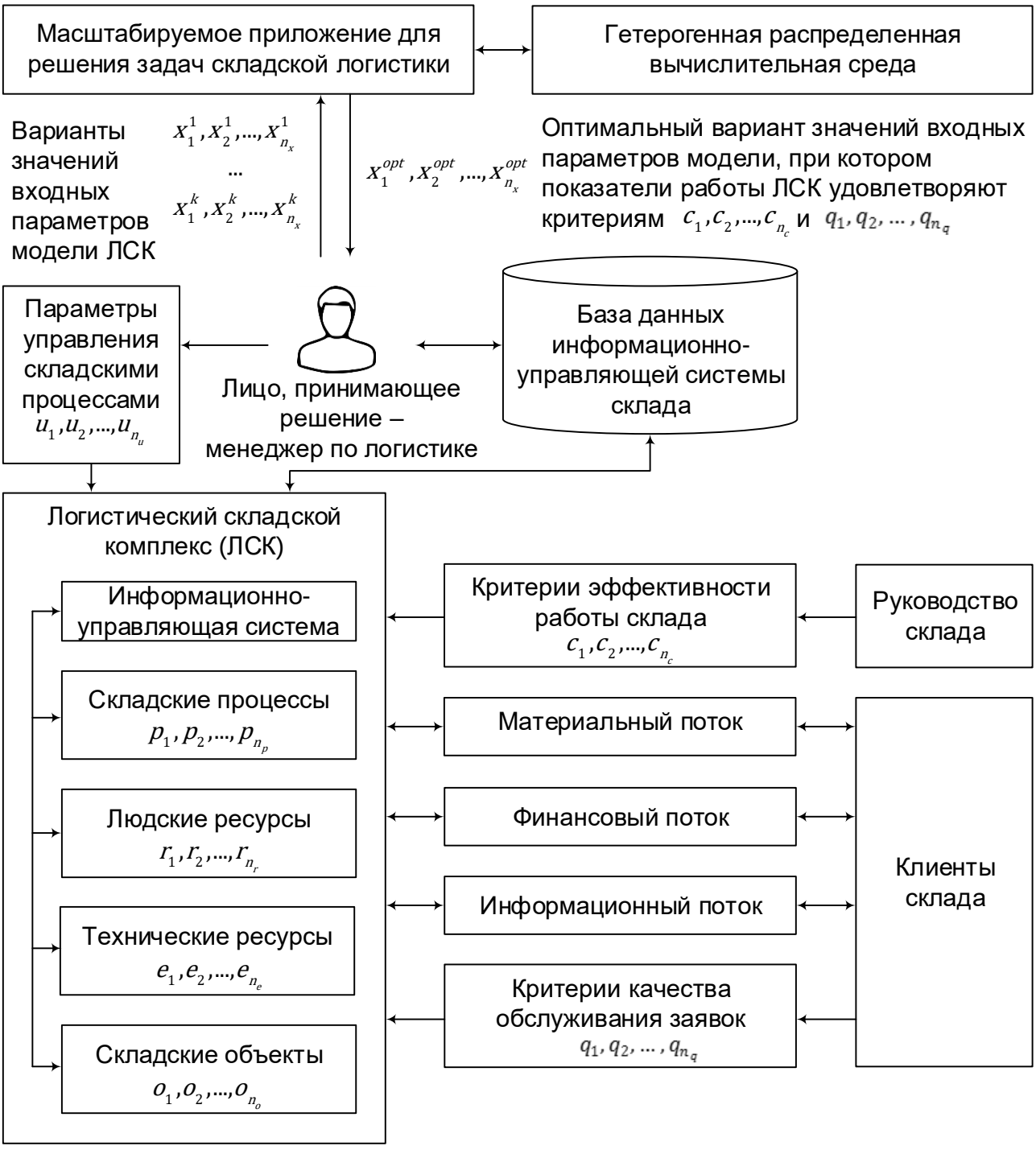


Рисунок 6.9 – Структурная схема управления ЛСК

Материальный поток, а также сопутствующие ему финансовый и информационный потоки являются внешними возмущениями процесса функционирования ЛСК. Материальный поток на складе характеризуется

следующими основными параметрами: номенклатура, ассортимент и качество груза; габаритные размеры; весовые характеристики; физико-химические характеристики груза; характеристики упаковки груза; условия договоров хранения.

Руководство склада определяет критерии $c_1, c_2, \dots, c_{n_c}$ качества его работы. Для каждого $c_i = \phi_i(x_1, x_2, \dots x_{n_x})$ заданы его предельные значения c_i^{min} и c_i^{max} такие, что $c_i^{min} \le c_i \le c_i^{max}$ а также условие оптимальности критерия $c_i \to \min(\max), i = \overline{1, n_c}$. Критерии качества работы склада включают в себя показатели доходов и расходов, рентабельности логистических операций, полезного использования людских и технических ресурсов, грузооборота и наполненности склада, а также другие параметры функционирования склада.

Клиенты задают критерии $q_1, q_2, \dots, q_{n_q}$ качества обслуживания их заявок. Для каждого $q_j = \varphi_j(x_1, x_2, \dots x_{n_x})$ определены его предельные значения q_j^{min} и q_j^{max} такие, что $q_j^{min} \le q_j \le q_j^{max}$ а также условие оптимальности критерия $q_j \to \min(\max), j = \overline{1, n_q}$. Критерии качества обслуживания включают в себя показатели процесса обслуживания очередей заявок и стоимости логистических операций, а также характеристики объектов хранения.

Для различных критериев c_i и q_j абстрактные связи $\phi_i(x_1, x_2, \dots x_{n_x})$ и $\varphi_j(x_1, x_2, \dots x_{n_x})$ могут быть представлены функциональными, статистическими, неоднозначными или иными отображениями. Зачастую критерии качества работы склада и качества обслуживания заявок его клиентов являются взаимно противоречивыми.

Оптимизация складской работы осуществляется лицом, принимающим решения (ЛПР), – менеджером по логистике. В процессе принятия решений он использует РППП для решения задач складской логистики.

Этот пакет включает библиотеку моделей, имитирующих процессы работы ЛСК для разных задач. Он поддерживает многовариантные расчеты в ГРВС путем генерации большого числа независимых заданий при решении задачи. Каждое

задание запускает требуемую модель с единственным вариантом значений ее входных переменных.

Методика проведения экспериментов представлена в [324]. ЛПР варьирует значения входных параметров $x_1, x_2, \dots, x_{n_x}$ выбранной им модели ЛСК и формирует множество их вариантов с помощью средств РППП для формулировки задачи и препроцессорной обработки данных.

Для каждого варианта генерируется свое задание. Задания выполняются в ГРВС параллельно.

Средствами пакета для постпроцессорной обработки данных и анализа результатов моделирования определяется оптимальный вариант значений $x_1^{opt}, x_2^{opt}, \dots, x_{n_x}^{opt}$ входных переменных модели, при котором показатели работы ЛСК удовлетворяют критериям $c_1, c_2, \dots, c_{n_c}$ и $q_1, q_2, \dots, q_{n_q}$.

ЛПР корректирует параметры $u_1, u_2, \dots, u_{n_u}$ управления складскими процессами, используя значения $x_{i_1}^{opt}, x_{i_2}^{opt}, \dots, x_{i_k}^{opt}$ варьируемых параметров из множества $\{x_1, x_2, \dots, x_{n_x}\}$, $1 \le i_j \le n_x$, $k \le n_u$. Корректировка параметров $u_1, u_2, \dots, u_{n_u}$ представляет собой управляющее воздействие ЛПР на процесс функционирования склада. Регулируются следующие параметры: стандарты, затраты и стоимость для логистических операций, структура затрат, категории клиентов, уровни их обслуживания и другие характеристики.

Таким образом, для нахождения оптимального варианта значений $x_1^{opt}, x_2^{opt}, \dots, x_{n_x}^{opt}$ решается следующая задача [75]:

$$x_l \to \min(\max),$$
$$x_l^{min} \le x_l \le x_l^{max},$$

при условии, что

$$c_i = \phi_i(x_1, x_2, \dots x_{n_x}) \to \min(\max),$$
$$q_j = \varphi_j(x_1, x_2, \dots x_{n_x}) \to \min(\max),$$
$$c_i^{min} \le c_i \le c_i^{max},\ q_j^{min} \le q_j \le q_j^{max},$$

где $l = \overline{1, n_x}$, $i = \overline{1, n_c}$, $j = \overline{1, n_q}$.

Данная задача решается с помощью технологии многокритериального выбора, предложенной в главе 2. РППП для решения задач складской логистики разработан с помощью Orlando Tools [66, 74].

6.3.3. Концептуальная модель процессов функционирования ЛСК

В данном разделе представлена концептуальная модель процессов функционирования ЛСК [74]. Она разработана в Orlando Tools на основе вычислительной модели, рассмотренной в разделе 2.1. Концептуальная модель представлена структурой

$$M_c =< P, V, W, Pr >,$$

где P – множество процессов функционирования склада, V – множество параметров процессов, W – множество логистических операций процессов, Pr – множество продукций, определяющих условия выполнения операций.

Элемент $p_i \in P$ описывает схему выполнения технологического процесса обслуживания, которая в общем случае может быть поливариантной: включающей несколько сценариев предоставления услуг.

С каждой операцией $w_i \in W$ связано два подмножества параметров $V_i^{in}, V_i^{out} \subset V$, определяющих соответственно входные и выходные параметры i-й операции.

Технологический процесс обслуживания клиентов определяет совокупность логистических операций, которые нужно непрерывно и ритмично выполнять в соответствии с установленным регламентом проведения складских работ и ограничениями, определяемыми технико-организационными параметрами склада. Он должен обеспечивать экономичность затрат, надежность процесса обслуживания клиентов и сохранность товара.

Оптимизация технологического процесса осуществляется путем повышения рентабельности выполнения операций, а также улучшения использования складских мощностей, людских и технических ресурсов.

На рисунках 6.10, 6.11, 6.12 а и 6.12 б приведены схемы следующих

технологических процессов: разгрузки товаров; отгрузки товаров; технологической оттайки камер; хранения товаров и их инвентаризации. Эти схемы относятся к фрагменту концептуальной модели, описывающей процессы непосредственного обслуживания клиентов.

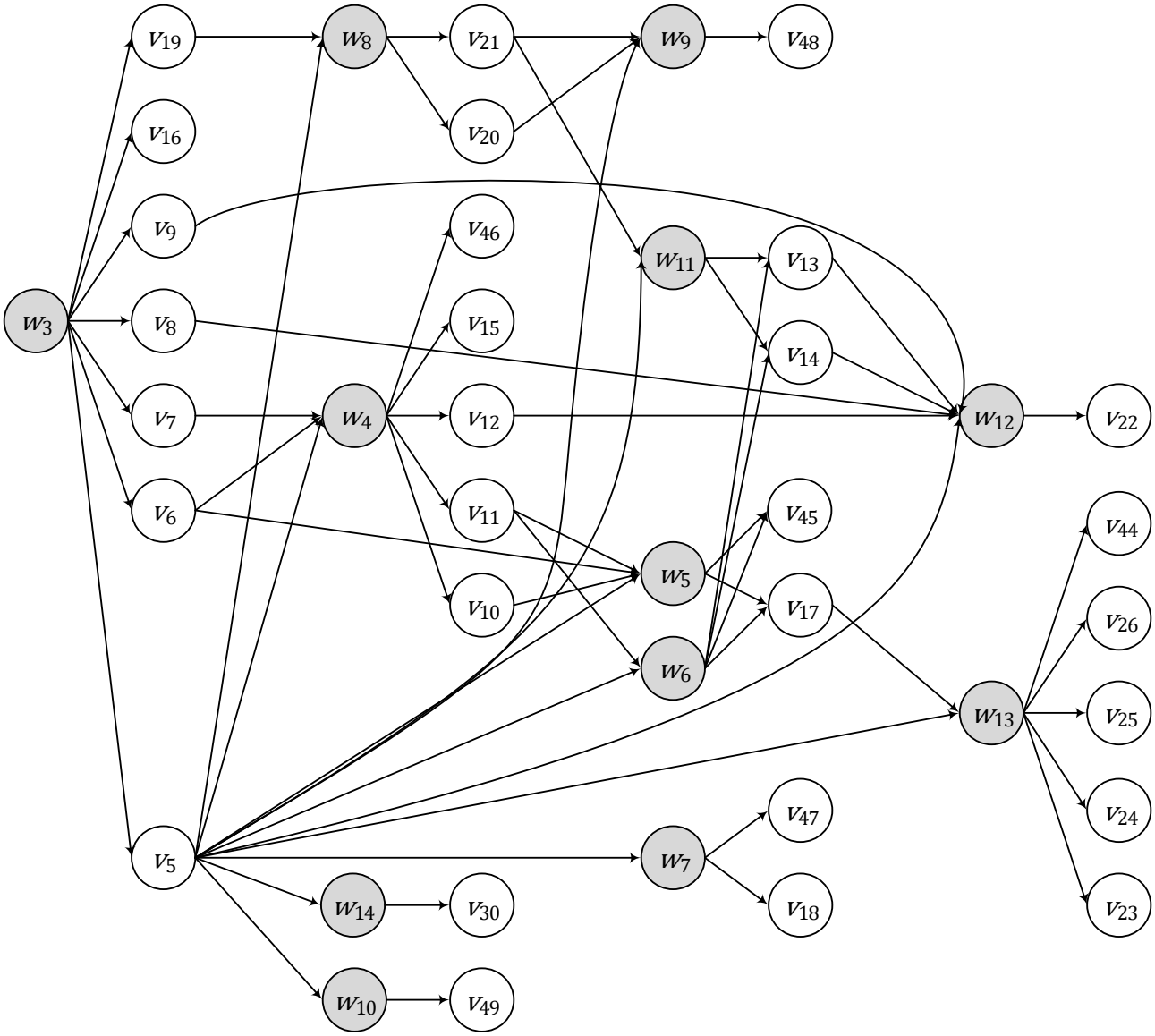


Рисунок 6.10 – Фрагмент концептуальной модели, описывающий процесс разгрузки товаров

Первые два процесса отражают обслуживание заявок в многофазной системе с многоканальными устройствами и очередями. Они включают цепочки последовательно выполняемых операций, в которых входные параметры каждой последующей операции являются выходными параметрами предшествующих

им операций. Остальные процессы представляют обслуживание заявок в однофазной системе с многоканальными устройствами и очередями. В рамках каждого процесса выполняется только одна операция.

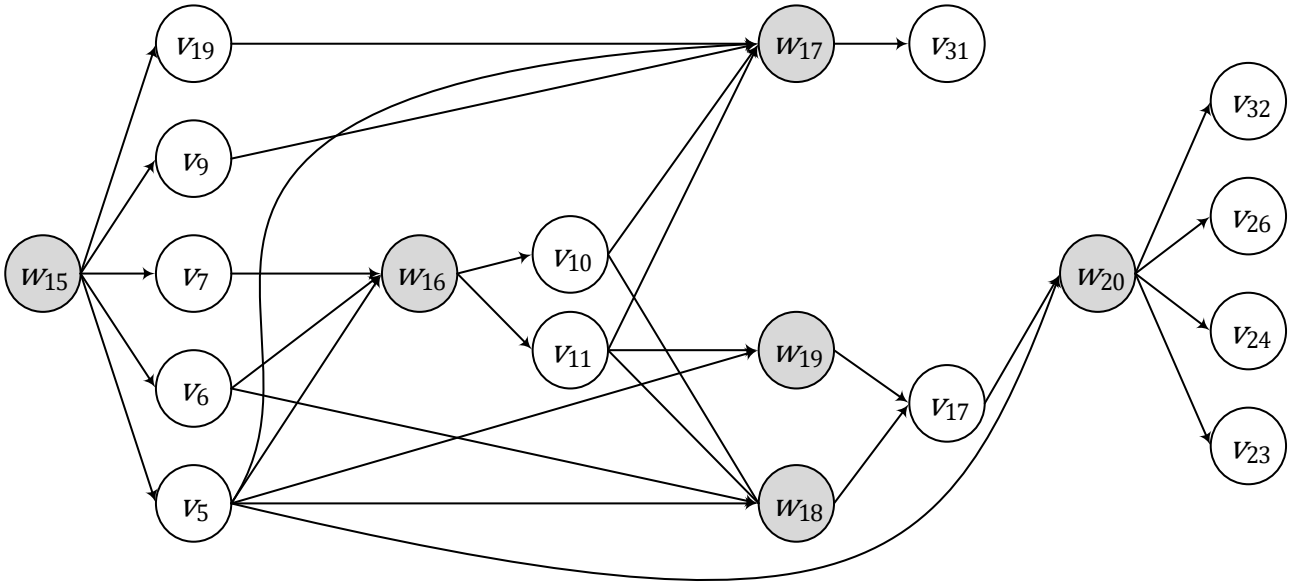
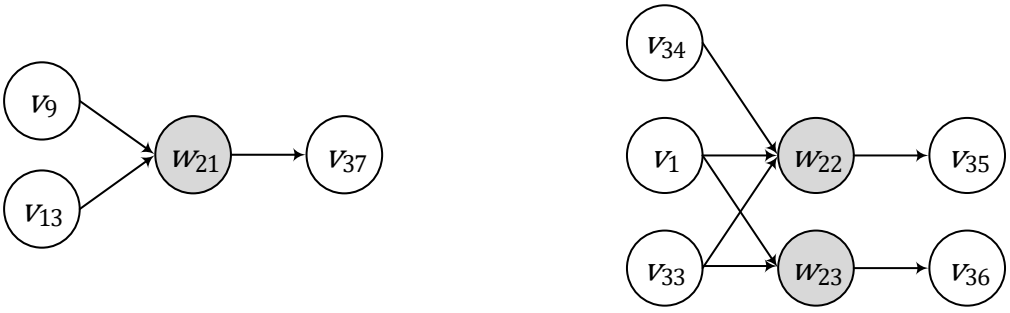


Рисунок 6.11 – Фрагмент концептуальной модели, описывающий процесс отгрузки товаров



а) б)

Рисунок 6.12 – Фрагменты концептуальной модели, описывающие соответственно процессы технологической оттайки камер (а), хранения товаров и их инвентаризации (б)

На рисунках представлены основные параметры перечисленных процессов и операции $w_3 - w_{23}$. Две дополнительные операции w_1 и w_2 используются для моделирования постановки задачи в процессе планирования складских процессов. Параметры v_5, v_{17} и v_{30} являются составными параметрами, агрегирующими

соответственно взаимосвязанные наборы параметров $\{v_1, v_2, v_3, v_4\}$, $\{v_7, v_9, v_{12}, v_{32}\}$ и $\{v_{27}, v_{28}\}$.

В концептуальную модель ЛСК включены логические параметры v_7, v_{15}, v_{16}, v_{18}, v_{22}, v_{31}, v_{38}, v_{39}, v_{40}, v_{41}, v_{42}, v_{44}, v_{45}, v_{46}, v_{47}, v_{48} и v_{49}, принимающие значения 0 или 1. Эти параметры используются для определения условий выполнения операций с помощью продукций вида

$$pr_j\text{: if } l_j \text{ then } w_{i_j},$$

где pr_j – имя j-й продукции, l_j – логическое выражение в левой части j-й продукции, w_{i_j} – i-я операция j-й продукции, которая может быть выполнена, если значение выражения l_j является истинным.

Определен следующий набор продукций, определяющих условия выполнения операций $w_1 - w_{22}$:

$$pr_1\text{: if } v_{38} \text{ then } w_3\ ;$$
$$pr_2\text{: if } v_{38} \text{ then } w_4\ ;$$
$$pr_3\text{: if } \bar{v}_7 v_{38} \text{ then } w_5\ ;$$
$$pr_4\text{: if } v_7 v_{38} \text{ then } w_6\ ;$$
$$pr_5\text{: if } v_{38} v_{45} v_{46} \text{ then } w_7\ ;$$
$$pr_6\text{: if } \bar{v}_7 v_{18} v_{38} v_{45} (\bar{v}_{46} \vee v_{47}) \text{ then } w_8\ ;$$
$$pr_7\text{: if } \bar{v}_7 v_{18} v_{38} v_{45} \text{ then } w_9\ ;$$
$$pr_8\text{: if } v_{15} v_{18} v_{38} v_{45} v_{48} \text{ then } w_{10}\ ;$$
$$pr_9\text{: if } \bar{v}_7 (\bar{v}_{15} \vee v_{49}) v_{18} v_{38} v_{45} \text{ then } w_{11}$$
$$pr_{10}\text{: if } (\bar{v}_{15} \vee v_{49}) v_{18} v_{38} v_{45} \text{ then } w_{12}\ ;$$
$$pr_{11}\text{: if } v_{18} v_{22} v_{38} v_{45} \text{ then } w_{13}\ ;$$
$$pr_{12}\text{: if } v_{16} v_{18} v_{38} v_{44} v_{45} \text{ then } w_{14}\ ;$$
$$pr_{13}\text{: if } v_{39} \text{ then } w_{15};$$
$$pr_{14}\text{: if } v_{39} \text{ then } w_{16};$$
$$pr_{15}\text{: if } v_{39} \text{ then } w_{17};$$
$$pr_{16}\text{: if } \bar{v}_7 v_{31} v_{39} \text{ then } w_{18}\ ;$$

pr_{17}: if $v_7 v_{31} v_{39}$ then w_{19} ;

pr_{18}: if v_{39} then w_{20};

pr_{19}: if v_{41} then w_{21};

pr_{20}: if v_{42} then w_{22};

pr_{21}: if v_{40} then w_{23}.

Детализированное описание параметров, операций и продукций приведено в Приложении К.

Разные сценарии выполнения технологических процессов могут существовать в зависимости от сформулированной постановки задачи в концептуальной модели. Например, схемы на рисунках 6.10, 6.11 и 6.12 б являются мультисценарными, содержащими разные последовательности выполнения логистических операций.

Выбор того или иного сценария зависит от текущей ситуации в рамках технологического процесса, которая определяется логическими параметрами концептуальной модели. В этом случае целесообразно применять динамическое планирование последовательности операций.

В [294] предложен алгоритм динамического планирования действий на вычислительной модели. В диссертационной работе используется модификация данного алгоритма применительно к концептуальной модели ЛСК.

В имитационной модели работа алгоритма динамического планирования операций процесса функционирования ЛСК включает три основные фазы.

1) На первой фазе формулируется непроцедурная постановка задачи, представляемая в концептуальной модели ЛСК операциями w_1 и w_2.
2) Затем в рамках второй фазы формируется список продукций, для которых имеются все необходимые входные данные (значения логических параметров) для логических выражений в левых частях продукций, и осуществляется вычисление значений этих выражений.
3) Далее на заключительной фазе формируется и выполняется список операций из правых частей продукций с истинными значениями их логических

выражений. Выполняются операции, для которых заданы все необходимые входные данные, и производится переход ко второй фазе.

Вторая и третья фаза повторяются в цикле до тех пор, пока есть хотя бы одна готовая к выполнению продукция на второй фазе и хотя бы одна готовая к выполнению операция на третьей фазе.

Ниже приведен псевдокод алгоритма динамического планирования.

Пусть концептуальная модель включает m операций и n продукций, V_{exp} и V_t представляют соответственно множества исходных и целевых параметров, L_j – множество параметров, входящих в логическое выражение l_j в левой части j-й продукции, а W^* и W^{**} являются булевыми векторами размерности m. Ниже приведены основные этапы алгоритма А.7 динамического планирования процесса обслуживания.

/* Фаза 1 */

A.7.1. $V_{exp} = V_1^{out};\ V_t = V_2^{in};\ W^* = \emptyset;\ W^{**} = \emptyset;$

A.7.2. **for** $i = \overline{1,m}$ **step** 1 **do**

$w_i^* = 0;\ w_i^{**} = 0;$

enddo;

/* Фаза 2 */

A.7.3. m1: $fl1 = 0;$

A.7.4. **for** $j = \overline{1,n}$ **step** 1 **do**

if $\left(L_j \subseteq V_{exp}\ \wedge\ l_j = 1\right)$ **then**

$w_{i_j}^* = \overline{w}_{i_j}^{**};\ \ fl1 = 1;$

endif;

enddo;

A.7.5. **if** $(fl1 = 0)$ **then**

return("Задача неразрешима");

endif;

/* Фаза 3 */

A.7.6. $fl2 = 0;$

A.7.7. **if** $(F^* \neq \emptyset)$ **then**

for $i = \overline{1,m}$ **step** 1 **do**

if $\left(V_i^{in} \subseteq V_{exp}\right)$ **then**

call f_i;

$V_{exp} = V_{exp} \cup V_i^{out}$;

if $\left(V_t \subseteq V_{exp}\right)$ **then**

return("Процесс обслуживания спланирован");

endif;

$w_i^* = 0;\ w_i^{**} = 1;\ fl2 = 1;$

endif;

enddo;

endif;

A.7.8. **if** $(fl2 = 0)$ **then**

return("Задача неразрешима");

endif;

A.7.9. **goto** m1;

Если задача планирования неразрешима, то следует изменить постановку задачи и вновь попытаться решить ее.

В следующем разделе приводится технология построения имитационных моделей ЛСК, в рамках которых реализуются рассмотренные выше концептуальная модель и алгоритм динамического планирования действий на данной модели.

6.3.4. Имитационная модель ЛСК

Имитационная модель включает множества одноканальных и многоканальных сервисов. В общем случае обслуживание заявок клиентов в модели является многофазным.

Обслуживание очереди заявок к сервису осуществляется на основе дисциплины «First Come, First Serve» (FCFS) с резервированием. Резервирование ресурсов производится для заявок, заранее включенных в расписание их обслуживания.

В модели могут возникать неисправности сервисов обслуживания клиентов. Предполагается, что потоки заявок на обслуживание и неисправностей обладают свойствами стохастичности, дискретности, динамичности, неординарности, неоднородности, нестационарности и отсутствия обратной связи.

Интенсивности поступления заявок, обслуживания заявок и возникновения неисправностей подчиняются различным законам вероятностного распределения случайных величин. Выбор законов распределения случайных величин осуществляется на основе реальных данных, полученных в процессе работы склада. Их параметры уточняются и согласовываются с менеджером ЛСК.

Имитационная модель строится на языке GPSS по концептуальной модели ЛСК на основе каркасного подхода [252] с помощью инструментального комплекса SIRIUS II [356]. В рамках данного подхода имитационная модель состоит из двух обязательных частей: каркаса и набора гнезд (рисунок 6.13).

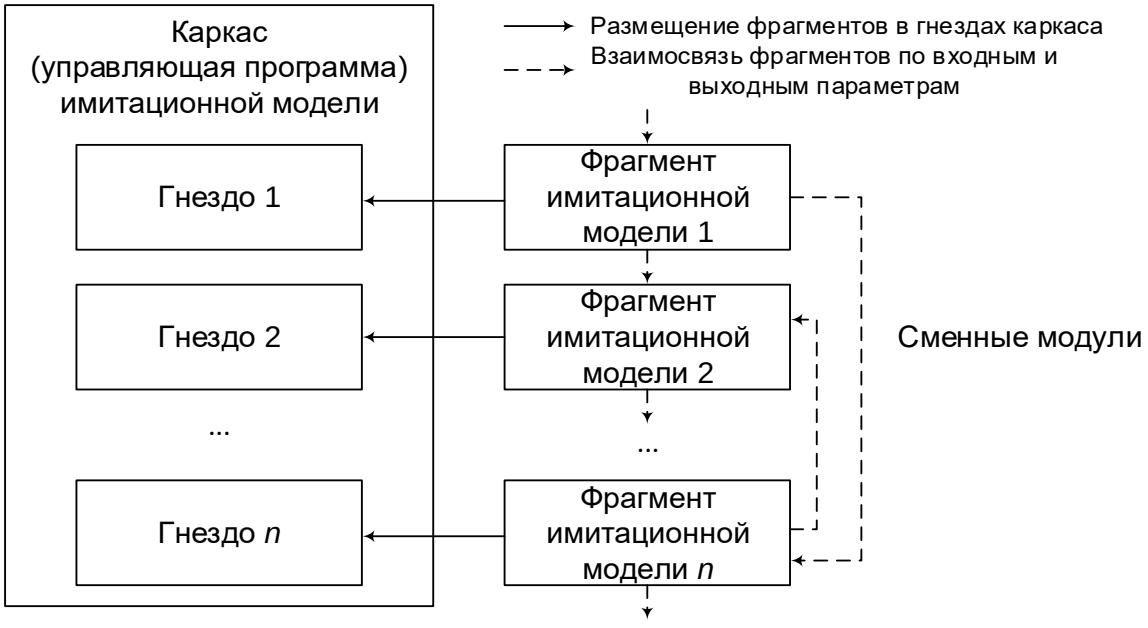


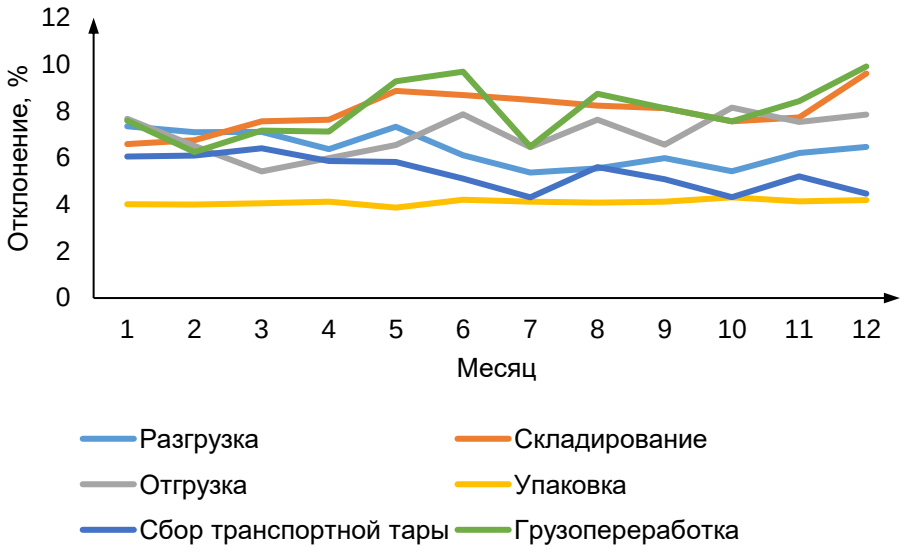
Рисунок 6.13 – Каркасный подход

Каркас имитационной модели представляет собой неизменяемую часть программы на GPSS и включает в себя совокупность изменяемых частей – гнезд, в которые помещаются сменные модули (фрагменты программы на GPSS, моделирующие отдельные процессы функционирования ЛСК).

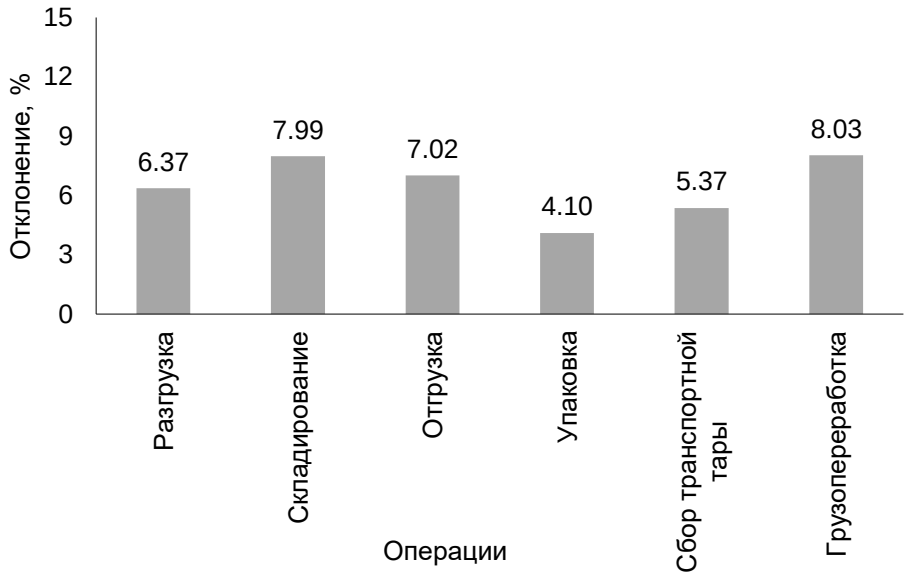
Динамический планировщик логистических операций реализуется в каркасе имитационной модели на основе концептуальной модели ЛСК. Вспомогательные аналитические модели расчета финансовых показателей функционирования ЛСК реализованы с помощью средств интегрирования системы GPSS World.

Проверка адекватности моделей осуществляется методами верификации и валидации. Верификация обеспечивает оценку корректности перевода концептуальной модели в имитационную программу. Правильность логики модели проверяется посредством интерактивного контроля за ходом моделирования при помощи встроенных в GPSS утилит отладки, анализа корректности результатов моделирования «крайние» значения (нулевые и несуществующие в реальности значения входных параметров модели); сравнения результатов моделирования с показателями, полученными путем аналитического расчета. Методы валидации применяются для проверки того, чтобы результаты моделирования обладали удовлетворительной точностью и не противоречат исследуемой системе. Валидация осуществляется путем сравнения откликов модели и реальной системы.

Отклонение результатов моделирования основных логистических операций от показателей реальной системы по месяцам года приведены на рисунке 6.14 а. Они показывают стабильность результатов моделирования. Усредненное отклонение результатов моделирования основных логистических операций от показателей реальной системы (рисунок 6.14 б) не превышает 8.03%, что позволяет считать используемые модели валидными. Стандартное отклонение результатов моделирования для логистических операций, приведенных на рисунке 6.14 б составляет соответственно 0.73, 0.86, 0.87, 0.11, 0.73 и 1.21. Данные результаты позволяют сделать вывод о достаточной точности работы имитационных моделей.



а)



б)

Рисунок 6.14 – Отклонение результатов моделирования основных логистических операций от реальных показателей по месяцам года (а) и их усредненное отклонение (б)

6.3.5. Вычислительный эксперимент

РППП, разработанный с помощью инструментального комплекса Orlando Tools, применен для исследования в ГРВС качества работы предприятия ООО «Иркутский хладокомбинат» и поддержки принятия решений по управлению им [66, 69, 74]. Данное предприятие является одним из крупнейших региональных складских комплексов, поддерживающих разнообразные температурные режимы хранения товаров, и включает склады типа А и В, а также рефрижераторный склад.

В настоящее время такие комплексы привлекают особое внимание, что

обусловлено следующими факторами:

- значительное число крупных российских производственных компаний, торговых сетей и иностранных компаний, работающих через дилерскую сеть в России, выбирают расширение сбыта продукции в регионах в качестве основного направления своего развития и остро нуждаются в качественных масштабных услугах складской логистики;
- существенно возрастает спрос на склады с низкотемпературными режимами хранения из-за увеличения потребления замороженных продуктов и полуфабрикатов, а также интенсивного развития животноводства в стране с учетом сезонного характера данного вида деятельности.

Данное предприятие позиционируется как накопительный складской комплекс (распределительная база) для закладки и хранения продукции с целью ее дальнейшей реализации в Восточной Сибири, так и транзитный склад при движении товара из восточных регионов России на Запад и в обратную сторону.

Рефрижераторный склад является вторым по величине промышленным хладокомбинатом в России на территории от Урала до Дальнего Востока, способным принимать на единовременное хранение до 20 тыс. тонн продукции.

Рефрижераторный склад имеет следующие характеристики:

- 6 этажей, 42 камеры с общей емкостью единовременного хранения 20 000 тонн и несколькими температурными режимами;
- 8 грузовых лифтов грузоподъемностью до 3 тонн каждый;
- 2 железнодорожных тупика;
- крытые рампы для автомобильного и железнодорожного транспорта, оборудованные электронными платформенными весами;
- парк погрузочно-разгрузочной техники;
- поддержка обслуживания грузов из 8 товарных групп;
- обеспечение допустимого «товарного соседства» (возможности складировать продукты питания различных товарных групп в одной камере);
- возможность предоставления услуг различным категориям клиентов, которые отличаются требуемым объемом и условиями хранения, типом

товаров, интенсивностью оборота, типом используемого транспорта, лояльностью и другими свойствами;

– объекты коммерческой недвижимости (непрофильные складские площади, офисы для клиентов, гаражи, места для обработки грузов, парковочные площадки для крупнотоннажного и легкового транспорта и др.), которые в зависимости от клиентского спроса могут быть использованы для иного назначения после их переоборудования;
– склад расположен в непосредственной близости к важной транспортной развязке, включающей грузовую железнодорожную станцию и федеральную автомобильную трассу.

Как показывают результаты моделирования транспортных потоков, широкий спектр транспортных средств существенно затрудняет планирование обслуживание потока заявок [73].

В соответствии со своими характеристиками складской комплекс оказывает следующие услуги: хранения на условиях аренды холодильных площадей или ответственного хранения; приемки и отправки грузов железнодорожным и автомобильным транспортом; раскредитования, оформления и отслеживание железнодорожных вагонов в пути следования; пользования железнодорожными тупиками; ветеринарного контроля; дозаморозки продукции; погрузочно-разгрузочных работ; утилизации бракованной продукции; предоставлении информационных услуг; оформления необходимой складской и транспортной документации. Предоставление складских услуг клиентам осуществляет ООО «Терминал Комплекс».

Специфической особенностью выполнения складских операций на Иркутском хладокомбинате является необходимость учета этажности рефрижераторного склада и использование грузовых лифтов. Эта особенность характерна для целого класса хладокомбинатов советской эпохи, расположенных на всем протяжении транссибирской железнодорожной магистрали, и, как правило, не учитывается в современных системах управления складом. Однако в настоящее время хладокомбинаты данного класса по-прежнему представляют существенную

долю на рынке холодильных услуг.

Большая трудоемкость подъема товара на этажи с помощью лифтов усложняет проведение погрузочно-разгрузочных работ, повышается вероятность простоя транспорта в ожидании выгрузки товара. Тем самым актуализируется оптимизация технологических процессов на складе.

Ниже приведены три типовые задачи исследования процессов функционирования хладокомбината на основе имитационного моделирования в ГРВС. В качестве основных показателей функционирования хладокомбината в приведенных ниже задачах использовались показатели качества и уровня стабильности функционирования, а также производительности ЛСК, приведенные в работах [256, 278]. Эти показатели положены в основу оценки качества логистического сервиса, выполнения логистических операций и функционирования исследуемой системы.

Базовыми параметрами склада, выделенными в процессе структурного анализа предметной области для рассматриваемых задач, являются число лифтов, кладовщиков, грузчиков, электропогрузчиков, бригад грузчиков, складских операций, клиентов, характеристик клиентов, позиций товаров, видов хранения товаров, видов товарной упаковки, параметров расписания обслуживания, сезонов, плановых и случайных заявок, а также списки клиентов, характеристик клиентов, позиций товаров и видов упаковки товаров, расписание прихода плановых заявок, интенсивность поступления случайных заявок, стоимость складских операций и хранения товаров, расписания работы бригад грузчиков и технического обслуживания электропогрузчиков.

Задача 1. Совершенствование процессов разгрузки и отгрузки товаров. Разгрузка – логистическая операция, заключающаяся в освобождении транспортного средства от груза. Отгрузка – логистическая операция, заключающаяся в подаче, ориентировании и укладке груза в транспортное средство. Технология выполнения погрузочно-разгрузочных работ на складе зависит от характера груза, от типа транспортного средства, а также от вида используемых средств механизации. Для обслуживания заявок используются

технические и человеческие ресурсы: электропогрузчики, водители электопогрузчиков, грузчики и кладовщики.

Рефрижераторный склад обеспечивает погрузку и разгрузку в процессе обслуживания входных и выходных материальных потоков в соответствии с плановыми и случайными заявками клиентов. Плановые заявки формируются каждые сутки и содержат информацию о времени разгрузки и отгрузки товаров, времени обслуживания заявки, видах и объемах товаров, а также необходимых ресурсах для выполнения необходимых операций. Примеры вариантов плановых и случайных заявок, используемых в процессе решения данной задачи, приведены в Приложении Л.

Необходимо определить управляющие воздействия в информационно-управляющей системе на параметры процесса обслуживания входных и выходных материальных потоков, чтобы улучшить следующие критерии: коэффициенты позезного использования кладовщиков, грузчиков, электропогрузчиков, бригад грузчиков и работы лифтов, а также среднее время выполнения логистических операций.

Шесть вышеперечисленных критериев используются в качестве наблюдаемых переменных модели. Множество входных параметров включает базовые параметры склада. Варьируемые входные параметры и диапазоны их значений определяются менеджером по логистике.

Задача 2. Реструктуризация процесса обслуживания клиентов. Рефрижераторный склад предоставляет услуги разным категориям клиентов, обуславливающим использование различных временны́х, технических и людских ресурсов. Необходимо определить управляющие воздействия в информационно-управляющей системе на параметры обслуживания клиентов с целью улучшения таких критериев, как наполнение склада и доход от логистических операций.

Два вышеперечисленные критерия используются в качестве наблюдаемых переменных модели. Множество входных параметров включает базовые параметры склада, а также число уровней обслуживания клиентов и их список. Варьируемые входные параметры и диапазоны их значений определяются

менеджером по логистике.

Реструктуризация процесса обслуживания клиентов включает категоризацию клиентов на основе качественных и количественных характеристик клиентов, определение уровня обслуживания для разных категорий клиентов и выбор стандартов выполнения логистических операций, связанных с их обслуживанием. Клиенты рефрижераторного склада подразделяются на четыре категории: A, B, C и D. Эти категории характеризуются различными объемами товарооборота в месяц: до 50 тонн (D), от 50 до 200 тонн (C), от 200 до 600 тонн (B) и более 600 тонн (A).

Больший объем грузооборота обуславливает повышение рентабельности логистических операций по сравнению с меньшими объемами грузооборота. В то же время чрезмерный рост клиентов с большим грузооборотом, с одной стороны, может привести к превышению пропускной способности склада, а с другой – повышает риски резкого снижения наполненности склада и прибыли в случае их отказа от обслуживания.

Задача 3. Переоборудование и сдача в аренду дополнительных объектов коммерческой недвижимости. Рефрижераторный склад реализует проект, связанный с переоборудованием и сдачей в аренду дополнительных объектов коммерческой недвижимости. Необходимо определить управляющие воздействия в информационно-управляющей системе на параметры процесса переоборудования объектов с целью увеличения дохода от складских объектов.

Данный критерий используется в качестве наблюдаемой переменной модели. Множество входных параметров включает базовые параметры склада, а также число объектов склада и их видов, число сезонов спроса на объекты хранения товаров, показатели спроса для каждого сезона и стоимость переоборудования объектов. Варьируемые входные параметры и диапазоны их значений определяются менеджером по логистике.

Процесс решения задач. Для описания предметной области пакета в инструментальном комплексе Orlando Tools используется вычислительная модель, рассмотренная в разделе 2.1. В ней логические и информационные отношения

между ее параметрами и операциями отражают схемные знания о рассматриваемой предметной области.

Фрагмент вычислительной модели, разработанный с помощью инструментального комплекса Orlando Tools, представлен на рисунке 6.15. Параметры и операции в редакторе Orlando Tools представлены соответственно овалами и прямоугольниками с закругленными углами. Стрелки отражают взаимосвязь между параметрами и операциями.

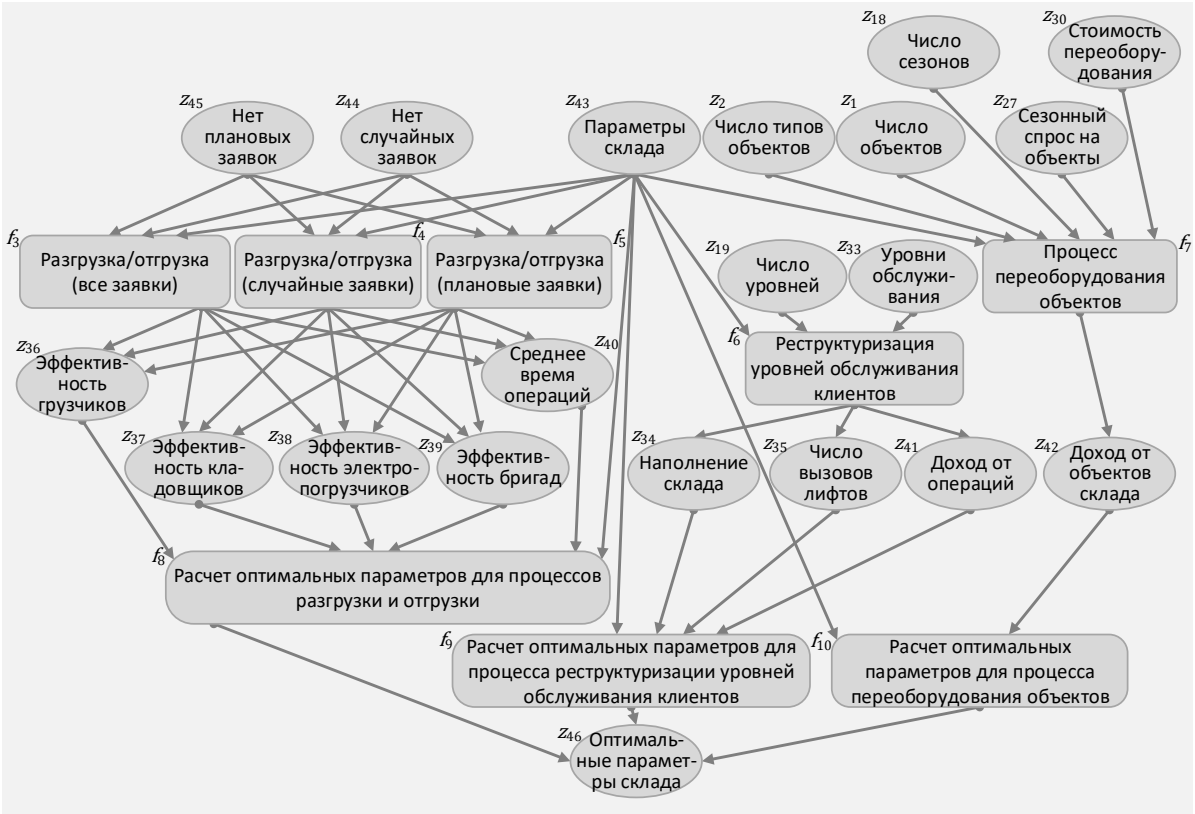


Рисунок 6.15 – Концептуальная модель пакета

На рисунке 7.15 z_{43} и z_{46} являются составными параметрами. Составной параметр z_{43} включает в себя следующие параметры:

– число лифтов (z_3), кладовщиков (z_4), грузчиков (z_5), электропогрузчиков (z_6), бригад грузчиков (z_7), логистических операций (z_8), клиентов (z_9), характеристик клиентов (z_{10}), позиций товаров (z_{11}), характеристик товаров (z_{12}), видов хранения товаров (z_{13}), типов упаковки товаров (z_{14}), параметров

расписаний обслуживания (z_{15}), плановых заявок (z_{16}) и случайных заявок (z_{17});

- списки клиентов (z_{20}), характеристик клиентов (z_{21}), позиций товаров (z_{22}) и типов упаковки товаров (z_{23});
- расписание обслуживания плановых заявок (z_{24}), среднее время поступления случайных заявок (z_{25}), половина интервала времени нормального распределения (z_{26}), расписание работы бригад грузчиков (z_{31}) и расписание технического обслуживания электропогрузчиков (z_{32});
- стоимость логистических операций (z_{28}) и хранения товаров (z_{29}).

Составной параметр z_{46} включает в себя параметры $z_1 - z_{33}$. Он представляет оптимальные значения этих параметров.

В концептуальную модель включены два параметра (критерия) z_{47} и z_{48}. Эти параметры определяют ограничения на время и надежность процесса решения задач, рассмотренных выше.

Операции $f_3 - f_5$ представляют различные сценарии погрузки и отгрузки товаров. В связи с этим в модель добавлены набор следующих продукций, определяющих условия выполнения операций $f_3 - f_5$:

$$pr_1\text{: if } \bar{z}_{44}\bar{z}_{45} \text{ then } f_3\ ,$$
$$pr_2\text{: if } \bar{z}_{44}z_{45} = 0 \text{ then } f_4\ ,$$
$$pr_3\text{: if } z_{44}\bar{z}_{45} = 0 \text{ then } f_5\ .$$

Операции $f_3 - f_{10}$ реализуются модулями $m_1 - m_7$ соответственно. Эти модули представляют собой GPSS-модели. Передача данных между модулями осуществляется через файлы. РППП обеспечивает параллельное выполнение копий модулей $m_1 - m_5$ с различными вариантами их входных данных, сгенерированные путем варьирования допустимых значений входных параметров.

Конечный пользователь пакета может сформулировать следующие постановки задач на концептуальной модели:

- вычислить z_{46} по z_{43}, z_{44}, z_{45} с заданными критериями z_{47}, z_{48};
- вычислить z_{46} по z_{19}, z_{33}, z_{43} с заданными критериями z_{47}, z_{48};

– вычислить z_{46} по $z_1, z_1, z_{18}, z_{27}, z_{43}$ с заданными критериями z_{47}, z_{48}.

В соответствии со сформулированными постановками задач исполнительная подсистема Orlando Tools построит следующие схемы решения задач:

$$s_1: [f_1] \to [f_3|f_4|f_5] \to [f_8] \to [f_2],$$
$$s_2: [f_1] \to [f_6] \to [f_8] \to [f_2],$$
$$s_3: [f_1] \to [f_7] \to [f_8] \to [f_2],$$

где квадратными скобками […] обозначены ярусы схемы, а символы '→' и '|' определяют соответственно последовательность и параллельность выполнения операций. Формальные операции f_1 and f_2 моделируют постановку задачи. Они соответственно определяют множество параметров Z_1^{out}, значения которых известны, и множество параметров Z_2^{in}, значения которых нужно вычислить, $Z_1^{in}, Z_2^{out} = \emptyset$. Схемы $s_1 - s_3$ обрабатываются в режиме интерпретации. Операции схем выполняются в асинхронном режиме по готовности данных. На втором ярусе схемы s_1 формально возможен параллельный запуск операций $f_3 - f_5$. Однако они реализуют альтернативные алгоритмы и выбор из них конкретной операции, которая будет выполнена, определяется продукциями $pr_1 - pr_3$.

Конечный пользователь пакета выбирает схему решения задачи из списка имеющихся схем и задает значения входных параметров схемы, а также критерии качества ее решения. Для варьируемых параметров он определяет их допустимые значения и выбирает тип эксперимента (полный или частичный), который влияет на число вариантов исходных данных.

Когда все исходные данные определены, исполнительная подсистема Orlando Tools автоматически генерирует варианты значений входных параметров (один вариант для каждого экземпляра схемы). Пользователь определяет многокритериальный метод выбора (лексикографический, мажоритарный или выбор по Парето) оптимальных значений наблюдаемых параметров. Затем он конфигурирует с помощью Orlando Tools необходимую вычислительную инфраструктуру ГРВС и запускает в ней распределенные вычисления. Один экземпляр схемы выполняется на одном ядре.

Исполнительная подсистема Orlando Tools распределяет все экземпляры схемы между используемыми вычислительными системами с учетом производительности их узлов относительно данной схемы. Интерпретатор Orlando Tools, который размещается в основном узле каждой вычислительной системы, распределяет нагрузку между ее узлами.

Для решения каждой задачи разработана своя имитационная модель. Модели, используемые для решения задач 2 и 3, являются расширениями модели, разработанной для решения задачи 1.

Эксперименты проведены на основе сформулированных задач 1-3. Моделируемый период складских работ составлял один год. Такой период выбран из-за того, что объемы товарооборота и требования к обслуживанию клиентов различаются по кварталам года. Единица времени моделирования – 1 минута. Диапазоны значений варьируемых параметров имели от 2 до 24 уровней. Число прогонов модели для одного варианта данных составляло в разных задачах от 100 до 200, что обеспечило ошибку значений наблюдаемых переменных, не превышающую 0.05.

В процессе решения задач 1-3

- улучшена долевая структура клиентов складского комплекса – численный состав категорий A-D (рисунок 6.16 а);
- увеличен и сбалансирован его суточный товарооборот (рисунок 6.16 б);
- увеличен доход от сдачи в аренду дополнительных объектов коммерческой недвижимости по сравнению с предыдущим годом (рисунок 6.16 в).

В том числе улучшены следующие критерии (рисунок 6.16 г):

- c_1 – коэффициент полезного использования кладовщиков (15%);
- c_2 – коэффициент полезного использования грузчиков (9%);
- c_3 – коэффициент полезного использования электропогрузчиков (12%);
- c_4 – коэффициент полезного использования бригад грузчиков (8 %);
- c_5 – коэффициент полезного использования лифтов (10 %);
- c_6 – среднее время выполнения логистических операций (6 %);

– c_7 – доход от логистических операций (17 %);

– c_8 – наполнение склада (8 %);

– c_9 – доход от складских объектов (34 %).

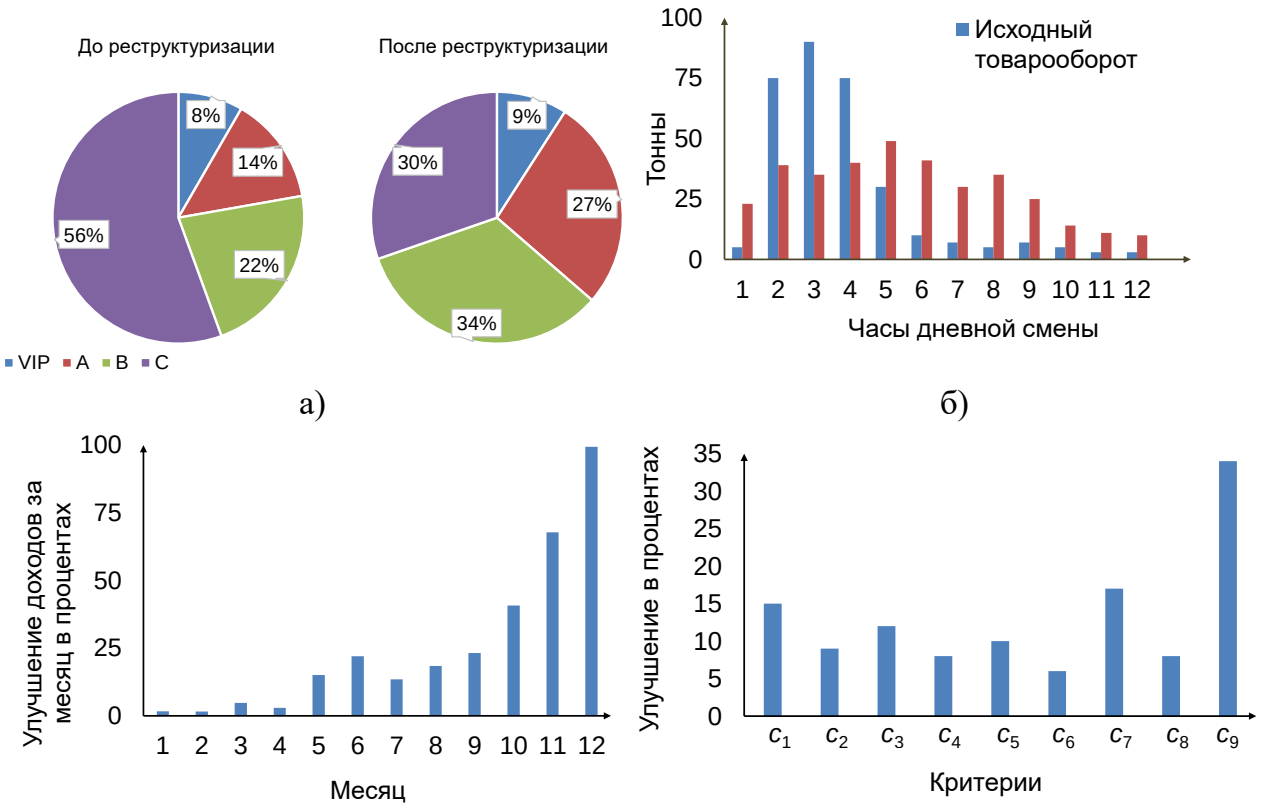


Рисунок 6.16 – Результаты решения задач: улучшение структуры клиентов (а), увеличение и балансировка суточного товарооборота (б), увеличение дохода от сдачи в аренду дополнительных объектов коммерческой недвижимости (в), улучшение критериев качества работы склада (г)

Улучшение критериев варьируются от 6 до 34%.

Вследствие решения задач и формирования управляющих воздействий в информационно-управляющей системе улучшена рентабельность ряда операций (рисунок 6.17). Данный результат во многом обеспечен уменьшением числа клиентов категории D и увеличением числа клиентов категорий A, B и C.

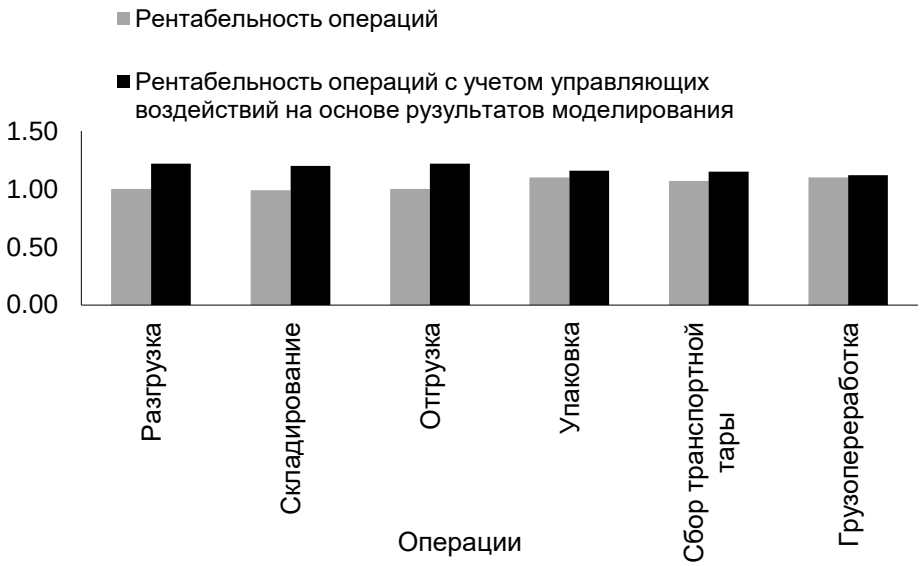


Рисунок 6.17 – Улучшение рентабельности операций

Результаты имитационного моделирования были использованы в управленческих службах рефрижераторного склада при разработке методики формирования плановых и учета случайных заявок клиентов, формировании состава и графика работ бригад грузчиков и кладовщиков, планировании технического обслуживания и обновления парка электропогрузчиков, определении категорий «лояльных» клиентов, а также расчете нормативов и финансовых показателей по выполнению логистических операций.

6.3.6. Экспериментальный анализ

Вычислительные эксперименты спланированы исходя из сформулированных выше постановок задач, в которых заданы ограничения на параметры $z_{47} \leq 7$ и $z_{48} \geq 0.99$. Значение параметра z_{47} измеряется в часах.

Время выполнения модели на эталонном узле с одним из вариантов данных составляет 14, 17 и 19 секунд в задачах 1-3 соответственно. Используя Orlando Tools можно спрогнозировать время и надежность схем решения задач для используемых вычислительных систем. В рамках вычислительных экспериментов были доступны четыре системы:

– персональный компьютер (Intel Core i3-4160, 2 ядра с гипертрейдингом, 3.6 GHz, 4 GB RAM), принадлежащий предприятию ООО «Иркутский

хладокомбинат»;

– персональный компьютер (Intel Core i7-4770, 4 ядра с гипертрейдингом, 3.4 GHz, 4 GB RAM), принадлежащий предприятию ООО «Иркутский хладокомбинат»;
– ПК-кластер (16 узлов с одним процессором Intel Core i3-4000M, 2 ядра с гипертрейдингом, 2.4 GHz, 2 GB RAM), который организован на базе персональных компьютеров одного из учебных классов МИЭЛ ИГУ;
– сегмент HPC-кластера «Академик В.М. Матросов» ЦКП ИСКЦ (15 узлов с двумя процессорами AMD Opteron 6276, 16 ядер, 2.3 GHz, 64 GB of RAM).

В таблице 28 приведены оценки t_1, t_2, t_3, и t_4 времени решения задач в часах для ПК 1, ПК 2, ПК-кластера и HPC-кластера соответственно. Значения времени, удовлетворяющие заданному ограничению, выделены серым цветом. Наиболее предпочтительные типы экспериментов из числа возможных выделены зеленым цветом.

Таблица 28 – Оценки времени решения задач

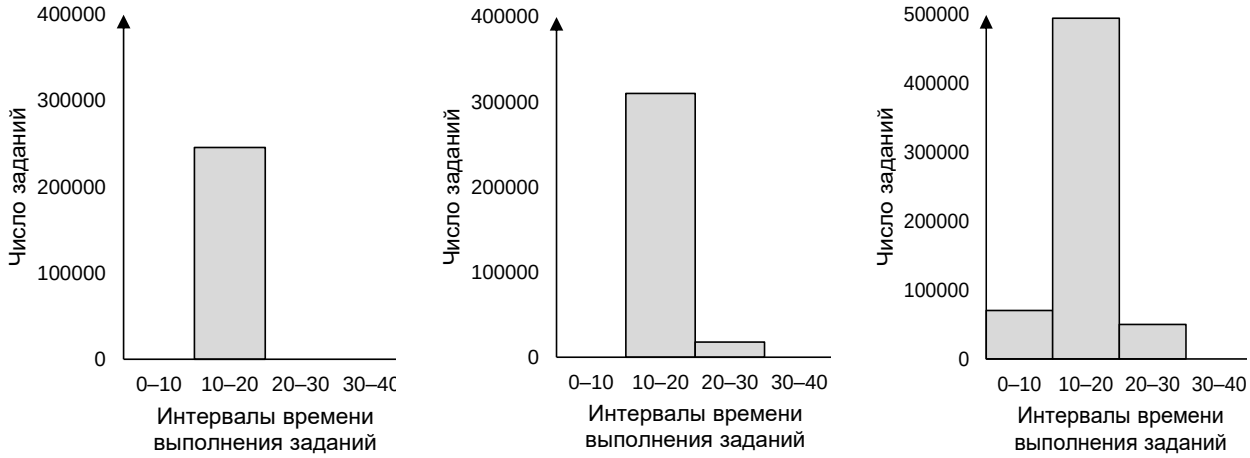
Задача	Тип эксперимента	Число вариантов	t_1	t_2	t_3	t_4
1	Полный	245760	265.48	149.33	37.33	1.99
	1/2	122880	132.74	74.67	18.67	1.00
	1/4	61440	66.37	18.67	9.33	0.51
	1/8	30720	33.19	18.67	4.67	0.26
2	Полный	327680	429.83	241.78	60.44	3.21
	1/2	163840	214.91	120.89	30.22	1.61
	1/4	81920	107.46	60.44	15.11	0.81
	1/8	40960	53.73	30.22	7.56	0.40
3	Полный	1140480	1672.00	940.50	235.13	12.54
	1/2	570240	836.00	470.25	117.56	6.28
	1/4	285120	418.00	235.13	58.78	3.14
	1/8	142560	209.00	117.56	29.39	1.58

Оценки показателей надежности схемы решения задачи 1 равны 95.3425,

96.1644, 0.9103 и 0.9999 при использовании ПК 1, ПК 2, ПК-кластера и HPC-кластера соответственно. Оценки показателей надежности для схем решения задач 2 и 3 эквиваленты приведенным выше оценкам для задачи 1.

Очевидно, что только HPC-кластер удовлетворяет заданной надежности вычислений. Таким образом, на основе полученных оценок были выбраны полный (задача 1 и 2) и 1/2 (задача 3) типы эксперимента. Из числа возможных (допустимых) вариантов проведения эксперимента были выбраны варианты с минимальными оценками времени решения задач.

На рисунке 6.18 а-в приведены графики плотности распределения времени выполнения трех потоков заданий для задач 1-3 соответственно. Число заданий в потоках составляло соответственно 245760, 327680 и 570240 единиц. Оценочное время решения задач 1-3 на одном ядре составляет соответственно более 14, 37 и 125 суток.



а) б) в)

Рисунок 6.18 – Графики плотности распределения времени выполнения заданий для задачи 1 (а), задачи 2 (б) и задачи 3 (в)

Время выполнения подавляющего числа заданий является достаточно близким друг к другу и находится в интервале от 10 до 20 секунд. Минимальное, среднее и максимальное время выполнения заданий приведено на рисунке 6.19.

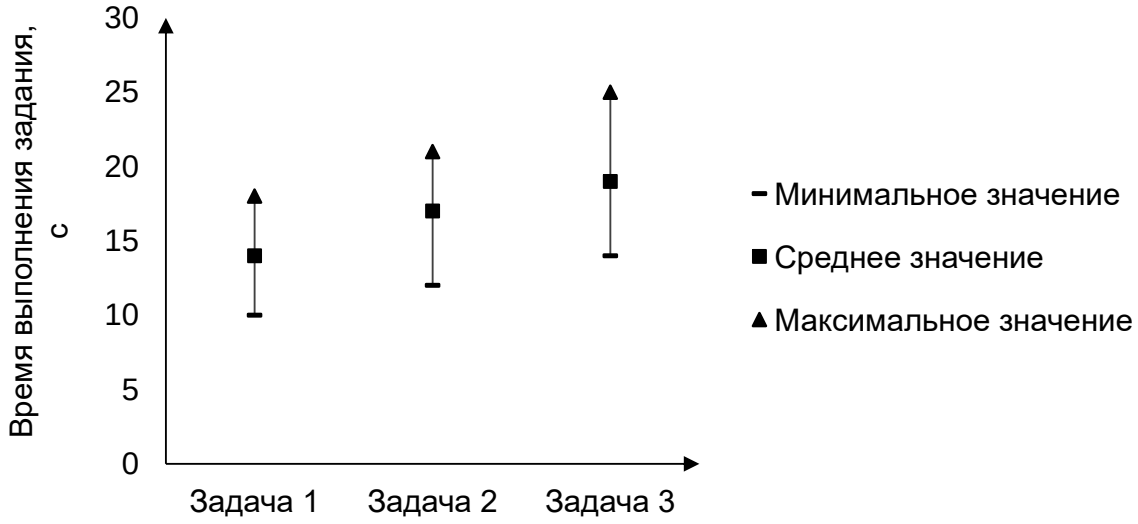
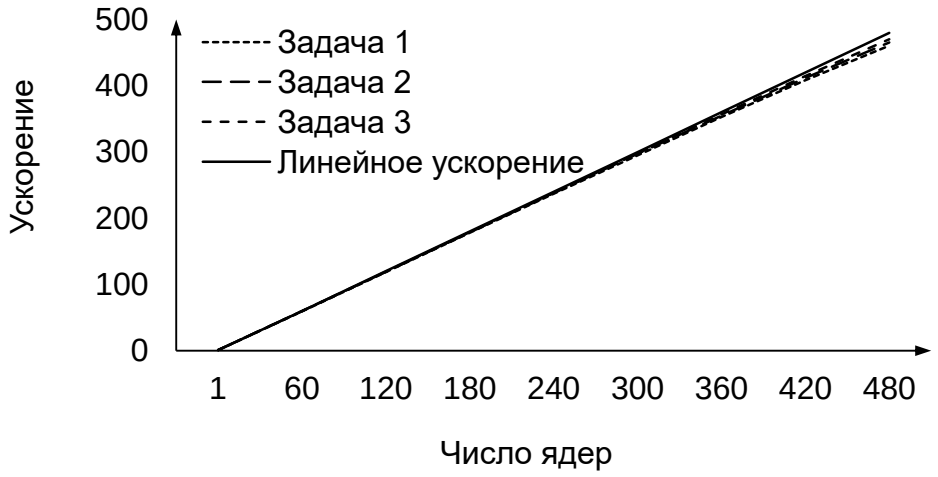


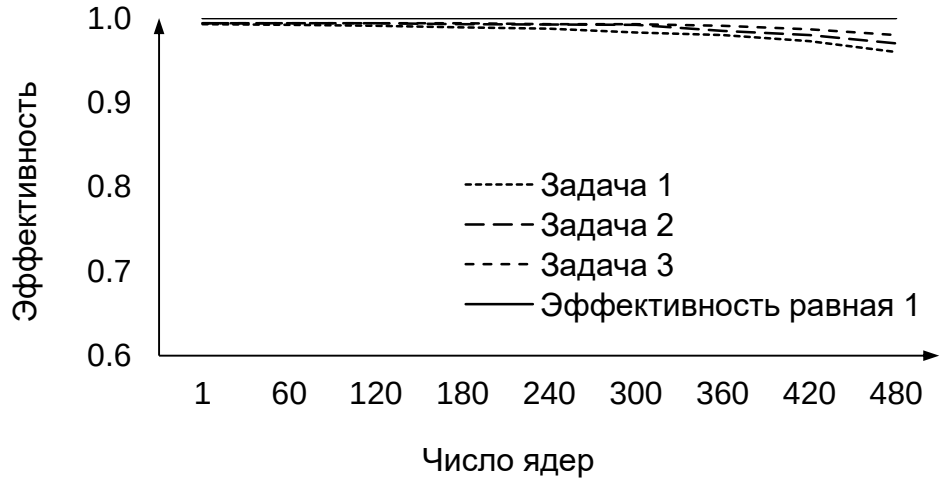
Рисунок 6.19 – Статистика времени выполнения заданий

На основе факторного анализа для каждой задачи были определены входные переменные моделей, влияющие на существенное отклонение времени выполнения заданий от его среднего значения. К таким переменным относится, например, варьируемый входной параметр имитационной модели, определяющий интенсивность поступления случайных заявок в задаче 1. Результаты факторного анализа позволили производить оценку времени выполнения заданий в процессе мультиагентного управления вычислениями с погрешностью в пределах 20%.

Рисунки 6.20 а и 6.20 б демонстрируют соответственно рост ускорения и стабильность эффективности использования ресурсов в процессе решения задач по мере увеличения числа используемых ядер. Данные результаты показывают высокую масштабируемость вычислений при выполнении заданий пакета. Задачи 1-3 упорядочены по возрастанию относительно числа обрабатываемых вариантов данных. Очевидно, что более ресурсоемкая задача 3 обеспечивает наилучшую масштабируемость.



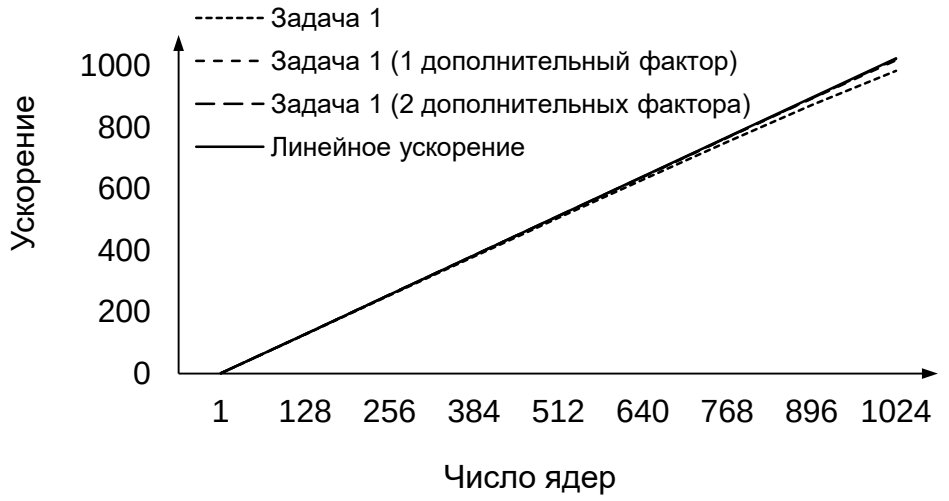
а)



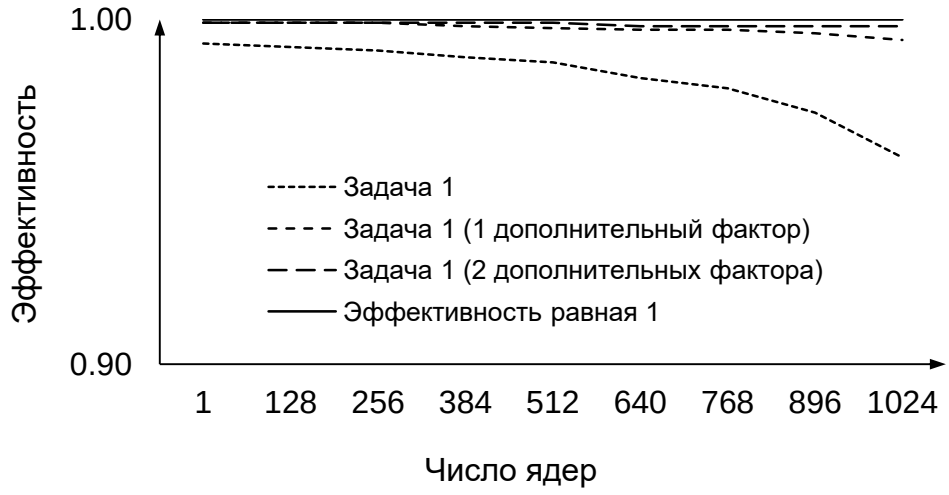
б)

Рисунок 6.20 – Изменение ускорения вычислений (а) и эффективности использования ресурсов (б) по мере увеличения числа используемых ядер

На рисунках 6.21 а и 6.21 б приведены соответственно оценки ускорения и эффективности, полученные для задачи 1 с разным числом факторов (входных переменных модели). Добавление одного нового фактора увеличивает число исходных вариантов данных в k раз, где k – это число уровней (допустимых значений) фактора. Это позволяет стабилизировать ускорение процесса вычислений и существенно повысить эффективность использования ресурсов.



а)

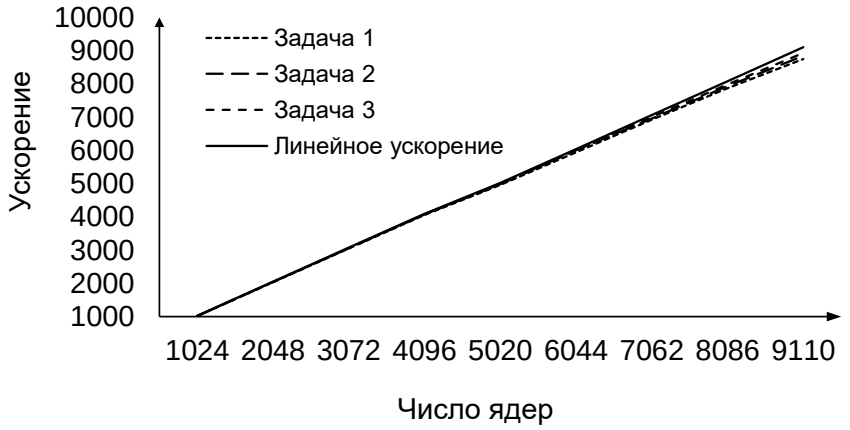


б)

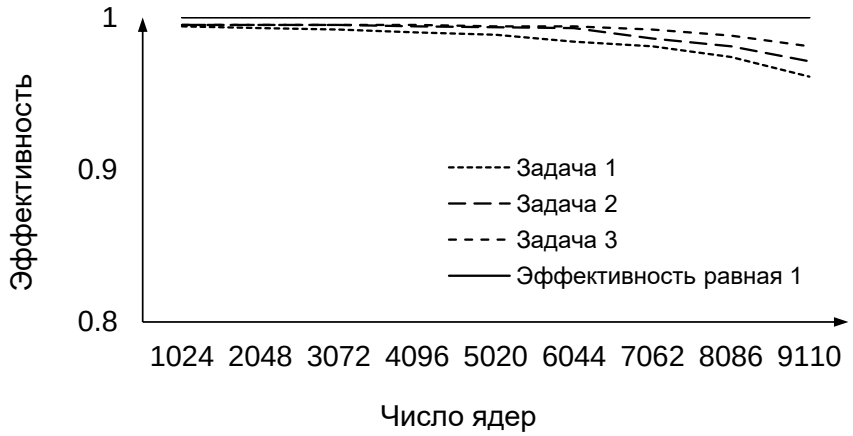
Рисунок 6.21 – Изменение ускорения вычислений (а) и эффективности использования ресурсов (б) при росте числа факторов

Для оценки масштабируемости вычислений проведено имитационное моделирование работы РППП в ГРВС, включающей в совокупности 30000 ядер. При этом для выполнения вычислений выделялось от 1024 до 9110 ядер разнородных узлов. На рисунках 6.22 а и 6.22 б приведены соответственно ускорение и эффективность, полученные в результате эксперимента. Результаты

моделирования подтверждают масштабируемость вычислений при росте числа ядер.



а)



б)

Рисунок 6.22 – Изменение ускорения вычислений (а) и эффективности использования ресурсов (б) при росте числа факторов

На рисунке 6.23 приведены результаты сравнительного анализа времени выполнения этапов решения задачи с 1024 вариантами исходных данных на основе моделирования с помощью разработанного РППП и «вручную». Были рассмотрены этапы, связанные с обработкой данных, заданий и результатов моделирования. Результаты выполнения этапов решения задачи представляют усредненные оценки времени выполнения этих этапов 20 студентами с помощью разработанного РППП

и «вручную». Очевидно существенное сокращение времени выполнения данных этапов решения задачи при использовании разработанного РППП.

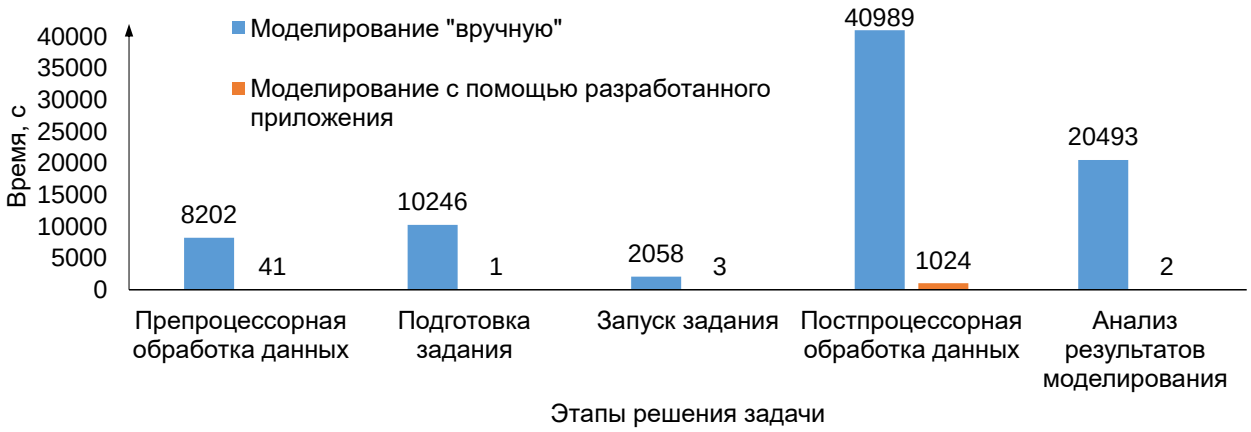


Рисунок 6.23 – Оценка времени, затрачиваемого на различных этапах

6.4. Выводы

В шестой главе продемонстрировано применение результатов диссертационного исследования при разработке РППП. Приведены научные и прикладные задачи, которые решены на основе РППП, разработанных с помощью инструментальных комплексов DISCOMP и Orlando Tools, а также в экспериментальной Грид, созданной на базе инструментального комплекса DISCENT. Более детально рассмотрены пакеты для выявления критических элементов отраслевых систем энергетики и решения задач складской логистики.

В целом в шестой главе получены следующие основные результаты:

– разработан РППП для решения задачи выявления критических элементов систем энергетики, применение которого позволило экспертам из ИСЭМ СО РАН составить список критически важных объектов газотранспортной сети России и предложить некоторые мероприятия по повышению живучести системы газоснабжения России;
– разработан РППП для решения задач складской логистики для специального класса рефрижераторных складов, которое, в отличие от известных средств подобного назначения, ориентировано на автоматизацию построения имитационной модели на основе описания предметной области и

выполнения многовариантных расчетов в гетерогенной среде с использованием разнородных вычислительных ресурсов;

– проведен экспериментальный анализ, который показал масштабируемость вычислений, выполняемых с помощью разработанных пакетов, при росте числа используемых ресурсов среды, а также сокращение сроков решения задач за счет автоматизации основных этапов подготовки и проведения экспериментов.

Результаты исследований, представленные в данной главе, опубликованы в [66, 67, 69, 73–76, 78, 81, 324, 332, 339, 356, 375].

Заключение

Главным результатом диссертации является разработка моделей, алгоритмов и инструментальных средств организации мультиагентного управления предметно-ориентированными распределенными вычислениями в ГРВС. Характерной особенностью предложенного подхода к организации мультиагентного управления в ГРВС является применение экономических механизмов регулирования спроса и предложения ресурсов среды.

Основные результаты диссертационной работы:

1) разработана агрегированная модель ГРВС, которая в сравнении с подобными моделями обеспечивает взаимосвязанное представление алгоритмических знаний предметных областей решаемых задач, а также знаний о программно-аппаратной инфраструктуре среды и административных политиках использования ее ресурсов;
2) предложены модели и алгоритмы определения показателей качества решения задач в ГРВС, базирующиеся на применении предложенной агрегированной модели среды и набора специализированных методов прогнозирования времени выполнения заданий;
3) создана система классификации заданий, позволяющая в отличие от известных систем подобного назначения привлечь дополнительные экспертные знания администраторов узлов ГРВС для детализации спецификаций вычислительных процессов решения пользовательских задач относительно особенностей ресурсов среды с целью снижения неопределенности в распределении заданий по узлам;
4) предложен мультиагентный алгоритм планирования вычислений и распределения ресурсов, характерными особенностями которого являются применение экономических механизмов регулирования спроса и предложения этих ресурсов, а также возможность его адаптации к различным моделям сочетания критериев качества решения задач и предпочтений владельцев ресурсов на основе методов дискретного многокритериального

выбора;

5) разработан пакетный подход к организации предметно-ориентированных вычислений, базирующийся на построении РППП, среда функционирования которых в отличие от других подобных сред может включать ресурсы HPC-кластеров, грид-систем, облачных инфраструктур и других программно-аппаратных компонентов, а также обеспечивать их интегрированное использование для решения крупномасштабных задач;
6) создана технология предметно-ориентированных распределенных вычислений в ГРВС, интегрирующая вышеупомянутые модели, алгоритмы, систему классификации заданий и пакетный подход, специализированные инструментальные средства создания и применения РППП, а также программно-аппаратные ресурсы среды в рамках единой технологической цепочки решения крупномасштабных задач.

Предложенная в диссертации технология предметно-ориентированных распределенных вычислений и мультиагентного управления ими в ГРВС, рассмотренных в работе, допускает свое естественное развитие и обобщение применительно к другим классам вычислительных и управляющих систем.

Литература

1. A graph-based computational framework for simulation and optimisation of coupled infrastructure networks / J. Jalving, S. Abhyankar, K. Kim, M. Hereld, V.M. Zavala // IET Generation, Transmission and Distribution. — 2017. — Vol. 11. — № 12. — P. 3163–3176.

2. A Self-organising Federation of Alchemi Desktop Grids / R. Ranjan, X. Chu, C.A. Queiroz, A. Harwood, R. Buyya. — Victoria: Department of Computer Science and Software Engineering of the University of Melbourne, 2007. — 9 p.

3. A self-organization model for complex computing and communication systems / D.C. Marinescu, J.P. Morrison, C. Yu, C. Norvik, H.J. Siegel // Proc. of the 2nd Intern. Conf. on Self-Adaptive and Self-Organizing Systems 2008. — IEEE CS Press, 2008. — P. 149–158.

4. Aad, G. The ATLAS Experiment at the CERN Large Hadron Collider / G. Aad, E. Abat, J. Abdallah et al. // J. of Instrumentation. — 2008. — Vol. 3. — P. S08003.

5. Abramovici, A. LIGO: the laser interferometer gravitational–wave observatory / A. Abramovici, W.E. Althouse, R.W.P. Drever et al. // Science. — 1992. — Vol. 256, № 5005. — P. 325–333.

6. Adhianto, L. HPCToolkit: Tools for performance analysis of optimized parallel programs / L. Adhianto, S. Banerjee, M. Fagan et al. // Concurrency and Computation: Practice and Experience. — 2010. — Vol. 22, № 6. — P. 685–701.

7. Allen, B. Globus: A Case Study in Software as a Service for Scientists / B. Allen, R. Ananthakrishnan, K. Chard et al. // Proc. of the 8th Workshop on Scientific Cloud Computing. — ACM, 2017. — P. 25–32.

8. Altameem, T. An Agent–based Approach for Dynamic Adjustment of Scheduled Jobs in Computational Grids / T. Altameem, M. Amoon // J. of Computer and Systems Sciences International. — 2010. — Vol. 49, № 5. — P. 765–772.

9. Altintas, I. Kepler: an extensible system for design and execution of scientific workflows / I. Altintas, C. Berkley, E. Jaeger et al. // Proc. of the 16th Intern. Conf. on Scientific and Statistical Database Management. — N.Y.: IEEE, 2004. — P. 423–424.

10. Amaris, M. A Comparison of GPU Execution Time Prediction Using Machine Learning and Analytical Modeling / M. Amaris, R.Y. de Camargo, M. Dyab et al. // Proc. of the 15th Intern. Symposium on Network Computing and Applications. — IEEE, 2016. — P. 326–333.

11. Amato, A. A Distributed Agent–Based Decision Support for Cloud Brokering / A. Amato, S. Venticinque // Scalable Computing: Practice and Experience. — 2014. — Vol. 15, № 1. — P. 65–78.

12. Amazon Elastic Compute Cloud [Электронный ресурс]. — 2018. — Режим доступа: https://aws.amazon.com/ru/ec2/ (дата обращения: 24.09.2021).

13. An introduction to software agents / Ed. J.M. Bradshaw // Software agents. — MIT press, 1997. — P. 3–46.

14. Auction protocols for decentralized scheduling / M.P. Wellman et al. // Games and economic behavior. — 2001. — Vol. 35, № 1. — P. 271–303.

15. Ausubel, L.M. The lovely but lonely Vickrey auction / L.M. Ausubel, P. Milgrom // Combinatorial auctions. — 2006. — Vol. 17. — P. 22–26.

16. Bagwell, L.S. Dutch auction repurchases: An analysis of shareholder heterogeneity / L.S. Bagwell // The Journal of Finance. — 1992. — Vol. 47, № 1. — P. 71–105.

17. Belhajjame, K. A flexible workflow model for process–oriented Applications / K. Belhajjame, G. Vargas–Solar, C. Collet // Proc. of the Second Intern. Conf. on Web Information Systems Engineering. — IEEE Computer Society, 2001. — Vol. 1, № 1. — P. 72–80.

18. Belhajjame, K. Automatic Annotations of Semantic Web Services Based on Workflow Definitions / K. Belhajjame, S.M. Embury, N.W. Paton et al. // ACM Transactions on the Web. — 2008. — Vol. 2, № 2. — P. 1–34.

19. Beloglazov, A. Improving Productivity in Design and Development of Information Technology (IT) Service Delivery Simulation Models / A. Beloglazov, D. Banerjee, A. Hartman, R. Buyya // J. of Service Research. — 2015. — Vol. 18, № 1. — P. 75–89.

20. Berman, F. Adaptive Computing on the Grid Using APLeS / F. Berman, R. Wolski, H. Casanova et al. // IEEE Transactions on Parallel and Distributed Systems. — 2003. — Vol. 14, № 4. — P. 369–382.

21. Berners, T. The Semantic Web / T. Berners // Scientific American. — 2001. — Vol. 120, № 3. — P. 220–225.

22. Berriman, G.B. Montage: a Grid enabled engine for delivering custom science–grade mosaics on demand / G.B. Berriman, E. Deelman, J. Good et al. // Proc. of SPIE: Conf. of Optimizing Scientific Return for Astronomy through Information Technologies. — 2004. — Vol. 5493. Doi: 10.1117/12.550551.

23. Bharathi, S. Characterization of scientific workflows / S. Bharathi, A. Chervenak, E. Deelman et al. // Proc. of the Third Workshop on Workflows in Support of Large–Scale Science (WORKS–2008). — IEEE, 2008. Doi: 10.1109/WORKS.2008.4723958.

24. Bruzzone, A. An Application Methodology for Logistics and Transportation Scenarios Analysis and Comparison within the Retail Supply Chain / A. Bruzzone, F. Longo // European J. Industrial Engineering. — 2014. — Vol. 18, № 1. — P. 112–142.

25. Bumgardner, V.K. OpenStack in Action / V.K. Bumgardner. — Manning Publications, 2016. — 358 p.

26. Buyya, R. Economic Models for Resource Management and Scheduling in Grid Computing / R. Buyya et al. // J. of Concurrency and Computation: Practice and Experience. — 2002. — Vol. 14, № 13-15. — P. 1507–1542.

27. Buyya, R. Market–Oriented Grid and Utility Computing / Ed. by R. Buyya, K. Bubendorfer. — Hoboken: John Wiley & Sons, 2010. — 644 p.

28. Buyya, R. Mastering Cloud Computing / R. Buyya, C. Vecchiola, S.T. Selvi. — Burlington: Morgan Kaufmann, 2013. — 452 p.

29. Buyya, R. Nimrod–G Resource Broker for Service–Oriented Grid Computing / R. Buyya, D. Abramson, J. Giddy // IEEE Distributed Systems Online. — 2001. — Vol. 2, № 7. — P. 1–3.

30. Buyya, R. Nimrod–G: An architecture for a resource management and scheduling system in a global computational grid / R. Buyya, D. Abramson, J. Giddy // Proc. of the 4th Intern. Conf. / Exhibition on High Performance Computing in the Asia–Pacific Region 2000. — IEEE CS Press, 2000. — Vol. 1. — P. 283–289.

31. Buyya, R. The grid economy / R. Buyya, D. Abramson, S. Venugopal // Proc. of the IEEE. — 2005. — Vol. 93, № 3. — P. 698–714.

32. Cao, J. ARMS: An agent–based resource management system for grid computing / J. Cao, S.A. Jarvis, S. Saini, D.J. Kerbyson, G.R. Nudd // Scientific Programming. — 2002. — Vol. 10, № 2. — P. 135–148.

33. Cao, J. GridFlow: Workflow Management for Grid Computing / J. Cao, S.A. Jarvis, S. Saini, G.R. Nudd // Proc. of the 3rd Intern. Symposium on Cluster Computing and the Grid. — IEEE CS Press, 2003. — P. 198–205.

34. Cardoso, J. The Web Ontology Language (OWL) and its Applications / J. Cardoso, A.M. Pinto // Encyclopedia of Information Science and Technology. M. Khosrow–Pour, Ed. — Hershey: Information Science Reference, 2015. — P. 754–766.

35. Casanova, H. NetSolve: A Network Server for Solving Computational Science Problems / H. Casanova, J. Dongarra // Intern. J. of Supercomputer Applications and High Performance Computing. — 1997. — Vol. 11, № 3. — P. 212–223.

36. Casavant, T.L. A Taxonomy of Scheduling in General–Purpose Distributed Computing Systems / T.L. Casavant, J.G. Kuhl // IEEE Transactions on Software Engineering. — 1988. — Vol. 14, № 2. — P. 141–154.

37. Cassady, R.JR. Auctions and Auctioneering / R. JR. Cassady // University of California Press, Berkley and Los Angeles, California, 1967. — 327 p.

38. Castronova, A.M. Models as web services using the open geospatial consortium (ogc) web processing service (wps) standard / A.M. Castronova, J.L. Goodall, M.M. Elag // Environmental Modelling & Software. — 2013. — Vol. 41. — P. 72–83.

39. Chakravarti, A.J. The organic grid: self-organizing computation on a peer-to-peer network / A.J. Chakravarti, G. Baumgartner, M. Lauria // IEEE Transactions on Systems. — 2005. — Vol. 35, № 3. — P. 373–384.

40. Chapin, S.J. Benchmarks and Standards for the Evaluation of Parallel Job Schedulers / S.J. Chapin, W. Cirne, D.G. Feitelson et al. // Lecture Notes in Computer Science. — 1999. — Vol. 1659. — P. 66–89.

41. Chirkin, A.M. Execution time estimation for workflow scheduling / A.M. Chirkin, A.S. Belloum, S.V. Kovalchuk et al. // Future Generation Computer Systems. — 2017. — Vol. 75. — P. 376–387.

42. Coppinger, V.M. Incentives and behavior in English, Dutch and sealed-bid auctions / V.M. Coppinger, V.L. Smith, J. Titus // Economic inquiry. — 1980. — Vol. 18. — № 1. — P. 1–22.

43. Cramton, P. An overview of combinatorial auctions / P. Cramton, Y. Shoham, R. Steinberg // ACM SIGecom Exchanges. — 2007. — Vol. 7, № 1. — P. 3–14.

44. Cure, O. RDF Database Systems / O. Cure, G. Blin. — Waltham: Morgan Kaufmann, 2014. — 256 p.

45. Czajkowski, K. Resource Co–allocation in Computational Grids / K. Czajkowski, I. Foster, C. Kesselman // Proc. of the 8th IEEE Intern. Symposium on High–Performance Distributed Computing (HPDC–8). — IEEE, 1999. — P. 219–228.

46. De Roure, D. The Semantic Grid: Past, Present and Future / D. De Roure, N.R. Jennings, N.R. Shadbolt // Proc. of the IEEE. — 2005. — Vol. 93, № 3. — P. 669–681.

47. De Weerdt, M. Introduction to Planning in Multiagent Systems / M. de Weerdt, B.J. Clement // Multiagent and Grid Systems, special issue: Planning in multiagent systems. — 2009. — Vol. 5, № 4. — P. 345–355.

48. Deelman, E. Pegasus: a Framework for MaPing Complex Scientific Workflows onto Distributed Systems / E. Deelman, G. Singh, M. H. Su et al. // Scientific Programming. — 2005. — Vol. 13, № 3. — P. 219–237.

49. Del Val, E. Self-organization in service discovery in presence of noncooperative agents / E. del Val, M. Rebollo, V. Botti // Neurocomputing. — 2016. — Vol. 176. — P. 81–90.

50. Di, S. Ex–post efficient resource allocation for self–organizing cloud / S. Di, C.L. Wang, L. Chen // Computers and Electrical Engineering. — 2013. — Vol. 39, № 7. — P. 2342–2356.

51. Di, S. Social-optimized win-win resource allocation for self-organizing cloud / S. Di, C.L. Wang, L. Cheng, L. Chen // Proc. of the Intern. Conf. on Cloud and Service Computing 2011. — IEEE CS Press, 2011. — P. 251–258.

52. Distributed agent-based online auction system / C. Bădică et al. // Computing and Informatics. — 2015. — Vol. 33, № 3. — P. 518–552.

53. Dubois, G. eHabitat, a multi-purpose Web Processing Service for ecological modeling / M. Schulz, J. Skøien, L. Bastin et al. // Environmental Modelling & Software. — 2013. — Vol. 41. — P. 123-133.

54. Durfee, E.H. Distributed Problem Solving and Planning / E.H. Durfee // Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence / Ed. by G. Weiss. — MIT Press, 1999. — P. 121–164.

55. Edelev, A.V. Combinatorial Modeling Approach to Find Rational Ways of Energy Development with Regard to Energy Security Requirements / A.V. Edelev, I.A. Sidorov // Lecture Notes in Computer Science. — 2017. — Vol. 10187. — P. 310–317.

56. Edelev, A. The Combinatorial Modelling Approach to Study Sustainable Energy Development of Vietnam / A. Edelev et al. // Communications in Computer and Information Science. — 2017. — Vol. 793 — P. 207–218.

57. Elmroth, E. Grid Resource Brokering Algorithms Enabling Advance Reservations and Resource Selection Based on Performance Predictions / E. Elmroth, J. Tordsson // Future Generation Computer Systems. — 2008. — Vol. 24, № 6. — P. 585–593.

58. Ernst, D. Understanding the Container Ecosystem: A Taxonomy of Building Blocks for Container Lifecycle and Cluster Management / D. Ernst, D. Bermbach, S. Tai // Proc. of the 2nd Intern. Workshop on Container Technologies and Container Clouds. — IEEE, 2016. — Article No. 28. Doi: 10.1145/3147704.3147735.

59. Fahringer, T. ASKALON: a Grid Application Development and Computing Environment / T. Fahringer, R. Prodan, R. Duan et al. // Proc. of the 6th IEEE / ACM Intern. Workshop on Grid Computing. — IEEE, 2005. — P. 122–131.

60. Farahani, A. Enabling Autonomic Computing Support for the JADE Agent Platform / A. Farahani, E. Nazemi, G. Cabri, N. Capodieci // Scalable Computing: Practice and Experience. — 2017. — Vol. 18, № 1. — P. 91–103.

61. Felice, F.D. Optimization of Manufacturing System through World Class Manufacturing / F.D. Felice, A. Petrillo // IFAC–PapersOnLine. — 2015. — Vol. 48, № 3. — P. 741–746.

62. Feng, Z. Optimization of hydropower reservoirs operation balancing generation benefit and ecological requirement with parallel multi-objective genetic algorithm / Z. Feng, W. Niu, C. Cheng // Energy. — 2018. — Vol. 153. — P. 706–718.

63. Feoktistov, A. Agent-Based DevOps of Software and Hardware Resources for Digital Twins of Infrastructural Objects / R. Kostromin, A. Feoktistov // Proc. of the 4th International Conference on Future Networks and Distributed Systems (ICFNDS 2020). — ACM, 2020. — P. 1–6.

64. Feoktistov, A. Collaborative Development and Use of Scientific Applications in Orlando Tools: Integration, Delivery, and Deployment / A. Feoktistov, S. Gorsky, I. Sidorov, I. Bychkov, A. Tchernykh, A. Edelev // Communications in Computer and Information Science. — 2020. — Vol. 1087. — P. 18–32.

65. Feoktistov, A. Configurable cost-quality optimization of cloud-based VoIP / A. Tchernykh, J.M. Cortés-Mendoza, I. Bychkov, A. Feoktistov, L. Didelot, P. Bouvry, G. Radchenko, K. Borodulin // J. of Parallel and Distributed Computing. — 2018. — Vol. 133. — P. 319–336.

66. Feoktistov, A. Development of Distributed Subject–Oriented Applications for Cloud Computing through the Integration of Conceptual and Modular Programming / A. Feoktistov, R. Kostromin, I. Sidorov, S. Gorsky // Proc. of the 41th Intern. Convention on information and communication technology, electronics and microelectronics (MIPRO–2018). — Riejka: IEEE, 2018. — P. 256–261.

67. Feoktistov, A. Framework for preparing subject data in testing modules of scientific applications / E.S. Fereferov, A.G. Feoktistov, I.V. Bychkov // Proc. of the 1st Intern. Workshop on Information, Computation, and Control Systems for Distributed Environments. — CEUR-WS Proceedings. — 2018. — Vol. 2212. — P. 70–77.

68. Feoktistov, A. Integration of Heterogeneous HPC–clusters Using OpenStack Platform / A. Feoktistov, I. Sidorov, V. Sergeev, R. Kostromin, V. Bogdanova // Параллельные вычислительные технологии (ПАВТ’2017): Короткие статьи и описания плакатов XI Междунар. конф. — Челябинск: Издательский центр ЮУрГУ, 2017. — С. 90–99.

69. Feoktistov, A. Job Flow Management for Virtualized Resources of Heterogeneous Distributed Computing Environment / I. Bychkov, A. Feoktistov, I. Sidorov, R. Kostromin // Procedia Engineering. — 2017. — Vol. 201. — P. 534–542.

70. Feoktistov, A. Methods and tools for evaluating the reliability of information and computation processes in grid and cloud systems / A. Feoktistov, I. Sidorov, R. Kostromin, G. Oparin, O. Basharina // Proceedings of the 1st International Workshop on Information, Computation, and Control Systems for Distributed Environments. — CEUR-WS Proc. — 2019. — Vol. 2430. — P. 60–69.

71. Feoktistov, A. Microservice-Based Approach to Simulating Environmentally-Friendly Equipment of Infrastructure Objects Taking into Account Meteorological Data / R. Kostromin, O. Basharina, A. Feoktistov, I. Sidorov // Atmosphere. — 2021. — Vol. 12, № 9: 1217. — P. 1–24.

72. Feoktistov, A. Multi–Agent Algorithm for Re–Allocating Grid–Resources and Improving Fault–Tolerance of Problem–Solving Processes / A. Feoktistov, R. Kostromin, I. Sidorov, S. Gorsky, G. Oparin // Procedia Computer Science. — 2019. — Vol. 150. — P. 171–178.

73. Feoktistov, A. Operating cost and quality of service optimization for multi-vehicle-type timetabling for urban bus systems / D. Pena, A. Tchernykh, S. Nesmachnow, R. Massobrio, A. Feoktistov, I. Bychkov, G. Radchenko, A. Drozdov, S. Garichev // J. of Parallel and Distributed Computing. – 2018. — Vol. 133. — P. 272–285.

74. Feoktistov, A. Orlando Tools: Development, Training, and Use of Scalable Applications in Heterogeneous Distributed Computing Environments / A. Tchernykh, A. Feoktistov, S. Gorsky, I. Sidorov, R. Kostromin, I. Bychkov, O. Basharina, A. Alexandrov, R. Rivera–Rodriguez // Communications in Computer and Information Science. — 2019. — Vol. 979. — P. 265–279.

75. Feoktistov, A. Simulation Modeling in Heterogeneous Distributed Computing Environments to Support Decisions Making in Warehouse Logistics / I. Bychkov, G. Oparin, A. Tchernykh, A. Feoktistov, V. Bogdanova, Yu. Dyadkin, V. Andrukhova, O. Basharina // Procedia Engineering. — 2017. — Vol. 201. — P. 524–533.

76. Feoktistov, A. Supercomputer Engineering for Supporting Decision-making on Energy Systems Resilience / I. Bychkov, A. Feoktistov, S. Gorsky, A. Edelev, I. Sidorov, R. Kostromin, E. Fereferov, R. Fedorov // Proc. of the 14th IEEE International Conference on Application of Information and Communication Technologies. — IEEE, 2020. — P. 1–6.

77. Feoktistov, A. Tender of computational works in heterogeneous distributed environment / A. Feoktistov // Proc. of the 2nd International Workshop on Information, Computation, and Control Systems for Distributed Environments. — CEUR-WS Proc. — 2020. — Vol. 2638. — P. 99–108.

78. Feoktistov, A. Toolkit for Simulation Modeling of Logistics Warehouse in Distributed Computing Environment / I. Bychkov, G. Oparin, A. Tchernykh, A. Feoktistov, V. Bogdanova, Yu. Dyadkin, V. Andrukhova, O. Basharina // Информационные технологии и нанотехнологии: Сб. тр. III Междунар. конф. и молодежной школы. — Самара: Изд–во Самарского гос. аэрокосмического ун-та, 2017. — С. 1106–1111.

79. Feoktistov, A. Virtualization of Heterogeneous HPC–clusters Based on OpenStack Platform / A. Feoktistov, I. Sidorov, V. Sergeev, R. Kostromin, V. Bogdanova // Вестник Южно–Уральского гос. ун-та. Сер. Вычисл. математика и информатика. — 2017. — Т. 6, № 2. — С. 37–48.

80. Feoktistov, A.G. Conceptual Model of Problem–Oriented Heterogeneous Distributed Computing Environment with Multi–Agent Management / I.V. Bychkov, G.A. Oparin, A.N. Tchernykh, A.G. Feoktistov, V.G. Bogdanova, S.A. Gorsky // Procedia Computer Science. — 2017. — Vol. 103. — P. 162–167.

81. Feoktistov, A.G. Distributed Computing Environment for Vulnerability Analysis of Energy Critical Infrastructures / A.V. Edelev, I.A. Sidorov, A.G. Feoktistov // Advances in Intelligent Systems Research. — 2018. — Vol. 158. — P. 37–42.

82. Feoktistov, A.G. Logical–Probabilistic Analysis of Distributed Computing Reliability / A.G. Feoktistov, I.A. Sidorov // Proc. of the 39th Intern. Convention on information and communication technology, electronics and microelectronics (MIPRO–2016). — Riejka: IEEE, 2016. — P. 247–252.

83. Feoktistov, A. Automation of Multi-Agent Control for Complex Dynamic Systems in Heterogeneous Computational Network / G. Oparin, A. Feoktistov, V. Bogdanova, I. Sidorov // AIP Conference Proceedings. — 2017. — Vol. 1798. — P. 0201171–02011710.

84. Feoktistov, A.G. The Service-Oriented Multiagent Approach to High-Performance Scientific Computing / I.V. Bychkov, G.A. Oparin, A.G. Feoktistov, V.G. Bogdanova, I.A. Sidorov // Lecture Notes in Computer Science. — 2017. — Vol. 10187. — P. 256–263.

85. Feoktistov, A.G. Predicting runtime of computational jobs in distributed computing environment / A. Feoktistov, O.Yu. Basharina // Proc. of the 2nd International Workshop on Information, Computation, and Control Systems for Distributed Environments. — CEUR-WS Proc. — 2020. — Vol. 2638. — P. 109–117.

86. Feoktistov, A. Service-Oriented Tools for Automating Digital Twin Development / R. Kostromin, A. Feoktistov, M. Voskoboinikov // Proceedings of the 4th Scientific-practical Workshop on Information Technologies: Algorithms, Models, Systems. — CEUR-WS Proceedings. — 2021. — Vol. 2984. — P. 95–100.

87. Feoktistov, A.G. Static-dynamic algorithm for managing asynchronous computations in distributed environments / A.G. Feoktistov, S.A. Gorsky // Proceedings of the 1st International Workshop on Advanced Information and Computation Technologies and Systems. — CEUR-WS Proceedings. — 2021. — Vol. 2858. — P. 64–73.

88. Foerster, T. Establishing an OGC Web Processing Service for generalization processes [Электронный ресурс] / T. Foerster, J. Stoter // ICA workshop on Generalization and Multiple Representation, 2006. — Режим доступа: https://www.researchgate.net/publication/228690818_Establishing_an_OGC_Web_Processing_Service_for_generalization_processes/ (дата обращения: 24.09.2021).

89. Foster, I. Brain meets brawn: Why grid and agents need each other / I. Foster, N.R. Jennings, C. Kesselman // Proc. of the Third Intern. Joint Conf. on Autonomous Agents and Multiagent Systems. — IEEE, 2004. — Vol. 1. — P. 8–15.

90. Frey, J. Condor–G: A Computation Management Agent for Multi–Institutional Grids / J. Frey et al. // J. of Cluster Computing. — 2002. — Vol. 5. — P. 237–246.

91. Fujimoto, R. Parallel and Distributed Simulation / R. Fujimoto // Proc. of the 2015 Winter Simulation Conf. — IEEE Press, Piscataway, 2015. — P. 45–59.

92. Galachyants, Y.P. Sequencing of the complete genome of an araphid pennate diatom Synedra acus subsp. radians from Lake Baikal / Y.P. Galachyants et al. // Dokl. Biochem. Biophys. — 2015. — Vol. 461, № 1. — P. 84–88.

93. Gangadharan, G.R. Open Source Solutions for Cloud Computing / G.R. Gangadharan // Computer, 2017. — Vol. 50, № 1. — P. 66–70.

94. Garg, S.K. Scheduling Parallel Applications on Utility Grids: Time and Cost Trade–off Management / S.K. Garg, R. Buyya, H.J. Siegel // Proc. of the 32nd Australasian Computer Science Conf. — New South Wales: Australian Computer Society Inc., 2009. P. 151–159.

95. Gil, Y. Examining the Challenges of Scientific Workflows / Y. Gil, E. Deelman, M. Ellisman et al. // EEE Computer. — 2007. — Vol. 40, № 12. — P. 24–32.

96. Globus Toolkit [Электронный ресурс]. — Режим доступа: http://toolkit.globus.org/toolkit / (дата обращения: 24.09.2021).

97. Goecks, J. Galaxy: a comprehensive Approach for Supporting accessible, reproducible, and transparent computational research in the life sciences / J. Goecks, A. Nekrutenko, J. Taylor // Genome biology. — 2010. — Vol. 11, № 8. — P. R86. Doi: 10.1186/gb-2010-11-8-r86.

98. Google AP Engine [Электронный ресурс]. — Режим доступа: https://cloud.google.com/aPengine / (дата обращения: 24.09.2021).

99. GridWay Metascheduler [Электронный ресурс]. — Режим доступа: http://www.gridway.org (дата обращения: 24.09.2021).

100. Grosu, D. Auction–based resource allocation protocols in grids / D. Grosu, A. Das // Proc. of the 16th IASTED Intern. Conf. on Parallel and Distributed Computing and Systems. — ACTA Press, 2004. — P. 20–27.

101. Guet, F. On the reliability of the probabilistic worst–case execution time estimates / F. Guet, L. Santinelli, J. Morio // Proc. of the 8th European Congress on Embedded Real Time Software and Systems. — HAL, 2016. — P. 1–11.

102. Gupta, S. Comparison of LXD, Docker and Virtual Machine / S. Gupta, D. Gera // Intern. J. of Scientific Engineering and Research. — 2016. — Vol. 7, № 9. — P. 1414–1417.

103. Gutierrez-Garcia, J.O. Self-organizing agents for service composition in cloud computing / J.O. Gutierrez-Garcia, K.M. Sim // Proc. of the 2nd Intern. Conf. on Cloud Computing Technology and Science 2010. — IEEE CS Press, 2010. — P. 59–66.

104. Harold, E.R. XML Bible / E.R. Harold. — Foster City: IDG Books Worldwide, 1999. — 1206 p.

105. Harrison, G.W. Theory and misbehavior of first-price auctions / G.W. Harrison // The American Economic Review. — 1989. — P. 749–762.

106. Henderson, R. Job Scheduling under the Portable Batch System / R. Henderson // Lecture Notes in Computer Science. — 1995. — Vol. 949. — P. 279–294.

107. Herrera, J. Porting of Scientific Applications to Grid Computing on GridWay / J. Herrera, E. Huedo, R.S. Montero et al. // Scientific Programming. — 2005. — Vol. 13, № 4. — P. 317–331.

108. Hershberger, J. Vickrey prices and shortest paths: What is an edge worth? / J. Hershberger, S. Suri // Proc. of the 42nd IEEE Symposium on Foundations of Computer Science. — IEEE Computer Society, 2001. — P. 252–259.

109. High performance computing for energy system optimization models: Enhancing the energy policy tool kit / T. Sharma, J. Glynna, E. Panosd et al. // Energy Policy. — 2019. — Vol. 128. — P. 66–74.

110. Hirales–Carbajal, A. Workload Generation for Trace Based Grid Simulations / A. Hirales–Carbajal, J.L. González–García, A. Tchernykh // Proceeding of the 1st Intern. Supercomputer Conf. in Mexico (ISUM–2010). — Guadalajara University Publisher, 2010. — P. 1–10.

111. HTCondor [Электронный ресурс]. — Режим доступа: https://research.cs.wisc.edu/htcondor/ (дата обращения: 24.09.2021).

112. Hwang, J. A component-based performance comparison of four hypervisors / J. Hwang, S. Zeng, F. Wu, T. Wood // Proc. of the IFIP/IEEE Intern. Symposium on Integrated Network Management. — IEEE, 2013. — P. 269–276.

113. Intelligent Distributed Computing / Eds. R. Buyya, S. Thampi // Advances in Intelligent Systems and Computing. — 2015. — Vol. 321. — 300 p.

114. Iordan, V. Multi–Agent Models in Workflow Design / V. Iordan // Multi–Agent Systems — Modeling, Interactions, Simulations and Case Studies / Ed. by Dr. Faisal Alkhateeb. — InTech, 2011. — P. 131–148.

115. Ivannikov, V.P. Estimation of dynamical characteristics of a parallel program on a model / V.P. Ivannikov, S.S. Gaisaryan, A.I. Avetisyan, V.A. Padaryan // Programming and Computer Software. — 2006. — Vol. 32, № 4. — P. 203–214.

116. Jain, R. An efficient Nash–implementation mechanism for network resource allocation / R. Jain, J. Walrand // Automatica. — 2010. — Vol. 46, № 8. — P. 1276–1283.

117. Jette, M.A. Slurm: Simple Linux Utility for Resource Management / M.A. Jette, A.B. Yoo, M. Grondona // Lecture Notes in Computer Science. — 2003. — Vol. 2862. — P. 44–60.

118. Jonsson, H. Identifying Critical Components in Technical Infrastructure Networks / H. Jonsson, J. Johansson, H. Johansson // Proc. Inst. Mech. Eng., Part O: J. Risk Reliability. — Vol. 222, № 2. — 2008. — P. 235–243.

119. JSON Schema [Электронный ресурс]. —– Режим доступа: http://json-schema.org/ (дата обращения: 24.09.2021).

120. Kalyaev, A.I. Multiagent Approach for Building Distributed Adaptive Computing System / A.I. Kalyaev // Procedia Computer Science. — 2013. — Vol. 18. — P. 2193–2202.

121. Kim, H. Exploring Application and Infrastructure Adaptation on Hybrid Grid–Cloud Infrastructure / H. Kim, Y. el–Khamra, S. Jha, M. Parashar // Proc. of the 19th ACM Intern. Symposium on High Performance Distributed Computing. — 2010. — P. 402–412.

122. Kleijkers, S. A mobile multi–agent system for distributed computing / S. Kleijkers, F. Wiesman, N. Roos // Proc. of the Intern. Workshop on Agents and P2P Computing. — Springer, Berlin, Heidelberg, 2002. — P. 158–163.

123. Kovtunenko, A. Distributed Streaming Data Processing in IoT Systems Using Multi–agent Software Architecture / A. Kovtunenko, A. Bilyalov, S. Valeev // Internet of Things, Smart Spaces, and Next Generation Networks and Systems. — Lecture Notes in Computer Science. — 2018. — Vol. 11118. — P. 572–583.

124. Kozhirbayev, Z. A Performance Comparison of Container–based Technologies for the Cloud / Z. Kozhirbayev, R.O. Sinnott // Future Generation Computer Systems. — 2017. — Vol. 68. — P. 175–182.

125. Krause, L. Phylogenetic classification of short environmental DNA fragments / L. Krause et al. // Nucleic acids research. — 2008. — Vol. 36, № 7. — P. 2230–2239.

126. Kumar, A. Scalable Multiagent Planning Using Probabilistic Inference / A. Kumar, M. Toussaint, S. Zilberstein // Proc. of the 22nd Intern. Joint Conf. on Artificial Intelligence. — AAAI Press, 2011. — P. 2140–2146.

127. Kundu, P. WSDL Specification of Services for Service Oriented Architecture (SOA) Based Implementation of a CRM Process / P. Kundu, D. Das, B. Ratha. // Intern. Journal of Scientific and Engineering Research. — 2012. — Vol. 3, № 10. — P. 1–24.

128. Kurowski, K. Multicriteria Aspects of Grid Resource Management / K. Kurowski, J. Nabrzyski, A. Oleksiak, J. Węglarz // Grid Resource Management. State of the Art and Future Trends / Eds. J. Nabrzyski, J. M. Schopf, J. Weglarz. — Springer US, 2004. — P. 271–293.

129. Kwon, Y. Precise execution offloading for Applications with dynamic behavior in mobile cloud computing / Y. Kwon, H.Yi, D. Kwon et al. // Pervasive and Mobile Computing. — 2016. — Vol. 27. — P. 58–74.

130. Lan, Z. A resource mapping method in Grids based on multi-unit auction mechanism / Z. Lan, W. Dazhen // Chinese Control and Decision Conference. — IEEE, 2009. — P. 5648–5653.

131. Landa, R. Self–Tuning Service Provisioning for Decentralized Cloud Applications / R. Landa et al. // IEEE Transactions on Network and Service Management. — 2016. — Vol. 13, № 2. — P. 197–211.

132. Laszewski, von G. GridAnt: A ClientControllable Grid Workflow System / G. Von Laszewski, K. Amin, M. Hategan et al. // Proc. of the 37th Annual Hawaii Intern. Conf. on System Sciences. — Washington: IEEE Computer Society, 2004. — P. 1–10.

133. Laszewski, von G. Java CoG Kit Karajan / Gridant workflow guide. Technical Report / Von G. Laszewski, M. Hategan. — Argonne National Laboratory, Argonne, Ill, USA, 2005. — 50 p.

134. Laxmi, CH.V.T.E.V. Application Level Scheduling (APLeS) in Grid with Quality of Service (QoS) / CH.V.T.E.V. Laxmi, K. Somasundaram // Intern. Journal of Grid Computing and Applications. — 2014. — Vol. 5, № 2. — P. 1–10.

135. Leitao, P. Parallelising Multi–agent Systems for High Performance Computing / P. Leitao, U. Inden, C.–P. Ruckemann // Proc. of the 3rd Intern. Conf. on Advanced Communications and Computation. — IARIA Publ., 2013. — P. 1–6.

136. Lin, W.Y. Dynamic auction mechanism for cloud resource allocation / W.Y. Lin, G.Y. Lin, H.Y. Wei // Proc. of the10th IEEE / ACM Intern. Conf. on Cluster, Cloud and Grid Computing (CCGrid). — IEEE Computer Society, 2010. — P. 591–592.

137. Litzkow, M. Condor — A Hunter of Idle Workstations / M. Litzkow, M. Livny, M. Mutka // Proc. of the 8th Intern. Conf. of Distributed Computing Systems (ICDCS). — Los Alamitos: IEEE CS Press, 1988. — P. 104–111.

138. Livny, J. High–throughput, kingdom–wide prediction and annotation of bacterial non–coding RNAs / J. Livny, H. Teonadi, M. Livny, M.K. Waldor // PLoS One 3. — 2008. — Vol. 3, № 9. — P. e3197. Doi: 10.1371/journal.pone.0003197.

139. Livny, M. High Throughput Resource Management / M. Livny, R. Raman // The Grid: Blueprint for a New Computing Infrastructure / Eds. I. Foster, C. Kesselman. — San Francisco: Morgan Kaufmann, 1999. — P. 311–337.

140. Luft, G. Energy Security Challenges for the 21st Century: A Reference Handbook / Eds. G. Luft and A. Korin. — Praeger, 2009. — 372 p.

141. Madni, S.H.H. Recent advancements in resource allocation techniques for cloud computing environment: a systematic review / S.H.H. Madni, M.S.A. Latiff, Y. Coulibaly // Cluster Comput. — 2017. — Vol. 20, № 3. — P. 2489–2533.

142. Madureira, A. Negotiation mechanism for self-organized scheduling system with collective intelligence / A. Madureira, I. Pereira, P. Pereira, A. Abraham // Neurocomputing. — 2014. — V. 132. — P. 97–110.

143. Maechling, P. SCEC CyberShake workflows—automating probabilistic seismic hazard analysis calculations / P. Maechling et al. // E.D.I. Taylor, D. Gannon, M. Shields (Eds.), Workflows for e–Science, Springer, 2006. Doi: 10.1007/978-1-84628-757-2_10.

144. Maeno, T. PanDA: distributed production and distributed analysis system for ATLAS / T. Maeno // J. of Physics: Conf. Series. — IOP Publishing, 2008. — Vol. 119, № 6. — P. 062036. Doi: 10.1088/1742-6596/119/6/062036.

145. Manolescu, D.A. Micro–workflow: a workflow architecture Supporting compositional object–oriented software development / D.A. Manolescu. PhD Thesis. — University of Illinois at Urbana–Champaign, 2001. — 221 p.

146. Mariotti, M. Strategies and systems towards grids and clouds integration: A DBMS–based solution / M. Mariotti, O. Gervasi, F. Vella, A. Cuzzocrea, A. Costantini // Future Generation Computer Systems. — 2018. — Vol. 88. — P. 718-729.

147. Mateescu, G. Hybrid computing — where HPC meets grid and cloud computing / G. Mateescu, W. Gentzsch, C.J. Ribbens // Future Gener. ComP. Sy. — 2011. — Vol. 27, № 5. — P. 440–453.

148. McAfee, R.P. Auctions and bidding / R.P. McAfee, J. McMillan // J. of economic literature. — 1987. — Vol. 25. — № 2. — P. 699–738.

149. Microsoft Windows Azure [Электронный ресурс]. — Режим доступа: https://azure.microsoft.com/ru–ru / (дата обращения: 24.09.2021).

150. Milgrom, P.R. A theory of auctions and competitive bidding / P.R. Milgrom, R.J. Weber // Econometrica: Journal of the Econometric Society. — 1982. — P. 1089–1122.

151. Mladen, A. Integration of High–Performance Computing into Cloud Computing Services / A. Mladen, S. Eric, D. Patrick // Handbook of Cloud Computing / Eds. B. Furht, A. Escalante. — N.Y.: Springer, 2010. — Vol. 3. — P. 255–276.

152. Mutz, A. Eliciting Honest Value Information in a Batch–queue Environment / A. Mutz, R. Wolski, J. Brevik // Proc. of the 8th IEEE / ACM Intern. Conf. on Grid Computing. — IEEE CS Press, 2007. — P. 291–297.

153. Nash, E. Applications of open geospatial web services in precision agriculture: a review / E. Nash, P. Korduan, R. Bill // Precision agriculture. — 2009. — Vol. 10, № 6. — P. 546–560.

154. Nash, J. Equilibrium points in n–person games // Proc. of the National Academy of Sciences of the United States of America. — 1950. — Vol. 36, № 1. — P. 48–49.

155. Natrajan, A. Grid Resource Management in Legion / A. Natrajan, M.A. Humphrey, A.S. Grimshaw // Grid Resource Management. State of the Art and Future Trends / Eds. J. Nabrzyski, J.M. Schopf, J. Weglarz. — Springer US, 2004. — P. 145–160.

156. Neeman, Z. The effectiveness of English auctions / Z. Neeman // Games and Economic Behavior. – 2003. — Vol. 43. — № 2. — P. 214–238.

157. Nejad, M. Truthful greedy mechanisms for dynamic virtual machine provisioning and allocation in clouds / M.M. Nejad, L. Mashayekhy, D. Grosu // IEEE transactions on parallel and distributed systems. — 2015. — Vol. 26, № 2. — P. 594–603.

158. Nikol, G., Service–Oriented Multi–tenancy (SO–MT): Enabling Multi–tenancy for Existing Service Composition Engines with Docker / G. Nikol, M. Trager, S. Harrer, G. Wirtz // Proc. of the IEEE Symposium on Service–Oriented System Engineering (SOSE–2016). — IEEE, 2016. — P. 238–243.

159. Nwana, H.S. Software Agents: An Overview / H.S. Nwana // Knowledge Engineering Review. — 1996. — Vol. 11, № 3. — P. 205–244.

160. Oinn, T. Taverna: a Tool for the Composition and Enactment of Bioinformatics Workflows / T. Oinn, M. Addis, J. Ferris et al. // Bioinformatics. — 2004. — Vol. 20, № 17. — P. 3045–3054.

161. Oleinikova, S.A. Mathematical and Software of the Distributed Computing System Work Planning on the Multiagent Approach Basis / S.A. Oleinikova et al. // Intern. J. of APlied Engineering Research. — 2016. — Vol. 11, № 4. — P. 2872–2878.

162. OProfile – A System Profiler for Linux — 2019 [Электронный ресурс]. — Режим доступа: https://oprofile.sourceforge.io/news/ (дата обращения: 24.09.2021).

163. Pederson, P. Critical infrastructure interdependency modeling: a survey of US and Intern. research / P. Pederson, D. Dudenhoeffer, S. Hartley, M. Permann // Idaho National Laboratory. — 2006. — Vol. 25. — P. 27.

164. Pekeč, A. Combinatorial auction design / A. Pekeč, M.H. Rothkopf // Management Science. — 2003. — Vol. 49. — № 11. — P. 1485–1503.

165. Plauth, M. A performance survey of lightweight virtualization techniques / M. Plauth, L. Feinbube, A. Polze // Lecture Notes Computer. Science. — 2017. — Vol. 10465. — P. 34–48.

166. Potapov, V.V. Identification of biological targets for virtual screening of inhibitors of replication of tick–borne encephalitis virus / V.V. Potapov et al. // Abstracts of the Eighth Intern. Conf. on Bioinformatics of Genome Regulation and Structure\Systems Biology. — Novosibirsk: Institute of Cytology and Genetics SB RUS, 2012. — P. 249.

167. Pozdnyak, E.I. The techniques and tools for the solving bioinformatics tasks in the distributed computing systems / E.I. Pozdnyak et al. // Abstracts of the Eighth Intern. Conf. on Bioinformatics of Genome Regulation and Structure\Systems Biology. — Novosibirsk: Institute of Cytology and Genetics SB RUS, 2012. — P. 253.

168. Prasad, A.S. A mechanism design Approach to resource procurement in cloud computing / A.S. Prasad, S. Rao // IEEE Transactions on Computers. — 2014. — Vol. 63, № 1. — P. 17–30.

169. Qin, H. Subject Oriented Autonomic Cloud Data Center Networks Model / H. Qin, L. Zhu // J. of Data Analysis and Information Processing. — 2017. — Vol. 5, № 3. — P. 87–95.

170. Qiu, L. Self-Organization Mechanisms for Service Composition in Cloud Computing / L. Oiu // Intern. Journal of Hybrid Information Technology. — 2014. — Vol. 7, № 2. — P. 321–330.

171. Qureshi, M.B. Survey on Grid Resource Allocation Mechanisms / M.B. Qureshi et al. // J. of Grid Computing. — 2014. — Vol. 12, № 2. — P. 399–441.

172. ReactOS Project [Электронный ресурс]. — Режим доступа: https://reactos.org (дата обращения: 24.09.2021).

173. Rezaee, A. A Multi–Agent Architecture for QoS Support in Grid Environment / A. Rezaee et al. // J. of Computer Science. — 2008. — Vol. 4, № 3. — P. 225–231.

174. Rinaldi, S. Idenlifying, Understanding, and Analyzing Critical Infrastructure lnterdependencies / S. Rinaldi, J. Peerenboom, T. Kelly // IEEE Control Systems Magazine. — IEEE, 2001. — Vol. 21, № 6. — P. 11–25.

175. Rings, T. Grid and cloud computing: oportunities for integration with the next generation network / T. Rings, G. Caryer, J. Gallop, J. Grabowski, T. Kovacikova, S. Schulz, I. Stokes–Rees // J. of Grid Computing. — 2009. — Vol. 7, № 3. — Article no. 375. — P. 1–19.

176. Robinson, S. Conceptual Modeling for Simulation / S. Robinson // Proc. of the 2014 Winter Simulation Conf. — Piscataway: IEEE Press, 2013. — P. 377–388.

177. Rodriguez, A. Algorithms for Dynamic Scheduling of Unit Execution Time Tasks / A. Rodriguez, A. Tchernykh, K. Ecker // European J. of Operational Research. — 2003. — Vol. 146, № 2. — P. 403–416.

178. Ross, D. Doug Ross Talks about Structured Analysis / D. Ross. — IEEE Computer, 1985. — 230 p.

179. Sandholm, T. Issues in computational Vickrey auctions / T. Sandholm // Intern. J. of Electronic Commerce. — 2000. — Vol. 4, № 3. — P. 107–129.

180. Schriber, T.J. Inside Discrete–Event Simulation Software: How It Works and Why It Matters / T.J. Schriber, D.T. Brunner, J.S. Smith // Proc. of the 2014 Winter Simulation Conf. — Piscataway: IEEE Press, 2014. — P. 132–146.

181. Self-organizing agent communities for autonomic resource management / M. Jacyno, S. Bullock, N. Geard et al. // Adaptive Behavio. — 2013. — Vol. 21, № 1. — P. 3–28.

182. Self-organizing resource allocation for autonomic network / T. Eymann, M. Reinickke, O. Ardaiz et al. // Proc. of the 14th International Workshop on Database and Expert Systems Applications 2003. — IEEE CS Press, 2003. — P. 656–660.

183. Serrano, N. Infrastructure as a Service and Cloud Technologies / N. Serrano, G. Gallardo, J. Hernantes // IEEE Software. — 2015. — Vol. 32, № 2. — P. 30–36.

184. Serugendo, G.D.M. Self-organization in multi-agent systems / G.D.M. Serugendo, M.P. Gleizes, A. Karageorgos // The Knowledge Engineering Review. — 2005. — Vol. 20, № 2. — P. 165–189.

185. Shi, Z. Advanced Artificial Intelligence / Z. Shi. — Hackensack: World scientific, 2011. — 624 p.

186. Singh, A. A novel agent based autonomous and service composition framework for cost optimization of resource provisioning in cloud computing / A. Singh, D. Juneja, M. Malhotra // J. of King Saud University — Computer and Information Sciences. — 2015. — Vol. 29, № 1. — P. 19–28.

187. Singh, A. Agent Based Framework for Scalability in Cloud Computing / A. Singh, M. Malhotra // Intern. J. of Computer Science and Engineering Technology. — 2012. — Vol. 3, № 4. — P. 41–45.

188. Singh, S. QoS–aware autonomic resource management in cloud computing: a systematic review / S. Singh, I. Chana // ACM Computing Surveys. — 2016. — Vol. 48, № 3. — Article No. 42. Doi: 10.1145/2843889.

189. Sloot, P.M.A. Computational e–Science: Studying complex systems in silico [Электронный ресурс] / P.M.A. Sloot, D. Frenkel, H.A. Van der Vorst et al. // A National Coordinated Initiative. White Paper, February 2007. [Электронный ресурс]. — Режим доступа: http://www.science.uva.nl/research/scs/papers/archive/Sloot2007a.pdf (дата обращения: 24.09.2021).

190. Slurm Workload Manager [Электронный ресурс]. — Режим доступа: http://slurm.net/ (дата обращения: 24.09.2021).

191. Smith, B. / B. Smith. Beginning JSON. —– Apress: New York, NY, USA, 2015. —– 324 p.

192. Spruth, I.W.G. Discovering and Classifying Regions in Workflow Graphs / I.W.G. Spruth. Diploma thesis in computer science. — Publisher of the University of Tubingen, 2005. — 60 p.

193. Streit, A. UNICORE: Getting to the heart of Grid technologies / A. Streit // eStrategies. — 2009. — Vol. 3. — P. 8–9.

194. Sukhoroslov, O.V. Development of distributed computing applications and services with Everest cloud platform / O.V. Sukhoroslov, A.O. Rubtsov, S.Y. Volkov // Компьютерные исследования и моделирование. — 2015. — Т. 7, № 3. — С. 593–599.

195. Sulistio, A. A Taxonomy of Computer–Based Simulations and Its MaPing to Parallel and Distributed Systems Simulation Tools / A. Sulistio // Software: Practice and Experience. — 2004. — Vol. 34, № 7. — P. 653–673.

196. Talia, D. Cloud Computing and Software Agents: Towards Cloud Intelligent Services / D. Talia // Proc. of the 12th Workshop on Objects and Agents. CEUR-WS Proc., 2011. — Vol. 741. — P. 2–6.

197. Talia, D. Workflow Systems for Science: Concepts and Tools [Электронный ресурс] / D. Talia // ISRN Software Engineering. — 2013. — Режим доступа: https://www.hindawi.com/J.s/isrn/2013/404525/ (дата обращения: 24.09.2021).

198. Tan, X. Building an elastic parallel OGC web processing service on a cloud-based cluster: A case study of remote sensing data processing service / X. Tan, L. Di, M. Deng et al. // Sustainability. — 2015. — Vol. 7, № 10. — P. 14245–14258.

199. Tanaka, M. Strategy–proof pricing for cloud service composition / M. Tanaka, Y. Murakami // IEEE Transactions on Cloud Computing. — 2016. — Vol. 4, № 3. — P. 363–375.

200. Tannenbaum, T. Condor — A Distributed Job Scheduler / T. Tannenbaum, D. Wright, K. Miller, M. Livny // Beowulf Cluster Computing with Linux / Eds. T.L. Sterling, W. GroP, E. Lusk. — MA, USA: The MIT Press, 2002. — P. 307–350.

201. Tao, J. A Note on New Trends in Data–Aware Scheduling and Resource Provisioning in Modern HPC Systems / J. Tao, J. Kolodziej, R. Ranjan et al. // Future Generation Computer Systems. — 2015. — Vol. 51, № C. — P. 45–46.

202. Taylor, I. Applications Within Grid Computing and Peer–to–peer Environments / I. Taylor, M. Shields, I. Wang, O. Rana // J. of Grid Computing. — 2004. — Vol. 1. — P. 199–217.

203. Teodoro, G. Application performance analysis and efficient execution on systems with multi–core CPUs, GPUs and MICs: a case study with microscopy image analysis / G. Teodoro, T. Kurc, G. Andrade et al. // The Intern. J. of high performance computing Applications. — 2017. — Vol. 31, № 1. — P. 32–51.

204. The Opa Language: Rapid and Secure Web Development [Электронный ресурс]. Режим доступа: http://opalang.org/ (дата обращения: 24.09.2021).

205. Toporkov, V. Anticipation Scheduling in Grid with Stakeholders Preferences / V. Toporkov, D. Yemelyanov, A. Toporkova // Russian Supercomputing Days. Springer, Cham, 2017. — P. 482–493.

206. TORQUE Resource manager [Электронный ресурс]. — Режим доступа: http://www.adaptivecomputing.com/products/torque / (дата обращения: 24.09.2021).

207. USC Epigenome Center [Электронный ресурс]. — Режим доступа: http://epigenome.usc.edu (дата обращения: 24.09.2021).

208. Vecchiola, C. Deadline–driven provisioning of resources for scientific Applications in hybrid clouds with Aneka / C. Vecchiola, R.N. Calheiros, D. Karunamoorthy, R. Buyya // Future Gener. ComP. Sy. — 2012. — Vol. 28, № 1. — P. 58–65.

209. Venkataraman, S. Ernest: Efficient Performance Prediction for Large–Scale Advanced Analytics / S. Venkataraman, Z. Yang, M.J. Franklin // Proc. of the 13th USENIX Symposium on Networked Systems Design and Implementation. — USENIX Association, 2016. — P. 363–378.

210. Vickrey, W. Counterspeculation, Auctions, and Competitive Sealed Tenders / W. Vickrey // The J. of Finance. — 1961. — Vol. 16, № 1. — P. 8–37.

211. Vidyarthi, D.P. Scheduling in Distributed Computing Systems: Analysis, Design and Models / D.P. Vidyarthi, B.K. Sarker, A.K. Tripathi, L.T. Yang. — Springer Science+Business Media, 2009. — 300 p.

212. Vorobev, S. Analysis of the Importance of Critical Objects of the Gas Industry with the Method of Determining Critical Elements in Networks of Technical Infrastructures / S. Vorobev, A. Edelev // Proc. of the 10th Intern. Conf. on Management of Large-Scale System Development (MLSD). — IEEE, 2017. Doi: 10.1109 / MLSD.2017.8109707.

213. Walsh, A. UDDI, SOAP, and WSDL: The Web Services Specification Reference Book / A. Walsh. —Pearson Education, 2002. — 305 p.

214. Wang, W. Profiling program behavior for anomaly intrusion detection based on the transition and frequency property of computer audit data / W. Wang, X. Guan et al. // Computers and security. — 2006. — Vol. 25, № 7. — P. 539–550.

215. Webber, J. REST in practice / J. Webber // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). — 2010. — Vol. 6285. — P. 7.

216. Wilhelm, R. The worst–case execution–time problem — overview of methods and survey of tools / R. Wilhelm, J. Engblom, A. Ermedahlet al. // ACM Transactions on Embedded Computing Systems. — 2008. — Vol. 7, № 3. — P.c1–52.

217. Winzer, C. Conceptualizing energy security / C. Winzer // Energy policy. — 2012. — Vol. 46. —P. 36–48.

218. Wirth, N. Extended Backus–naur Form (EBNF) / N. Wirth. // ISO / IEC, 1996. — Vol. 14977. — P. 2996.

219. Wooldridge, M. Intelligent agents: Theory and practice / M. Wooldridge, N.R. Jennings // The knowledge engineering review. — 1995. — Vol. 10, № 2. — P. 115–152.

220. XML Schema [Электронный ресурс]. —– Режим доступа: https://www.w3.org/TR/xmlschema-0/ (дата обращения: 24.09.2021).

221. YarKhan, A. GridSolve: The Evolution of a Network Enabled Solver / A. YarKhan, J. Dongarra, K. Seymour // Proc. of the IFIP TC2 / WG2.5 Working Conf. on Grid–Based Problem Solving Environments. — Springer, 2007. — P. 215–224.

222. Yu, J. A Taxonomy of Workflow Management Systems for Grid Computing / J. Yu, R. Buyya // J. of Grid Computing. — 2005. — Vol. 3, № 3–4. — P. 171–200.

223. Zhang, Y. Contrastive analysis of three parallel modes in multi-dimensional dynamic programming and its application in cascade reservoirs operation / Y. Zhang, Z. Jiang, C. Ji, P. Sun // J. of Hydrology. — 2015. — Vol. 529. — P. 22–34.

224. Zhang, Y. Auction–based resource allocation in cognitive radio systems / Y. Zhang et al. // IEEE Communications Magazine. — 2012. — Vol. 50, № 11. — P. 108–120.

225. Zheng, G. Simulation–based performance prediction for large parallel machines / G. Zheng, T. Wilmarth, P. Jagadishprasad, L.V. Kale // Intern. J. of Parallel Programming. — 2005. — Vol. 33, № 2–3. — P. 183–207.

226. Афанасьев, А.П. Интеграция инструментария IAR–net с технологией ICE / А.П. Афанасьев, В.В. Волошинов, О.В. Сухорослов // Информ. технологии и вычисл. системы. — 2008. — С. 38–50.

227. Бабаев, И.О. СПОРА — система программирования с автоматическим синтезом программ / И.О. Бабаев, С.С. Лавров, Г.А. Нецветаева и др. // Применение методов математической логики. — Таллин: Изд–во Института кибернетики АН ЭССР, 1983. — С. 29–41.

228. Баглыков, А.Н. Метод прогнозирования времени выполнения программ / А.Н. Баглыков // Современная наука: теоретический и практический взгляд: Сб. статей Междунар. науч.–практ. конф. — Уфа: АЭТЕРНА, 2015. — Ч. 1. — С. 3–6.

229. Бандман, О.Л. Мелкозернистый параллелизм в вычислительной математике / О.Л. Бандман // Программирование. — 2001. — № 4. — С. 1–18.

230. Баранов, А.В., Тихомиров А.И. Применение закрытого аукциона первой цены в территориально распределенной системе с абсолютными приоритетами // ИТНОУ: информационные технологии в науке, образовании и управлении. — 2017. — № 3 (3). — С. 62–71.

231. Бахвалов, Н.С. Численные методы / Н. С. Бахвалов. — М.: Наука, 1973. — Т. 1. — 632 с.

232. Башарина, О.Ю. Методика анализа, оценки и прогнозирования динамики основных показателей функционирования складского логистического комплекса / О.Ю. Башарина, С.И. Носков // Фундаментальные исследования. — 2013. — № 11–6. — С. 1103–1107.

233. Башарина, О.Ю. Моделирование торгово–складского комплекса в распределенной вычислительной среде / О.Ю. Башарина, А.В. Ларина, Н.Г. Суханова, А.Г. Феоктистов // Моделирование. Теория, методы и средства: Материалы IV Междунар. науч.–практ. конф. — Новочеркасск: ЮРГТУ, 2006. — Ч. 3. — С. 28–32.

234. Башарина, О.Ю. Пакет моделирования складской логистики: разработка и комплексирование в Orlando Tools / О.Ю. Башарина, С.А. Горский // Программные продукты и системы. — 2012. — № 1. — С. 89–91.

235. Башарина, О.Ю. Решение задач складской логистики на основе применения методологии системного анализа / О.Ю. Башарина, С.И. Носков // Современные технологии. Системный анализ. Моделирование. — 2014. — № 1. — С. 70–74.

236. Беломестных, Т.В. Поиск Tc1–подобных ДНК транспозонов в транскриптомах байкальских сиговых рыб / Т.В. Беломестных и др. // Тез. док. V Междунар. школs молодых ученых по молекулярной генетике “Непостоянство генома”. — М.: Мегакаталог, 2012. — С. 12.

237. Богомолов, И.В. Проблемы масштабируемости облачных сред и поиск причин деградации центрального сервиса идентификации OpenStack Keystone / И.В. Богомолов, А.В. Алексиянц, О.Д. Борисенко, А.И. Аветисян // Известия ЮФУ. Техн. серия. — 2016. — № 12. — С. 130–140.

238. Боев, В.Д. Моделирование систем. Инструментальные средства GPSS World / В.Д. Боев. — СПб.: БХВ–Петербург, 2004. — 368 с.

239. Борисенко, О.Д. Создание виртуальных кластеров Apache Spark в облачных средах с использованием систем оркестрации / О.Д. Борисенко, Р.К. Пастухов, С.Д. Кузнецов // Тр. Института системного программирования РАН. — 2016. — Т. 28, вып. 6. — С. 111–120.

240. Бухановский, А.В. Интеллектуальные программные комплексы компьютерного моделирования сложных систем: концепция, архитектура и примеры реализации / А.В. Бухановский, С.В. Ковальчук, С.В. Марьин // Известия высших учебных заведений. Приборостроение. — 2009. — Т. 52, № 10. — С. 5–24.

241. Бухановский, А.В. Информационно–аналитический обзор по критической технологии «Технологии и программное обеспечение высокопроизводительных распределенных вычислительных систем: технологические тренды, приоритетные направления, перспективы развития, основные организации, оценка рынков, сопоставление российских и мировых результатов» // А.В. Бухановский — СПб.: НИУ ИТМО, 2013. — 31 с.

242. Бухановский, А.В. Онтология интеграции знаний на основе технологии iPSE в интеллектуальной среде «Облачной» модели / А.В. Бухановский, С.В. Иванов, С.В. Ковальчук, Ю.И. Нечаев // Онтология проектирования. — 2012. — № 4 (6). — С. 18–27.

243. Буч, Г. Язык UML: Руководство пользователя / Г. Буч, Дж. Рамбо, А. Джекобсон. — М.: ДМК, 2000. — 496 с.

244. Бычков, И.В. Инфраструктурный подход к обработке пространственных данных в задачах управления территориальным развитием / И.В. Бычков, Г.М. Ружников, В.В. Парамонов и др. // Вычислительные технологии. — 2018. — Т. 23, № 4. — С. 15-31.

245. Вальковский, В.А. Синтез параллельных программ и систем на вычислительных моделях / В.А. Вальковский, В.Э. Малышкин. — Новосибирск: Наука, 1988. — 129 с.

246. Васюков, А.В. Использование контейнеров в инфраструктуре разработки, запуска и автоматического тестирования научных приложений / А.В. Васюков, А.С. Ермаков, К.А. Беклемышева // Материалы 4–й Всерос. науч.–техн. конф. «Суперкомпьютерные технологии». — Ростов н / Д: Изд–во ЮФУ, 2016. — Т. 2. — С. 141–143.

247. Велихов, В.Е. Интеграция гетерогенных вычислительных мощностей НИЦ «Курчатовский институт» для проведения масштабных научных исследований / В.Е. Велихов и др. // Известия ЮФУ. Техническая серия. — 2016. — № 11. — С. 88–100.

248. Воеводин, В.В. Параллельные вычисления / В.В. Воеводин, Вл.В. Воеводин. — СПб.: БХВ–Петербург, 2002. — 608 с.

249. Воеводин, В.В. Решение больших задач в распределенных вычислительных средах / В.В. Воеводин // Автоматика и телемеханика. — 2007. — № 5. — С. 32–45.

250. Гаврилова, Т.А. Базы знаний интеллектуальных систем / Т.А. Гаврилова, В.Ф. Хорошевский. — СПб.: Питер, 2000. — 384 с.

251. Гаджинский, А.М. Современный склад. Организация, технологии, управление и логистика / А.М. Гаджинский. — М.: ТК Велби, Изд–во Проспект, 2005. — 176 с.

252. Горбунов–Посадов, М.М. Расширяемые программы / М.М. Горбунов–Посадов. — М.: Полиптих, 1999. — 336 с.

253. Горбунов–Посадов, М.М. Системное обеспечение пакетов прикладных программ / М.М. Горбунов–Посадов, Д.А. Корягин, В.В. Мартынюк. — М.: Наука, 1990. — 208 с.

254. Городецкий, В.И. Самоорганизация и многоагентные системы I. Модели многоагентной самоорганизации / В. И. Городецкий // Известия РАН. Теория и системы управления. — 2012. — № 2. — С. 92–120.

255. Городецкий, В.И. Самоорганизация и многоагентные системы. II. Приложения и технология разработки / В.И. Городецкий // Известия РАН. Теория и системы управления. — 2012. — № 3. — С. 55–74.

256. Дыбская, В.В. Логистика для практиков: Эффективные решения в складировании и грузопереработке / В.В. Дыбская. — М.: ВИНИТИ РАН, 2002. — 264 с.

257. Дядькин, Ю.А. Инструментальный комплекс имитационного моделирования разнородной распределенной вычислительной среды / Ю.А. Дядькин, Е.С. Фереферов // Вычислительные технологии. — 2016. — Т. 21, № 3. — С. 18–32.

258. Еделев, А.В. Применение распределенных вычислений для выявления критически важных объектов газотранспортной сети России / А.В. Еделев, С.М. Сендеров, И.А. Сидоров // Информационные и математические технологии в науке и управлении. — 2016. — № 1 (27). — С. 55–62.

259. Еделев, А.В. Формирование вариантов развития энергетики Вьетнама методами комбинаторного моделирования / А.В. Еделев и др. // Программные продукты и системы. — 2017. — Т. 30, № 2. — С. 172–179.

260. Ершов, А.П. Пакеты программ как методология решения прикладных проблем / А.П. Ершов, В.П. Ильин // Пакеты прикладных программ: Проблемы и перспективы. — М.: Наука, 1982. — С. 4–18.

261. Жоголев, Е.А. Технологические основы модульного программирования / Е.А. Жоголев // Программирование. — 1980. — № 2. — С. 44–49.

262. Зорин, Д.А. Алгоритм синтеза архитектуры вычислительной системы реального времени с учетом требований к надежности / Д.А. Зорин, В.А. Костенко // Известия РАН. Теория и системы управления. — 2012. — № 3. — С. 76–83.

263. Иванов, В.В. Методы вычислений на ЭВМ: Справочное пособие / В.В. Иванов. — Киев: Наукова думка, 1986. — 584 с.

264. Иерархическое моделирование систем энергетики / под. ред. Н.И. Воропая, В.А. Стенникова. — Новосибирск: Академическое изд-во «Гео», 2020. — 314 с.

265. Ильин, В.П. Вопросы технологии пакетов программ для задач математической физики / В.П. Ильин // Разработка пакетов прикладных программ. — Новосибирск: Наука, 1982. — С. 113–129.

266. Калашян, А.Н. Структурные модели бизнеса: DFD–технологии / А.Н. Калашян, Г.Н. Калянов. — М.: Финансы и статистика, 2003. — 256 с.

267. Каляев, И.А. Самоорганизация в мультиагентных системах / И.А. Каляев, А.Р. Гайдук, С.Г. Капустян // Известия Южного федерального ун-та. Техн. науки. — 2010. — Т. 104, № 3. — С. 14–20.

268. Каляев, И.А. Самоорганизующиеся распределенные системы управления группами интеллектуальных роботов, построенные на основе сетевой модели / И.А. Каляев, А.Р. Гайдук, С.Г. Капустян // Управление большими системами. — 2011. — № 30–1. — С. 605–639.

269. Кельтон, В. Имитационное моделирование / В. Кельтон, А. Лоу. — СПб.: Питер; Киев: BHV, 2004. — 847 с.

270. Киселев, А. Управление метакомпьютерными системами / А. Киселев, В. Корнеев, Д. Семенов, И. Сахаров // Открытые системы. — 2005. — № 2. — С. 11–16.

271. Коваленко, В.Н. Комплексное программное обеспечение грида вычислительного типа / В.Н. Коваленко // Препринты ИПМ им. М.В. Келдыша РАН. — 2007. — № 10. — 39 с.

272. Коваленко, В.Н. Основные положения метода опережающего планирования для Грид вычислительного типа / В.Н. Коваленко, Е.И. Коваленко, Д.А. Корягин, Э.З. Любимский // Вестник Самарского гос. ун–та. Естественнонаучная серия. — 2006. — № 4. — С. 238–264.

273. Когаловский, М.Р. Концептуальное и онтологическое моделирование в информационных системах / М.Р. Когаловский, Л.А. Калиниченко // Программирование. — 2009. — Т. 35, № 5. — С. 3–25.

274. Козлов, Н.И. Организация вычислительных работ / Н.И. Козлов. — М.: Наука, 1981. — 240 с.

275. Кондратьев, А. Современные тенденции в исследовании критической инфраструктуры в зарубежных странах / А. Кондратьев // Зарубежное военное обозрение. — 2012. — № 1. — С. 19–30.

276. Конноли, Т. Базы данных: проектирование, реализация и сопровождение. Теория и практика / Т. Конноли, К. Бегг. — М.: Вильямс, 2003. — 1440 с.

277. Коновалов, М.Г. Управление заданиями в гетерогенных вычислительных системах / М.Г. Коновалов, Ю.Е. Малашенко, И.А. Назарова // Известия РАН. Теория и системы управления. — 2011. — № 2. — С. 43–61.

278. Кононенко, О. Анализ финансовой отчетности / О. Кононенко, О. Маханько. — Харьков: Фактор, 2006. — 200 с.

279. Корсуков, А.С. Оценка надежности функционирования интегрированной кластерной системы с метапланировщиком Gridway / А.С. Корсуков // Программные продукты и системы. — 2013. — № 1. — С. 67–69.

280. Крюков, А.П. Основные подходы к построению грид-инфраструктуры национальной нанотехнологической сети / А.П. Крюков, А.П. Демичев, В.А.

Ильин, Л.В. Шамардин // Механика, управление и информатика. — 2011. № 5, — С. 51–68.

281. Липаев, В.В. Технология сборочного программирования / В.В. Липаев, А.А. Штрик. — М.: Радио и связь, 1992. — 272 с.

282. Лорьер, Ж.–Л. Системы искусственного интеллекта / Ж.–Л. Лорьер. — М.: Мир. — 1991. — 568 с.

283. Майерс, Г. Надежность программного обеспечения. — М.: Мир, 1980. — 360 с.

284. Марчук, Г.И. Методы вычислительной математики / Г.И. Марчук. — М.: Наука, 1980. — 536 с.

285. Массель, Л.В. Киберопасность как одна из стратегических угроз энергетической безопасности России / Л.В. Массель, Н.И. Воропай, С.М. Сендеров, А.Г. Массель // Вопросы кибербезопасности. — 2016. — № 4 (17).

286. Массель, Л.В. Разработка многоагентных систем распределенного решения энергетических задач с использованием агентных сценариев / Л.В. Массель, В.И. Гальперов // Известия Томского политехнического университета, инжиниринг георесурсов. — 2015. — Т. 5. — С. 45–53.

287. Можаев, А.С. Современное состояние и некоторые направления развития логико–вероятностных методов анализа систем / А.С. Можаев // Теория и информационная технология моделирования безопасности сложных систем. — Препринт 101. — СПб.: Изд–во ИПМАШ РАН, 1994. — С. 23–53.

288. Николенко, С.И. Теория экономических механизмов / С.И. Николенко. — М.: Интуит.ру; Бином. Лаборатория знаний, 2009. — 207 с.

289. Новиков, Д.А. Иерархические модели управления / Д.А. Новиков // Материалы пленарного заседания 5–й Российской мультиконф. по проблемам управления. — М.: Изд–во ИПУ РАН, 2012. — С. 19–20.

290. Опарин, Г.А. Автоматизация разработки и применения пакетов программ для исследования динамики сложных управляемых систем / Г.А. Опарин // Автореф. дис. докт. техн. наук: 05.13.11. — Иркутск: Изд–во ИДСТУ СО РАН, 1998. — 40 с.

291. Опарин, Г.А. Булево моделирование планирования действий в распределенных вычислительных системах / Г.А. Опарин, А.П. Новопашин // Известия РАН. Теория и системы управления. — 2004. — № 5. — С. 105–108.

292. Опарин, Г.А. К теории планирования вычислительного процесса в пакетах прикладных программ / Г.А. Опарин // Пакеты прикладных программ. Методы и разработки. — Новосибирск: Наука, 1981. — С. 5–20.

293. Опарин, Г.А. Основанная на знаниях технология решения вычислительных задач / Г.А. Опарин // Информационные технологии контроля и управления транспортными системами. Вып. 6. — Иркутск: Изд–во Иркутского ин-та инженеров железнодорожного транспорта, 2000. — С. 3–15.

294. Опарин, Г.А. Планирование схем решения задач в инструментальном комплексе САТУРН / ПЗ / Г.А. Опарин, Д.Г. Феоктистов // Компьютерная логика, алгебра и интеллектное управление. — Иркутск: Изд–во ИрВЦ СО РАН, 1994. — Т. 1. — С. 3–13.

295. Пауэлл, Т. Полный справочник по JavaScript / Т. Пауэлл, Ф. Шнайдер. — М.: Вильямс, 2007. — 960 с.

296. Пипер, Ш. Новая эра в оценке производительности компьютерных систем / Ш. Пипер, П. Джоан, М. Сколт // Открытые системы. — 2007. — № 9. — С. 52–59.

297. Плакс, Т.П. Синтез параллельных программ на вычислительных моделях / Т.П. Плакс // Программирование. — 1977. — № 4. — С. 55–63.

298. Поздняк, Е.И. Генерализация алгоритма таксономического классификатора CARMA / Е.И. Поздняк, И.А. Сидоров, Ю.П. Галачьянц // Вестник Иркутского гос. техн. ун-та. — 2011. — № 9. — С. 11–15.

299. Поспелов, А.А. Логические методы анализа и синтеза схем / А.А. Поспелов. — М.–Ленинград: Энергия, 1964. — 320 с.

300. Поспелов, Д.А. Продукционные модели / Д.А. Поспелов // Искусственный интеллект. — Кн. 3: Модели и методы. — М.: Радио и связь, 1990. — С. 49–56.

301. Поспелов, Д.А. Ситуационное управление: теория и практика / Д.А. Поспелов. — М.: Наука, 1986. — 288 с.

302. Радченко, Г.И. Модели и методы профилирования и оценки времени выполнения потоков работ в суперкомпьютерных системах / Г.И. Радченко, Л.Б. Соколинский, А.В. Шамакина // Вычислительные методы и программирование. — 2013. — Т. 14, № 3. — С. 96–103.

303. Радченко, Г.И. Модель проблемно–ориентированной облачной вычислительной среды / Г.И. Радченко // Тр. Института системного программирования РАН. — 2015. — Т. 27, вып. 6. — С. 275–284.

304. Сальков, М.Ю. Имитационная модель процесса сервисного обслуживания / М.Ю. Сальков // Современная наука: теоретический и практический взгляд: Сб. статей Междунар. науч.–практ. конф. Уфа: АЭТЕРНА, 2015. — Ч. 1. — С. 37–40.

305. Сервис-ориентированная технология создания и применения децентрализованных мультиагентных решателей вычислительных задач / И.В. Бычков и др. // Вестник компьютерных и информ. технологий. — 2018. — № 12. — С. 36–44.

306. Система управления заданиями Cleo. — М.: НИВЦ МГУ, 2007. — 19 с.

307. Смагин, С.И. Генетический алгоритм составления расписания выполнения параллельных заданий в распределенной вычислительной системе / С.И. Смагин, Т.С. Шаповалов // Вычисл. технологии. — 2010. — Т. 15, № 5. — С. 107–122.

308. Соколинский, Л.Б. Методы управления ресурсами в проблемно–ориентированных вычислительных средах / Л.Б. Соколинский, А.В. Шамакина // Программирование. — 2016. — № 1. — С. 26–38.

309. СУППЗ [Электронный ресурс]. — Режим доступа: http://supz.jscc.ru / (дата обращения: 24.09.2021).

310. Сухорослов, О.В. Организация вычислений в гетерогенных распределенных средах / О.В. Сухорослов // Известия ЮФУ. Техн. серия. — 2016. — № 12. — С. 115–130.

311. Таха Хемди, А. Введение в исследование операция / Хемди А. Таха. — М.: Издательский дом «Вильямс», 2005. — 912 с.

312. Тель, Ж. Введение в распределенные алгоритмы. М.: МЦНМО, 2009. — 616 с.

313. Тимофеев, А.В. Мультиагентное управление и интеллектуальный анализ потоков данных / А.В. Тимофеев // Intern. J. Information Technologies and Knowledge. — 2013. — Vol. 7, № 3. — P. 282–285.

314. Тихомиров, А.И. Длительность проведения английского аукциона при планировании заданий с абсолютными приоритетами в распределенной вычислительной системе / А.И. Тихомиров, А.В. Баранов, В.В. Молоканов // Материалы XVIII Всерос. конф. молодых ученых по матем. моделированию и информ. технологиям. — Новосибирск: ИВТ СО РАН, 2017. — С. 93–94.

315. Толуев, Ю.И. Применение имитационного моделирования для исследования логистических процессов / Ю.И. Толуев // Имитационное моделирование. Теория и практика: Сб. II Всерос. науч.–практ. конф. — СПб.: ФГУП ЦНИИ ТС, 2005. — С. 71–76.

316. Топорков, В.В. Модели распределенных вычислений / В.В. Топорков. — М.: Физматлит, 2004. — 320 с.

317. Турский, В. Методология программирования / В. Турский // — М.: Мир, 1981. — 264 с.

318. Тыугу, Э.Х. Концептуальное программирование / Э.Х. Тыугу. — М.: Наука, 1984. — 256 с.

319. Тыугу, Э.Х. Решение задач на вычислительных моделях / Э.Х. Тыугу // Журнал вычисл. математики и матем. физики. — 1970. — Т. 10, № 3. — С. 716–733.

320. Уилкинсон, Дж.Х. Справочник алгоритмов на языке АЛГОЛ / Дж.Х. Уилкинсон, К. Райнш — М.: Машиностроение, 1976. — 389 с.

321. Ушаков, И.А. Вероятностные модели надежности информационно–вычислительных систем / И.А. Ушаков. — М.: Радио и связь, 1991. — 132 с.

322. Федорук, М.П. О перспективах Grid в сибирском регионе / М.П. Федорук, Д.Л. Чубаров, Ю.И. Шокин, А.В. Юрченко // Тр. Шестого совещания Российско–Казахстанской рабочей группы по вычисл. и информ. технологиям. — Алматы: КазНУ им. аль–Фараби, 2009. — С. 324–338.

323. Феоктистов, А.Г. ORLANDO TOOLS / С.А. Горский, А.Г. Феоктистов. Свидетельство о гос. регистрации программы для ЭВМ № 2007611625. — М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2007.

324. Феоктистов, А.Г. Автоматизация имитационного моделирования сложных систем в распределенной вычислительной среде / А.Г. Феоктистов, О.Ю. Башарина // Программные продукты и системы. — 2015. — № 3. — С. 75–79.

325. Феоктистов, А.Г. Автоматизация разработки и применения распределенных пакетов прикладных программ / А.Г. Феоктистов, И.А. Сидоров, С.А. Горский // Проблемы информатики. — 2017. — № 4. — С. 61–78.

326. Феоктистов, А.Г. Архитектура интеллектуального агента управления ресурсами распределенной вычислительной среды / И.А. Сидоров, А.Г. Феоктистов // Интеллектуальные системы: Тр. IX Междунар. симпозиума. — М.: РУСАКИ, 2010. — С. 228–232.

327. Феоктистов, А.Г. Аспекты имитационного моделирования процессов мультиагентного управления распределенными вычислениями / А.Г. Феоктистов, А.С. Корсуков, Ю.А. Дядькин // Информ. и матем. технологии в науке и управлении. — Иркутск: Изд–во ИСЭМ СО РАН, 2016. — № 4–1. — С. 118–126.

328. Феоктистов, А.Г. Библиотека алгоритмов децентрализованного управления распределенными вычислениями на основе мультиагентных технологий / А.Г. Феоктистов, А.С. Корсуков, Э.К. Вартанян, А.Н. Кантер. Свидетельство о гос. регистрации программы для ЭВМ № 2012660347. — М.: Федеральная служба по интеллектуальной собственности, 2012.

329. Феоктистов, А.Г. Библиотека алгоритмов для эффективного извлечения и применения проблемно–ориентированных знаний агентами: Свидетельство о гос. регистрации программы для ЭВМ № 2017663706 / А.Г. Феоктистов, Р.О. Костромин. — М.: Федеральная служба по интеллектуальной собственности, 2017.

330. Феоктистов, А.Г. Библиотека имитационных моделей систем массового обслуживания / А.В. Ларина, А.Г. Феоктистов, В.И. Дмитриев. Свидетельство о гос. регистрации программы для ЭВМ № 2009610133. — М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2009.

331. Феоктистов, А.Г. Генератор потоков заданий / А.С. Корсуков, А.Г. Феоктистов, В.И. Дмитриев. Свидетельство о гос. регистрации программы для ЭВМ № 2010615499. — М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2010.

332. Феоктистов, А.Г. Гетерогенная распределенная вычислительная среда для решения крупномасштабных задач исследования энергетической безопасности / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, И.А. Сидоров, С.А. Горский, Р.О. Костромин, А.В. Еделев // Информационные технологии и нанотехнологии: Сб. тр. V Междунар. конф. и молодежной школы (ИТНТ–2019). — Самара: Новая техника, 2019. — С. 445–454.

333. Феоктистов, А.Г. Децентрализованное управление вычислениями в распределенной мультиагентной среде / А.Г. Феоктистов, С.А. Миньков // Интеллектуальные системы: Тр. VII Междунар. симпозиума. — М.: РУСАКИ, 2006. — С. 478–481.

334. Феоктистов, А.Г. Децентрализованное управление потоками заданий в интегрированной кластерной системе / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, А.С. Корсуков // Вестник Новосибирского гос. ун-та. Серия: Информ. технологии. — 2011. — Т. 9, вып. 2. — С. 42–54.

335. Феоктистов, А.Г. Имитационная модель вычислительной системы решения ресурсоемких экономических задач / А.В. Ларина, А.Г. Феоктистов. Свидетельство о гос. регистрации программы для ЭВМ № 2009614044. — М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2009.

336. Феоктистов, А.Г. Имитационная модель гетерогенной распределенной вычислительной среды / А.Г. Феоктистов. Свидетельство о гос. регистрации программы для ЭВМ № 2018619931. — М.: Федеральная служба по интеллектуальной собственности, 2018.

337. Феоктистов, А.Г. Инструментальная распределенная вычислительная САТУРН–среда / Г.А. Опарин, А.Г. Феоктистов // Программные продукты и системы. — 2002. — № 2. — С. 27–30.

338. Феоктистов, А.Г. Инструментальные средства имитационного моделирования предметно–ориентированных распределенных вычислительных систем / А.Г. Феоктистов, А.С. Корсуков, Ю.А. Дядькин // Системы управления, связи и безопасности. — 2016. — № 4. — С. 30–60.

339. Феоктистов, А.Г. Инструментальные средства разработки распределенных пакетов прикладных программ на основе модульного подхода / А.Г. Феоктистов, И.А. Сидоров, С.А. Горский // Марчуковские научные чтения: Труды Международной конференции. — Новосибирск: Изд-во ИВМиМГ СО РАН, 2017. — С. 950-956.

340. Феоктистов, А.Г. Инструментальные средства разработки распределенных пакетов программ / А.Г. Феоктистов, И.А. Сидоров, С.А. Горский // ИТНОУ: Информационные технологии в науке, образовании и управлении. — 2017. — № 4. — С. 32–37.

341. Феоктистов, А.Г. Инструментальный комплекс DISCENT / Г.А. Опарин, А.Г. Феоктистов, А.С. Корсуков. Свидетельство о гос. регистрации программы для ЭВМ № 2009614676. — М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2009.

342. Феоктистов, А.Г. Инструментальный комплекс DISCOMP / И.А. Сидоров, А.Г. Феоктистов. Свидетельство о гос. регистрации программы для ЭВМ № 2008615180. — М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2008.

343. Феоктистов, А.Г. Инструментальный комплекс для автоматизации имитационного моделирования систем массового обслуживания / А.Г. Феоктистов, Ю.А. Дядькин. Свидетельство о гос. регистрации программы для ЭВМ № 2017616449. — М.: Федеральная служба по интеллектуальной собственности. — 2017.

344. Феоктистов, А.Г. Инструментальный комплекс для организации веб–интерфейсов к вычислительным кластерам (WIDT) / А.С. Корсуков, А.Г. Феоктистов, В.И. Дмитриев. Свидетельство о гос. регистрации программы для ЭВМ № 2007613651. — М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2007.

345. Феоктистов, А.Г. Инструментальный комплекс для организации гетерогенных распределенных вычислительных сред / И.В. Бычков, А.С. Корсуков, Г.А. Опарин, А.Г. Феоктистов // Информ. технологии и вычисл. системы. — 2010. — № 1. — С. 45–54.

346. Феоктистов, А.Г. Интеллектный подход к автоматизации моделирования сложных управляемых систем / С.Н. Васильев, Г.А. Опарин, А.Г. Феоктистов // Современные проблемы прикладной математики и механики: Тр. междунар. конф. RDAMM–2001. — Новосибирск: ИВТ СО РАН, 2001. — Т. 6. — Ч. 2. — С. 159–168.

347. Феоктистов, А.Г. Испытание и оценка надежности интегрированных кластерных систем на основе их комплексного моделирования / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, А.С. Корсуков // Вестник компьютерных и информ. технологий. — 2013. — № 3. — С. 3–8.

348. Феоктистов, А.Г. Классификация масштабируемых программных комплексов / А.Г. Феоктистов, А.С. Корсуков, О.Ю. Башарина // Вестник Иркутского гос. техн. ун-та. — 2017. — Т. 21, № 11. — С. 92–103.

349. Феоктистов, А.Г. Логико–вероятностные аспекты алгоритма управления распределенными вычислениями / А.Г. Феоктистов // Вычислительные технологии. — 2016. — Т. 21, № 3. — С. 91–102.

350. Феоктистов, А.Г. Масштабируемое приложение для поиска глобальных минимумов многоэкстремальных функций / И.В. Бычков, Г.А. Опарин, А.Н. Черных, А.Г. Феоктистов, С.А. Горский, Р. Ривера–Родригес // Автометрия. — 2018. — Т. 54, № 1. — С. 98–105.

351. Феоктистов, А.Г. Масштабируемое приложение для решения задач складской логистики / А.Г. Феоктистов. Свидетельство о гос. регистрации

программы для ЭВМ № 2018616531. — М.: Федеральная служба по интеллектуальной собственности, 2018.

352. Феоктистов, А.Г. Методика динамического анализа времени выполнения программ в гетерогенных распределенных вычислительных средах / А.Г. Феоктистов, О.Ю. Башарина // Вестник Иркутского гос. техн. ун-та. — 2018. — Т. 22, № 6. — С. 109–119.

353. Феоктистов, А.Г. Методология концептуализации и классификации потоков заданий масштабируемых приложений в разнородной распределенной вычислительной среде / А.Г. Феоктистов // Системы управления, связи и безопасности. — 2015. — № 4. — С. 1–25.

354. Феоктистов, А.Г. Методы и средства извлечения знаний в мультиагентной системе управления распределенными вычислениями / Р.О. Костромин, А.Г. Феоктистов, Ю.А. Дядькин // Материалы 10–й Всерос. мультиконф. — Ростов н / Д: Изд–во ЮФУ, 2017. — Т. 3. — С. 117–119.

355. Феоктистов, А.Г. Модели и инструментальные средства организации распределенных вычислений / Г.А. Опарин, А.Г. Феоктистов // Параллельные вычисления и задачи управления: Тр. IV Междунар. конф. — М.: Изд–во ИПУ РАН, 2008. — С. 1126–1135.

356. Феоктистов, А.Г. Моделирование систем массового обслуживания в гетерогенной распределенной вычислительной среде / А.Г. Феоктистов, Ю.А. Дядькин, Е.С. Фереферов, О.Ю. Башарина // Имитационное моделирование. Теория и практика (ИММОД–2017): Тр. VIII Всерос. науч.–практ. конф. по имитационному моделированию и его применению в науке и промышленности. — С.–П.: НП «НОИМ», 2017. — С. 554–558.

357. Феоктистов, А.Г. Поддержка управления живучестью систем энергетики на основе комбинаторного подхода / И.В. Бычков, С.А. Горский, А.В. Еделев, Р.О. Костромин, И.А. Сидоров, А.Г. Феоктистов, Е.С. Фереферов, Р.К. Федоров // Известия РАН. Теория и системы управления. — 2021. — № 6. — С. 122–135.

358. Феоктистов, А.Г. Мультиагентное управление вычислительной системой на основе метамониторинга и имитационного моделирования / И.В. Бычков,

Г.А. Опарин, А.Г. Феоктистов, И.А. Сидоров, В.Г. Богданова, С.А. Горский // Автометрия. — 2016. — Т. 52, № 2. — С. 3–9.

359. Феоктистов, А.Г. Мультиагентное управление распределенными вычислениями на основе сервис–ориентированного подхода / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, В.Г. Богданова, А.А. Пашинин // Суперкомпьютерные технологии: Материалы III Всерос. науч.–техн. конф. — Ростов н / Д: Изд–во ЮФУ, 2014. — Т. 2. — С. 80–84.

360. Феоктистов, А.Г. Мультиагентные методы и инструментальные средства управления в сервис–ориентированной распределенной вычислительной среде / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, В.Г. Богданова, А.А. Пашинин // Тр. Института системного программирования РАН. — 2014. — Т. 26, вып. 5. — С. 65–82.

361. Феоктистов, А.Г. Мультиагентный алгоритм построения остаточной схемы решения задачи в гетерогенной распределенной вычислительной среде / А.Г. Феоктистов, Р.О. Костромин, И.А. Сидоров, С.А. Горский // Материалы 5–й Всерос. науч.–техн. конф. «Суперкомпьютерные технологии». — Ростов н / Д: Изд–во Южного федерального ун–та, 2018. — Т. 2. — С. 71–75.

362. Феоктистов, А.Г. Мультиагентный алгоритм распределения вычислительных ресурсов на основе экономического механизма регулирования их спроса и предложения / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, А.Н. Кантер // Вестник компьютерных и информ. технологий. — 2014. — № 1. — С. 39–45.

363. Феоктистов, А.Г. Мультиагентный подход к распределению вычислительных ресурсов на основе экономического механизма регулирования их спроса и предложения / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, А.Н. Кантер // VI Всерос. мультиконф. по проблемам управления: Материалы мультиконф. в 4–х Т. — Ростов н / Д: Изд–во ЮФУ, 2013. — Т. 4. — С. 22–26.

364. Феоктистов, А.Г. Мультиагентный подход к управлению высокопроизводительными вычислениями / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, В.Г. Богданова, И.А. Сидоров, А.А. Пашинин //

Суперкомпьютерные технологии: Материалы IV Всерос. науч.–техн. конф. — Ростов н / Д: Изд–во ЮФУ, 2016. — Т. 2. — С. 136–140.

365. Феоктистов, А.Г. Мультиагентный подход к управлению распределенными вычислениями в интегрированной кластерной системе / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, А.С. Корсуков // Управление в распределенных сетецентрических и мультиагентных системах: Материалы IV Всерос. мультиконф. по проблемам управления. — Таганрог: Изд–во НИИ МВС ЮФУ, 2011. — Т. 1. — С. 224–226.

366. Феоктистов, А.Г. Мультиагентный подход к управлению распределенными вычислениями в кластерной Grid–системе / В.Г. Богданова, И.В. Бычков, А.С. Корсуков, Г.А. Опарин, А.Г. Феоктистов // Известия РАН. Теория и системы управления. — 2014. — № 5. — С. 95–105.

367. Феоктистов, А.Г. Мультиагентный подход к управлению сервис–ориентированными высокопроизводительными вычислениями / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, В.Г. Богданова, И.А. Сидоров, А.А. Пашинин // Вестник компьютерных и информ. технологий. — 2016. — № 9. — С. 35–41.

368. Феоктистов, А.Г. Непрерывная интеграция функционального наполнения распределенных пакетов прикладных программ в Orlando Tools / А.Г. Феоктистов, С.А. Горский, И.А. Сидоров, Р.О. Костромин, Е.С. Фереферов, И.В. Бычков // Тр. Института системного программирования РАН. — 2019. — Т. 31, № 2. — С. 83–96.

369. Феоктистов, А.Г. Обеспечение надежности системы интеллектуальных агентов в распределенной вычислительной САТУРН–среде / Г.А. Опарин, А.Г. Феоктистов // Интеллектуальные системы: Тр. VI междунар. симпозиума. — М.: РУСАКИ, 2004. — С. 436–439.

370. Феоктистов, А.Г. Планирование потоков заданий в Grid / А.С. Корсуков, А.Г. Феоктистов // Информационные и математические технологии в науке и управлении: Тр. XV Байкальской Всерос. конф.: — Иркутск: Изд–во ИСЭМ СО РАН, 2010. — С. 247–252.

371. Феоктистов, А.Г. Планировщик надежных планов решения задач в интегрированной кластерной системе / А.Г. Феоктистов, Э.К. Вартанян.

Свидетельство о государственной регистрации программы для ЭВМ № 2011617591. — М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2011.

372. Феоктистов, А.Г. Разработка и применение проблемно–ориентированных мультиагентных систем управления распределенными вычислениями / А.Г. Феоктистов, Р.О. Костромин // Известия ЮФУ. Техн. науки. — 2016. — № 11. — С. 65–74.

373. Феоктистов, А.Г. Разработка и применение распределенных пакетов прикладных программ / И.А. Сидоров, Г.А. Опарин, А.Г. Феоктистов // Программные продукты и системы. — 2010. — № 2. — С. 108–111.

374. Феоктистов, А.Г. Распределение заданий в интегрированной кластерной системе на основе их классификации / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, А.С. Корсуков // Вычислительные технологии. — 2013. — Т. 18, № 2. — С. 25–32.

375. Феоктистов, А.Г. Распределенная вычислительная среда для анализа уязвимости критических инфраструктур в энергетике / А.В. Еделев, С.М. Сендеров, Н.М. Береснева, И.А. Сидоров, А.Г. Феоктистов // Системы управления, связи и безопасности. — 2018. — № 3. — С. 197–231.

376. Феоктистов, А.Г. Распределенная имитационная модель вычислительного кластера / А.А. Александров, А.Г. Феоктистов, В.И. Дмитриев. Свидетельство о гос. регистрации программы для ЭВМ № 2009610134. — М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2009.

377. Феоктистов, А.Г. Распределенная система интеллектуальных почтовых клиентов в САТУРН–среде / Г.А. Опарин, А.Г. Феоктистов // Проблемы управления и моделирования в сложных системах: Тр. IV междунар. конф. — Самара: ИПУСС РАН, 2002. — С. 373–379.

378. Феоктистов, А.Г. Распределенный решатель булевых уравнений большой размерности: методы и средства управления вычислениями / Г.А. Опарин, А.Г. Феоктистов, А.П. Новопашин, В.Г. Богданова // Проблемы управления и

моделирования в сложных системах: Тр. VII междунар. конф. — Самара: Изд–во Самарского научного центра РАН, 2005. — С. 113–116.

379. Феоктистов, А.Г. Сервис анализа временных рядов природно-климатических показателей окружающей среды инфраструктурных объектов байкальской природной территории / А.Г. Феоктистов, Р.О. Костромин, О.Ю. Башарина // Оптика атмосферы и океана. Физика атмосферы: Материалы XXVII Международного симпозиума. — М.: Изд-во ИОА СО РАН, 2021. — Т. F. — С. 96–99.

380. Феоктистов, А.Г. Сервис подготовки и запуска имитационных моделей функционирования инфраструктурных объектов в распределенной вычислительной среде. / Р.О. Костромин, А.Г. Феоктистов. Свидетельство о государственной регистрации программы для ЭВМ № 2021662946. —– М.: Федеральная служба по интеллектуальной собственности, 2021.

381. Феоктистов, А.Г. Сервис–ориентированный мультиагентный подход к управлению распределенными вычислениями / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, В.Г. Богданова, А.А. Пашинин // Автоматика и телемеханика. — 2015. — № 11. — С. 118–131.

382. Феоктистов, А.Г. Сервис–ориентированный подход к организации распределенных вычислений с помощью инструментального комплекса DISCENT / И.В. Бычков, Г.А. Опарин, А.Г. Феоктистов, В.Г. Богданова, А.С. Корсуков // Информ. технологии и вычисл. системы. — 2014. — № 2. — С. 7–15.

383. Феоктистов, А.Г. Система децентрализованного управления распределенными вычислениями в Grid / Г.А. Опарин, А.Г. Феоктистов, А.С. Корсуков. Свидетельство о гос. регистрации программы для ЭВМ № 2009614675. — М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2009.

384. Феоктистов, А.Г. Система интеллектуальных агентов распределения ресурсов в интегрированной кластерной системе / А.Г. Феоктистов, А.Н. Кантер. Свидетельство о гос. регистрации программы для ЭВМ № 2011617593. — М.:

Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2011.

385. Феоктистов, А.Г. Система распределенного моделирования DSS / А.В. Ларина, А.Г. Феоктистов, В.И. Дмитриев. Свидетельство о гос. регистрации программы для ЭВМ № 2009610675. — М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2009.

386. Феоктистов, А.Г. Система управления заданиями для вычислительного кластера / А.С. Корсуков, А.Г. Феоктистов. Свидетельство о гос. регистрации программы для ЭВМ № 2007611624. —– М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам, 2007.

387. Феоктистов, А.Г. Средства и методы организации учебной информационно–вычислительной среды / А.Г. Феоктистов, В.И. Дмитриев, А.С. Корсуков, А.В. Ларина // Вестник МГПУ. Серия: Информатика и информатизация обучения. — 2007. — № 2 (9). — С. 37–41.

388. Феоктистов, А.Г. Технология имитационного моделирования распределенных пакетов знаний / А.Г. Феоктистов // Вестник Томского гос. ун-та. Приложение. — 2006. — № 18. — С. 248–251.

389. Феоктистов, А.Г. Технология разработки распределенных пакетов знаний / Г.А. Опарин, А.Г. Феоктистов // Проблемы управления и моделирования в сложных системах: Тр. III междунар. конф. — Самара: Самарский научный центр РАН, 2001. — С. 496–502.

390. Феоктистов, А.Г. Управление заданиями в гетерогенной распределенной вычислительной среде на основе знаний / А.Г. Феоктистов, Р.О. Костромин, Ю.А. Дядькин // Вестник компьютерных и информ. технологий. — 2018. — № 2. — С. 10–17.

391. Феоктистов, А.Г. Управление сложной системой на основе методологии многокритериального выбора управляющих воздействий / А.Г. Феоктистов // Фундаментальные исследования. — 2015. — № 9–1. — С. 82–86.

392. Феоктистов, А.Г. Язык спецификации вычислительных моделей в масштабируемых пакетах прикладных программ / А.Г. Феоктистов, С.А. Горский // Современные наукоемкие технологии. — 2016. — № 7. — С. 84–88.

393. Хьюз, Дж. Структурный подход к программированию / Дж. Хьюз, Дж. Мичтом. — М.: Мир, 1980. — 280 с.

394. Цейтин, Г.С. На пути к сборочному программированию / Г.С. Цейтин // Программирование. — 1990. — № 1. — С. 78–92.

395. ЦКП Иркутский суперкомпьютерный центр СО РАН [Электронный ресурс]. — Режим доступа: http://hpc.icc.ru / (дата обращения: 24.09.2021).

396. Человек и вычислительная техника / Под ред. В.М. Глушкова. — Киев: Наукова думка, 1971. — 294 с.

397. Шалимов, А.В. Метод компактного представления программ на основе частотных характеристик их поведения / А.В. Шалимов // Таврический вестник информатики и математики. — 2008. — № 2. — С. 243–250.

398. Шамакина, А.В. Обзор технологий распределенных вычислений / А.В. Шамакина // Вестник ЮФУ. Сер. Вычисл. математика и информатика. — 2014. — Т. 3, № 3. — С. 51–85.

399. Шоломов, Л.А. Логические методы исследования дискретных моделей выбора / Л.А. Шоломов. − М.: Наука, 1989. — 288 с.

400. Эксафлопные технологии. Концепция по развитию технологии высокопроизводительных вычислений на базе суперЭВМ эксафлопсного класса (2012–2020 гг.). — М.: Росатом, 2011. — 112 с.

401. Эрлих, А.И. Диалоговая система моделирования альтернатив и выбора решения в проектировании / А.И. Эрлих // Представление знаний в человеко–машинных и робототехнических системах. — М.: Изд–во Вычислительного центра АН СССР, 1984. — Т. С.2. — С. 209–220.

402. Якобовский, М.В. Облачный сервис для решения многомасштабных задач нанотехнологии на кластерах и суперкомпьютерах / М.В. Якобовский, А.А. Бондаренко, А.В. Выродов и др. // Известия ЮФУ. Техн. серия. — 2016. — № 12. — С. 103–114.

Список принятых сокращений

АМ – агрегированная модель

БЗ – база знаний

ВМ – виртуальная машина

ВС – виртуальное сообщество

ГРВС – гетерогенная распределенная вычислительная система

ИАС – информационно-аналитическая система

ИГУ – ФГБОУ ВО «Иркутский государственный университет»

ИДСТУ СО РАН – ФГБУН «Институт динамики систем и теории управления имени В.М. Матросова Сибирского отделения Российской академии наук»

ИНЦ – Иркутский научный центр

ИСКЦ – Иркутский суперкомпьютерный центр СО РАН

ИСЭМ СО РАН – ФГБУН «Институт систем энергетики им. Л.А. Мелентьева Сибирского отделения Российской академии наук»

ЛПР – лицо, принимающее решение

ЛСК – логистический складской комплекс

МАС – мультиагентная система

МИЭЛ – Международный институт экономики и лингвистики

ООО – общество с ограниченной ответственностью

ОС – операционная система

ПК – персональный компьютер

ПО – программное обеспечение

ППП – пакет прикладных программ

РППП – распределенный пакет прикладных программ

СУПЗ – система управления прохождением заданий

СХД – система хранения данных

ЦКП Центр коллективного пользования

ЭВМ – электронная вычислительная машина

API – Application Programming Interface

AppLeS – Application Level Scheduling

ARMS – Agent-based Resource Management System

CASE – Computer-Aided Software Engineering

CPU – Central Processing Unit

Condor DAGMan – Condor Directed Acyclic Graph Manager

DFD – Data Flow Diagrams

DISCENT – Distributed Computing Environment Toolkit

DISCOMP – DIStributed COmputing system of Modular Programming

EGEE – Enabling Grids for E-sciencE

EMI – European Middleware Initiative

ERM – Entity-Relationship Model

FCFS – First Come, First Serve

FLOPS – Floating Operations Per Second

GPSS – General Purpose Simulation System

GT – Globus Toolkit

HPC – High Performance Computing

HPCSOMAS – High-Performance Computing Service-Oriented Multi-Agent System

HTML – HyperText Markup Language

JDL – Job Description Language

LIGO – Laser Interferometer Gravitational Wave Observatory

LSF – Load Sharing Facility

MAAG – Multi-Agent Architecture for Grid Environment

Mac OS – Macintosh Operating System

MAGE – Mobile Agent-based Grid Environment

MS – Microsoft

OGC – Open Geospatial Consortium

Orlando Tools – Object Relations in LANguage of DescriptiOns Tools

OWL – Ontology Web Language

PanDa – Production and Distributed Analysis

PBS – Portable Batch System

PHP – Hypertext Preprocessor

PLUS – Programming Language Under Simulation

RAM – Random Access Memory

RDF – Resource Description Framework

REST – Representational State Transfer

SADT – Structured Analysis and Design Technique

SIPHT – sRNA Identification Protocol using High-throughput Technology

SLURM – Simple Linux® Utility for Resource Management

SOAP – Simple Object Access Protocol

SWF – Standard Workloads Format

UML – Unified Modeling Languag

UNICORE – Uniform Interface to Computing Resources

WLMS – Workload Management System

WMS – workflow management systems

WPS – Web Processing Service

WSDL – Web Services Description Language

XML – eXtensible Markup Language

Приложение А

Акты о внедрении

Акт о внедрении результатов диссертационного исследования в ИСЭМ СО РАН.

Министерство науки и высшего образования
Российской Федерации
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ
БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ НАУКИ
ИНСТИТУТ СИСТЕМ ЭНЕРГЕТИКИ
им. Л.А. МЕЛЕНТЬЕВА
СИБИРСКОГО ОТДЕЛЕНИЯ
РОССИЙСКОЙ АКАДЕМИИ НАУК
(ИСЭМ СО РАН)

664033, Иркутск-33, ул. Лермонтова, 130
Тел. (395-2) 42-47-00
Факс (395-2) 42-67-96
E-mail: info@isem.irk.ru
от 08.11.2021 г. № 15515/02-07-11

Утверждаю
Директор ИСЭМ СО РАН
чл.-корр. РАН
Стенников В.А.
«15» ноября 2021 г.

Акт
о внедрении результатов диссертационного исследования Феоктистова Александра Геннадьевича «Организация предметно-ориентированных распределенных вычислений в гетерогенной среде на основе мультиагентного управления заданиями», представленного на соискание ученой степени доктора технических наук по научной специальности 05.13.11 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

Настоящим актом подтверждается внедрение результатов диссертационного исследования Феоктистова Александра Геннадьевича «Организация предметно-ориентированных распределенных вычислений в гетерогенной среде на основе мультиагентного управления заданиями» в Институте систем энергетики им. Л.А. Мелентьева Сибирского отделения Российской академии наук (ИСЭМ СО РАН) с целью поддержки автоматизации создания и применения распределенных пакетов прикладных программ.

Использование разработанных в рамках диссертационного исследования инструментальных средств создания и применения распределенных пакетов прикладных программ обеспечило организацию требуемой вычислительной среды для решения актуальной задачи выявления критических элементов систем энергетики. Выполнение многовариантных расчетов в такой гетерогенной распределенной вычислительной среде с мультиагентным управлением заданиями обеспечило существенное сокращение сроков подготовки и проведения вычислительных экспериментов при решения данной задачи.

Заместитель директора по научной работе,
заведующий отделом энергетической безопасности,
доктор технических наук, профессор — Сендеров С.М.

Старший научный сотрудник
лаборатории живучести систем энергетики,
кандидат технических наук — Еделев А.В.

Акт о внедрении результатов диссертационного исследования в ООО «Катанна»

Общество с ограниченной ответственностью

«КАТАННА»

664043, г. Иркутск, ул. Медведева д,1 оф. 408 тел.48-76-89 тел./факс 48-76-98

Акт

о внедрении результатов диссертационного исследования Феоктистова Александра Геннадьевича «Организация предметно-ориентированных распределенных вычислений в гетерогенной среде на основе мультиагентного управления заданиями», представленного на соискание ученой степени доктора технических наук по научной специальности 05.13.11 – Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей

Настоящим актом подтверждается внедрение результатов диссертационного исследования Феоктистова Александра Геннадьевича «Организация предметно-ориентированных распределенных вычислений в гетерогенной среде на основе мультиагентного управления заданиями» в Обществе с ограниченной ответственностью (ООО) «Катанна» с целью решения актуальных практических задач анализа и оптимизации уровня обслуживания в рамках складской логистики.

Применение разработанного в рамках диссертационного исследования распределенного пакета прикладных программ для решения задач складской логистики обеспечило необходимую автоматизированную информационно-вычислительную поддержку при выборе оптимальных уровней обслуживания относительно логистических услуг, предоставляемых используемым складским комплексом. Проведение расчетов в гетерогенной распределенной вычислительной среде обеспечило существенное сокращение сроков решения задач за счет автоматизации основных этапов подготовки и проведения вычислительных экспериментов, а также эффективного мультиагентного управления процессом выполнения заданий на ресурсах среды.

Генеральный директор ООО «Катанна» Тарабукина Ирина Геннадьевна

«12» ноября 2021 г.

Справки об использовании результатов интеллектуальной деятельности в МИЭЛ ИГУ.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования

«ИРКУТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
(ФГБОУ ВО «ИГУ»)

МЕЖДУНАРОДНЫЙ ИНСТИТУТ ЭКОНОМИКИ И ЛИНГВИСТИКИ

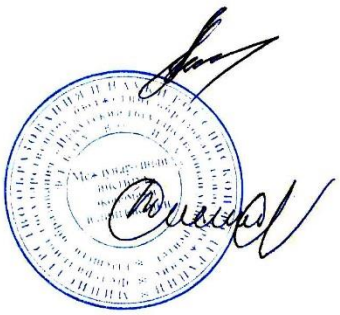
СПРАВКА

об использовании
результатов интеллектуальной деятельности
для автоматизации проведения научных исследований

Следующие результаты интеллектуальной деятельности «Система управления заданиями для вычислительного кластера» (Свидетельство об официальной регистрации программы для ЭВМ № 2007611624), «ORLANDO TOOLS» (Свидетельство об официальной регистрации программы для ЭВМ № 2007611625), «Инструментальный комплекс для организации Web-интерфейсов к вычислительным кластерам (WIDT)» (Свидетельство об официальной регистрации программы для ЭВМ № 2007613651), «Библиотека имитационных моделей систем массового обслуживания» (Свидетельство о государственной регистрации программы для ЭВМ № 2009610133), «Распределенная имитационная модель вычислительного кластера» (Свидетельство о государственной регистрации программы для ЭВМ № 2009610134), «Система распределенного моделирования DSS» (Свидетельство о государственной регистрации программы для ЭВМ № 2009610675) и «Генератор потоков заданий» (Свидетельство о государственной регистрации программы для ЭВМ № 2010615499) успешно использованы в Международном институте экономики и лингвистики федерального государственного бюджетного образовательного учреждения высшего образования «Иркутский государственный университет» МИЭЛ ФГБОУ ВО «ИГУ» в рамках НИР № 111-09-103 «Создание и внедрение в учебный процесс вычислительного кластера международного факультета» (2008-2010 гг., научный руководитель к.х.н., проф. Дмитриев В.И.).

Использование данных результатов интеллектуальной деятельности позволило автоматизировать процесс проведения научных исследований и ускорило получение основных результатов НИР.

Директор МИЭЛ ФГБОУ ВО «ИГУ»,
кандидат химических наук, доцент В.Я. Андрухова



Зав. кафедрой естественных дисциплин
МИЭЛ ФГБОУ ВО «ИГУ»,
кандидат химических наук, профессор В.И. Дмитриев

«26» октября 2016 г.

Акты о внедрении результатов интеллектуальной деятельности в ООО «Терминал комплекс».

УТВЕРЖДАЮ

Генеральный директор
ООО «Терминал Комплекс»

_____Ю.А.Толкачева
«28» 03 2017 г.

АКТ № 1-24/03/2017
О ВНЕДРЕНИИ ПРОГРАММЫ ДЛЯ ЭВМ
«ИНСТРУМЕНТАЛЬНЫЙ КОМПЛЕКС
ДЛЯ АВТОМАТИЗАЦИИ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ
СИСТЕМ МАССОВОГО ОБСЛУЖИВАНИЯ
В ЭКСПЛУАТАЦИЮ

Данный акт удостоверяет, что программа для ЭВМ «Инструментальный комплекс для автоматизации имитационного моделирования систем массового обслуживания» успешно внедрена в 2017 г. в эксплуатацию в ООО «Терминал комплекс».

Программа предназначена для автоматизации создания и применения имитационных моделей систем массового обслуживания. Программа обеспечивает следующие возможности: описание концептуальной модели предметной области исследуемой системы, формирование каркаса имитационной модели системы, наполнение каркаса программными фрагментами, представленными на языке GPSS и моделирующими процессы функционирования системы, компоновку и трансляцию всех возможных вариантов модели, их отладку и тестирование, а также автоматизацию планирования и проведения многовариантных экспериментов с имитационной моделью.

Использование внедренной программы обеспечило эффективное создание и применение имитационных моделей для решения ряда задачи оптимизации транспортных и складских логистических процессов, и тем самым способствовало существенному улучшению показателей качества логистических сервисов в ООО «Терминал комплекс».

Генеральный директор
ООО «Терминал Комплекс» Ю.А. Толкачева

«28» 03 2017 г.

ООО «Терминал комплекс»

АКТ О ВНЕДРЕНИИ

Данный акт удостоверяет, что следующие результаты интеллектуальной деятельности «ORLANDO TOOLS» (Свидетельство об официальной регистрации программы для ЭВМ № 2007611625), «Система управления заданиями для вычислительного кластера» (Свидетельство об официальной регистрации программы для ЭВМ № 2007611624), «Имитационная модель вычислительной системы решения ресурсоемких экономических задач» (Свидетельство о государственной официальной регистрации программы для ЭВМ № 2009614044), «Библиотека имитационных моделей систем массового обслуживания» (Свидетельство об официальной регистрации программы для ЭВМ № 2009610133), «Система распределенного моделирования DES» (Свидетельство об официальной регистрации программы для ЭВМ № 2009610675), «Инструментальный комплекс DISCENT» (Свидетельство об официальной регистрации программы для ЭВМ № 2009614676), «Программный комплекс для автоматизации имитационного моделирования в интегрированной кластерной системе» (Свидетельство о государственной регистрации программы для ЭВМ № 2012618317) успешно внедрены в эксплуатацию в ООО «Терминал комплекс».

Применение внедренных результатов интеллектуальной деятельности обеспечило оптимизацию транспортных и складских логистических процессов, повышение эффективности и качества логистического сервиса в ООО «Терминал комплекс».

Генеральный директор
ООО «Терминал Комплекс» Ю.А. Толкачева

«__18__» __апреля__ 2016 г.

Приложение Б

Описание вычислительной модели с помощью расширенной формы Бэкуса-Наура

вычислительная модель = объект {объект};

объект = модуль | параметр | операция | продукция |

| постановка задачи | схема решения задачи | задание |

| ресурс | административная политика | вычислительная история | агент;

модуль = базовый модуль | системный модуль;

базовый модуль = имя модуля ’,’ список формальных параметров;

системный модуль = имя модуля ‘,’ список формальных параметров;

имя модуля = строка;

список формальных параметров = имя параметра ‘,’ назначение параметра

{‘,’ имя параметра ‘,’ назначение параметра};

назначение параметра = “входной” | “выходной”;

параметр = базовый параметр | системный параметр;

базовый параметр = имя базового параметра ‘,’ тип параметра ‘,’ структура данных

‘,’ [индекс 1] ‘,’ [индекс 2] ‘,’

[параметр, на основе которого создается список ‘,’

число элементов списка] ‘,’ [имя файла] ‘,’

[область допустимых значений];

системный параметр = имя системного параметра ‘,’ тип параметра ‘,’

структура данных ‘,’ [индекс 1] ‘,’ [индекс 2] ‘,’

[параметр, на основе которого создается список ‘,’

число элементов списка] ‘,’ [имя файла];

имя базового параметра = имя параметра;

имя системного параметра = имя параметра;

имя параметра = строка;

тип параметра = “integer” | “long” | “Boolean” | “float” | “double” | “char” | “string” |

| “wide string” | “file” | “unknown”;

структура данных = “скаляр” | “вектор” | “матрица” | “список” |
| “произвольная структура”

имя файла = строка;

индекс 1 = имя параметра;

индекс 2 = имя параметра;

параметр, на основе которого создается список = имя параметра;

число элементов списка = имя параметра;

область допустимых значений = нижняя граница области допустимых значений ‘,’
верхняя граница области допустимых значений |
| число {‘,’ число} | строка {‘,’ строка}

операция = базовая операция | системная операция | составная операция;

базовая операция = имя операции ‘,’ список параметров операции ‘,’ имя модуля;

имя операции = строка;

системная операция = имя операции ‘,’ список параметров операции ‘,’ имя модуля;

имя операции = строка;

составная операция = имя операции {‘,’ имя операции};

список параметров операции = имя параметра ‘,’ назначение параметра
{‘,’ имя параметра ‘,’ назначение параметра};

продукция = имя продукции ‘,’ логическое выражение ‘,’ логическое выражение ‘,’
имя базовой операции ‘,’ имя системной операции ‘,’
приоритет продукции;

имя продукции = строка;

имя базовой операции = имя операции;

имя системной операции = имя операции;

логическое выражение = имя параметра {логический оператор имя параметра};

логический оператор = ‘∧’ | ‘∨’ | ‘ ¯ ’;

приоритет продукции = натуральное число;

постановка задачи = имя постановки задачи ‘,’ [список исходных параметров] ‘,’
[список целевых параметров] ‘,’ [список операций] ‘,’
[список критериев качества решения задачи] ‘,’

[список ресурсов];

[список критериев эффективности использования ресурсов];

имя постановки задачи = строка;

список исходных параметров = имя параметра {‘,’ имя параметра};

список целевых параметров = имя параметра {‘,’ имя параметра};

список операций = имя операции {‘,’ имя операции};

список критериев качества решения задачи = имя системного параметра

оператор число {‘,’ имя системного параметра оператор число};

список критериев эффективности использования ресурсов =

список критериев эффективности использования ресурса ‘,’

{список критериев эффективности использования ресурса};

список критериев эффективности использования ресурса = имя ресурса ‘,’

имя системного параметра оператор число

{‘,’ имя системного параметра оператор число};

схема решения задачи = имя схемы решения задачи ‘,’

список исходных параметров ‘,’

список целевых параметров ‘,’ список операций ‘,’

список критериев качества решения задачи ‘,’

список ресурсов ‘,’

список критериев эффективности использования ресурсов

‘,’ список ограничений;

имя схемы решения задачи = строка;

задание = имя задания ‘,’ имя схемы решения задачи ’,’ список характеристик;

имя задания = строка;

класс заданий = имя класса заданий ‘,’ список характеристик;

список ресурсов = ресурс {‘,’ ресурс};

ресурс = имя ресурса ‘,’ список характеристик;

имя ресурса = строка;

список характеристик = характеристика { ‘,’ характеристика};

характеристика = системный параметр;

вычислительные ресурсы = список ресурсов;

каналы связи = список ресурсов;

сетевые устройства = список ресурсов;

административная политика = список ограничений;

список ограничений = ограничение {‘,’ ограничение};

ограничение = имя системного параметра оператор число | строка;

пользователь = имя пользователя ‘,’ логин ‘,’ пароль ‘,’ группа пользователей;

имя пользователя = строка;

логин = строка;

пароль = строка;

группа пользователей = имя группы ‘,’ пользователь {‘,’ пользователь};

имя группы = строка;

вычислительная история = задание ‘,’ имя системного параметра;

агент = имя агента ‘,’ список характеристик ‘,’ имя ресурса;

имя агента = строка;

оператор = ‘=’ | ‘≠’ | ‘>’ | ‘<’ | ‘≥’ | ‘≤’;

строка = буква {буква | цифра | специальный символ};

нижняя граница области допустимых значений = число;

верхняя граница области допустимых значений = число;

число = [‘+’ | ‘-‘] натуральное число [‘.’ [натуральное число]]

[(‘e’ | ‘E’) [‘+’ | ‘-‘] натуральное число];

натуральное число = цифра {цифра};

буква = ‘A’ | ‘B’ | … | ‘Z’ | ‘a’ | ‘b’ | … | ‘z’;

цифра = ‘0’ | ‘1’ | ‘2’ | ‘3’ | ‘4’ | ‘5’ | ‘6’ | ‘7’ | ‘8’ | ‘9’;

специальный символ = ‘_’;

Приложение В

Характеристики узлов в эксперименте по прогнозированию времени выполнения заданий

В данном приложении приведены вычислительные характеристики эталонного и целевого узлов, рассмотренных во второй главе диссертации при описании алгоритма прогнозирования выполнения программных модулей (таблица В.1). Представлены показатели использования компонентов эталонного узла при решении задач перемножения целочисленных и вещественных матриц (таблицы В.2 и В.3). Описаны вычислительные характеристики узлов кластеров ГРВС (таблица В.4).

Таблица В.1 – Вычислительные характеристики узлов

Номер характеристики	Характеристика	Узел	
		Эталонный	Целевой
1.	Частота процессора (c_1), Ггц	3.0	2.0
2.	Число целочисленных операций, выполняемых процессором в секунду (c_2), Gips	11.99	7.87
3.	Число операций с плавающей точкой, выполняемых процессором в секунду (c_3), Гфлопс	5.93	2.67
4.	Пропускная способность кэш-памяти первого уровня (c_4), Гбит/с	46.9	31.38
5.	Латентность кэш-памяти первого уровня (c_5), нс	1.5	1.6
6.	Пропускная способность кэш-памяти второго уровня (c_6), Гбит/с	19.08	10.4
7.	Латентность кэш-памяти второго уровня (c_7), нс	13.2	11.2
8.	Пропускная способность оперативной памяти (c_8), Гбит/с	10.4	83.7
9.	Латентность оперативной памяти (c_9), нс	4.01	2.41

Таблица В.1 – Вычислительные характеристики узлов (продолжение)

Номер характеристики	Характеристика	Узел	
		Эталонный	Целевой
1.	Частота процессора (c_1), Ггц	3.0	2.0
2.	Число целочисленных операций, выполняемых процессором в секунду (c_2), Gips	11.99	7.87
3.	Число операций с плавающей точкой, выполняемых процессором в секунду (c_3), Гфлопс	5.93	2.67

Таблица В.2 – Показатели $h_1 - h_{12}$ использования компонентов эталонного узла при решении задач перемножения целочисленных матриц

h	n				
	1000	1500	2000	2500	3000
h_1	50708000768	169752002560	401968005120	784008019968	1354304061440
h_2	43143999488	147588005888	375827988480	715814010880	1288384020480
h_3	31118000128	104951996416	250564001792	490354999296	849509023744
h_4	6161500160	20511000576	48405000192	94425497600	162880995328
h_5	9700000	10700000	15800854	20800000	22900000
h_6	1110086725	9577700352	24692899840	51783299072	96418897920
h_7	264129996	60100000	6353600852	393800000	1219200000
h_8	98001224	7620000	615629977	80600000	790499968
h_9	200056	550000	1180000	2970000	4430000
h_{10}	8000000	18000000	32000000	50000000	72000000
h_{11}	4000000	9000000	16000000	25000000	36000000

Таблица В.3 – Показатели $h_1 - h_{12}$ использования компонентов эталонного узла при решении задач перемножения вещественных матриц

h	n				
	1000	1500	2000	2500	3000
h_1	50694000640	169817997312	401813995520	784619995136	969225994240
h_2	59855998976	190114004992	507904000000	1069264011264	1587370000384
h_3	32444000256	109456998400	259550003200	507339997184	503564992512
h_4	6132499968	20514500608	48427999232	94509998080	96257499136
h_5	7900000	11700000	15000000	28300000	27900000
h_6	4281400064	17497800704	44496498688	77962903552	93280100352
h_7	141600000	454900000	1286300032	3949100032	9209600000
h_8	132200000	437600000	1145699968	2273100032	13438699520
h_9	720000	1540000	3970000	9360000	12030000
h_{10}	16000000	36000000	64000000	100000000	144000000
h_{11}	8000000	18000000	32000000	50000000	72000000

Таблица В.4 – Вычислительные характеристики узлов кластеров ГРВС

c	Кластер	
	«Академик В.М. Матросов», сегмент 1	«Академик В.М. Матросов», сегмент 2
c_1	2.3	2.1
c_2	19.3	327.51
c_3	27.7	473.4
c_4	91.2	820.83
c_5	2.9	1.2
c_6	39.7	286.77
c_7	42.1	3.8
c_8	12.4	65
c_9	113.1	64.8
c_{10}	3	0.14

Таблица В.4 – Вычислительные характеристики узлов кластеров ГРВС (продолжение)

c	Кластер	
	«Академик В.М. Матросов», сегмент 1	«Академик В.М. Матросов», сегмент 2
c_{11}	121.3	510.6
c_{12}	115.2	390.1

Приложение Г

Интерфейс классификатора заданий

Главное меню классификатора заданий включает четыре вкладки (рисунок Г.1):

– «Характеристики»;
– «Классы»;
– «Ресурсы»;
– «Классы и ресурсы».

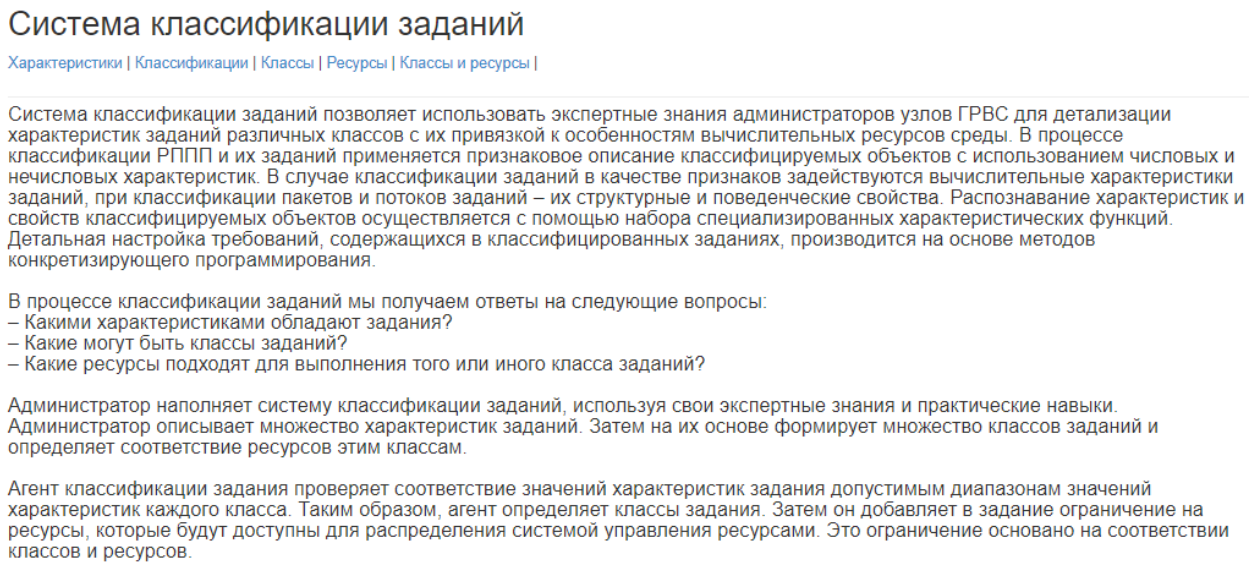


Рисунок Г.1 – Скриншот главного меню классификатора заданий

Вкладка «Характеристики» предназначена для создания, редактирования, просмотра и удаления характеристик заданий (рисунок Г.2). Вкладка содержит следующие поля для описания характеристик:

– поле «Имя характеристики», предназначенное для задания содержательного имени характеристики;
– поле «Комментарии», предназначенное для описания назначения характеристики;

– поле «Тип», предназначенное для задания типа характеристики из числа допустимых типов данных («Integer», «Float», «Double», «Boolean» и «String»);
– поле «Вес», предназначенное для задания численного выражения важности характеристики;
– поле «Ранг», предназначенное для задания степени важности характеристики;
– поле «Область допустимых значений».


Show | Add

Характеристики

N	Действия	Имя характеристики	Комментарии	Тип	Вес	Ранг	Область допустимых значений
1	Edit | Delete	Число ядер на узле	Число вычислительных элементов (ядер) на каждом узле	integer	1	1	[1; 36]
2	Edit | Delete	Число узлов	Число вычислительных устройств	integer	1	1	[1; 20]
3	Edit | Delete	Имя приложения	Имя программного приложения, выполняемого в задании	string	1	1	any value
4	Edit | Delete	Число виртуальных машин	Число виртуальных машин, которые будут запущены на одном узле	integer	1	1	[1; 36]
5	Edit | Delete	Тип гостевой ОС	Тип операционной системы для виртуальных машин	string	1	1	1-256
6	Edit | Delete	Промежуточное ПО для виртуальных машин	Используемый менеджер ресурсов	string	1	1	1-256
7	Edit | Delete	Объем ОЗУ	Требуемый объем оперативной памяти	integer	1	1	[8; 128]
8	Edit | Delete	Путь до исполняемого файла	Полный путь до файла для запуска	string	1	1	1-256

Рисунок Г.2 – Скриншот вкладки «Характеристики»

Кнопка «Show» служит для просмотра списка характеристик. Кнопки «Add» и «Edit» предназначены для открытия форм добавления новой (рисунок Г.3) и редактирования существующей (рисунок Г.4) характеристики. Кнопка «Delete» предназначена для удаления выбранной характеристики.

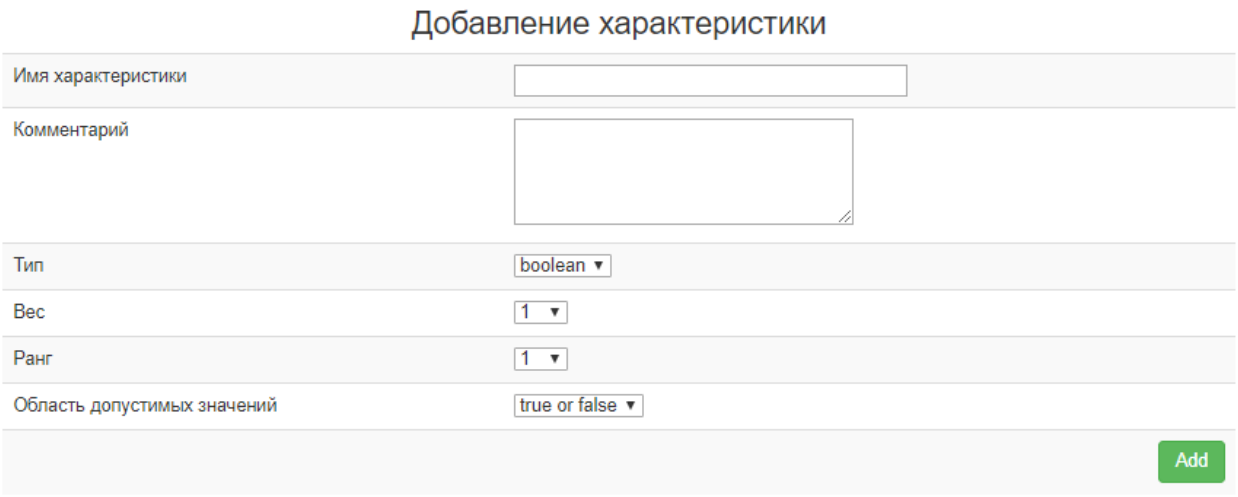


Рисунок Г.3 – Скриншот формы для добавления характеристики

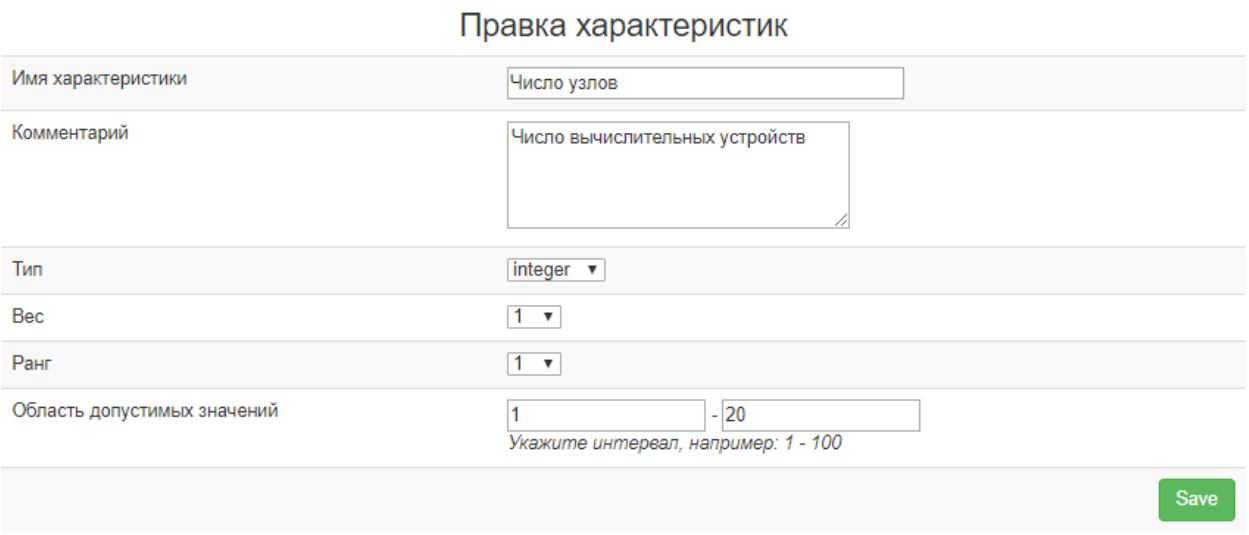


Рисунок Г.4 – Скриншот формы для редактирования характеристики

Вкладка «Классы» предназначена для создания, редактирования, просмотра и удаления классов заданий (рисунок Г.5). Вкладка содержит следующие поля для описания классов:

– поле «Имя класса», предназначенное для задания содержательного имени класса;
– поле «Комментарии», предназначенное для описания назначения класса.

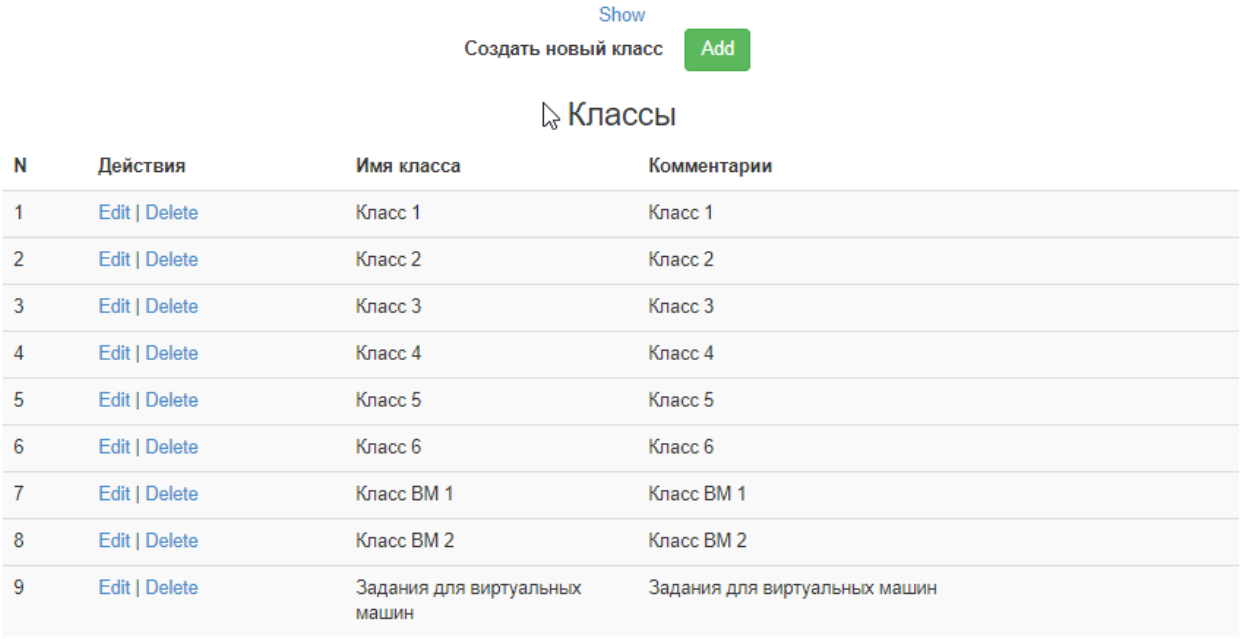


Рисунок Г.5 – Скриншот вкладки «Классы»

Кнопка «Show» служит для просмотра списка классов. Кнопки «Add» и «Edit» предназначены для открытия форм добавления нового (рисунок Г.6) и редактирования существующего (рисунок Г.7) класса. Кнопка «Delete» предназначена для удаления выбранного класса. Каждая из форм «Add» и «Edit» содержит подчиненную форму со списком характеристик вновь создаваемого или редактируемого класса. Подчиненная форма включает поля, аналогичные полям, которые содержатся на вкладке «Характеристики».

Show

Добавить класс

Имя класса

Комментарии

Добавить характеристику к классу

N	Добавить	Имя характеристики	Кмомментарии	Тип	Обязательный	Вес	Ранг	Область допустимых значений
1	☐	Число ядер на узле	Число вычислительных элементов (ядер) на каждом узле	integer	false	1	1	[1; 36]
2	☐	Максимальное время выполнения	Максимальное время выполнения в секундах	integer	false	1	1	[1; 1728000]
3	☐	Виртуализация	Поддержка виртуализации узлом	boolean	false	1	1	none
4	☐	Число узлов	Число вычислительных устройств	integer	false	1	1	[1; 20]
5	☐	Имя приложения	Имя программного приложения, выполняемого в задании	string	false	1	1	
6	☐	Число виртуальных машин	Число виртуальных машин, которые будут запущены на одном узле	integer	false	1	1	[1; 36]
7	☐	Тип гостевой ОС	Тип операционной системы для виртуальных машин	string	false	1	1	1-256
8	☐	Промежуточное ПО для виртуальных машин	Используемый менеджер ресурсов	string	false	1	1	1-256
9	☐	Объем ОЗУ	Требуемый объем оперативной памяти	integer	false	1	1	[8; 128]

Рисунок Г.6 – Скриншот формы для добавления класса

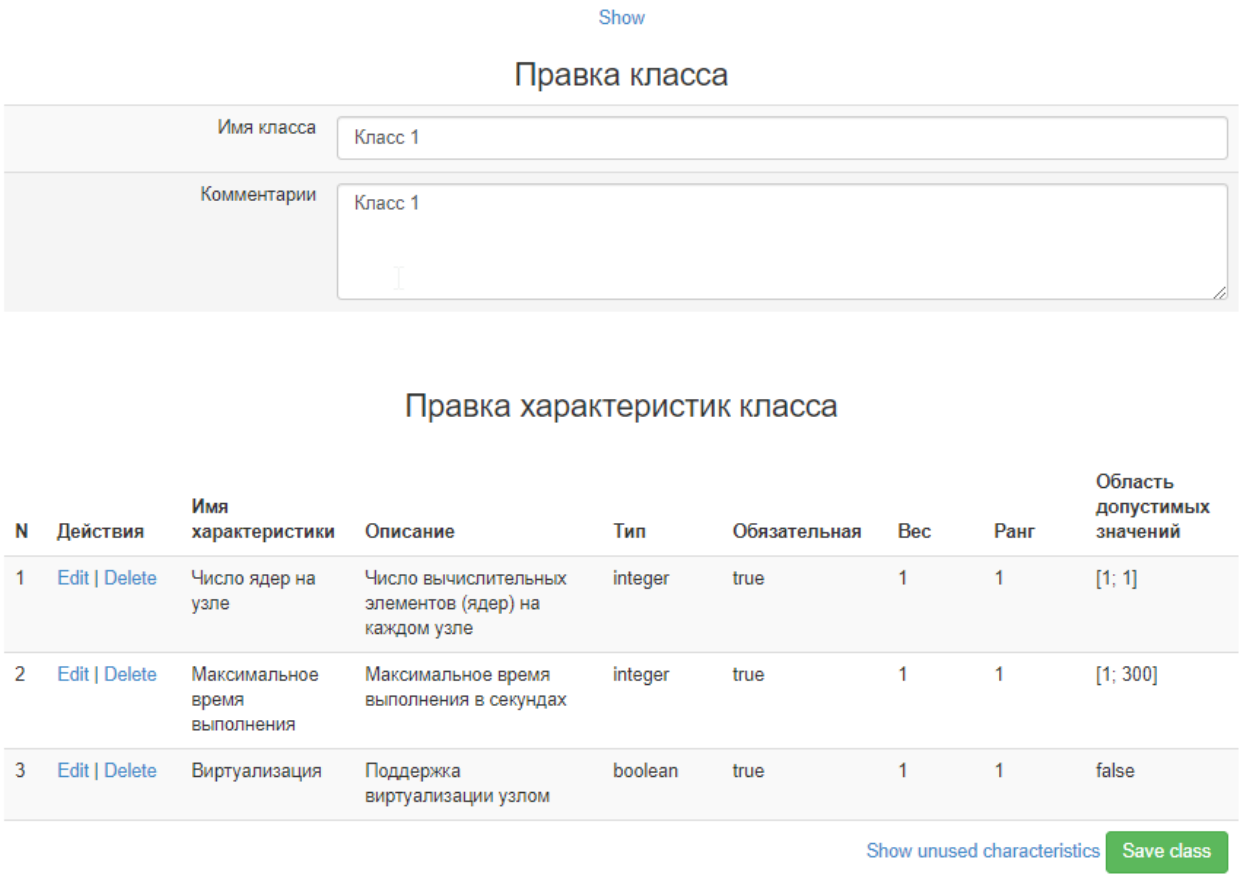


Рисунок Г.7 – Скриншот формы для редактирования класса

Вкладка «Ресурсы» предназначена для создания, редактирования, просмотра и удаления ресурсов ГРВС (рисунок Г.8). Вкладка содержит следующие поля для описания характеристик:

– поле «Имя ресурса», предназначенное для задания содержательного имени ресурса.

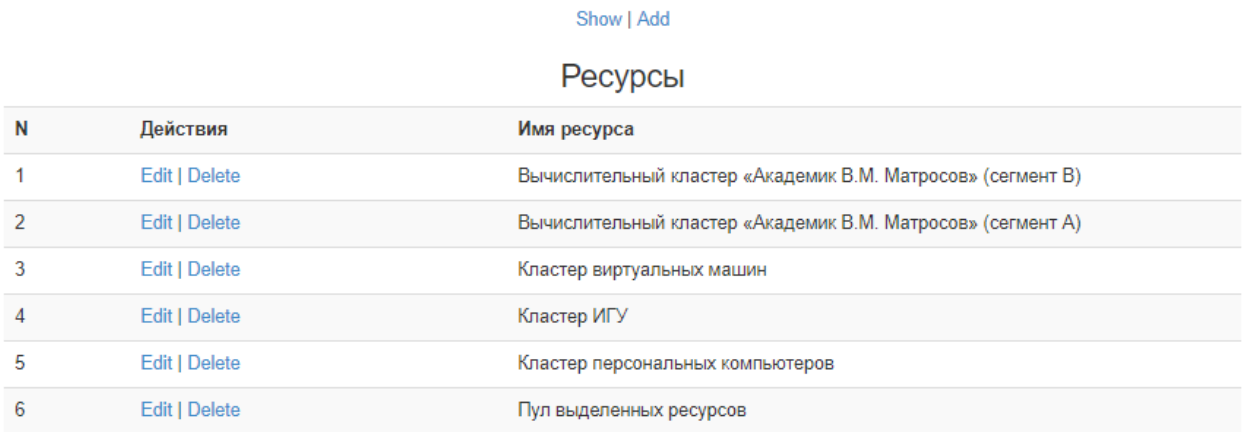


Рисунок Г.8 – Скриншот вкладки «Ресурсы»

Кнопка «Show» служит для просмотра списка ресурсов. Кнопки «Add» и «Edit» предназначены для открытия форм добавления ресурса (рисунок Г.9) и редактирования существующего (рисунок Г.10) ресурса, которые содержат поля «Имя ресурса» и «Комментарий». Кнопка «Delete» предназначена для удаления выбранного ресурса.

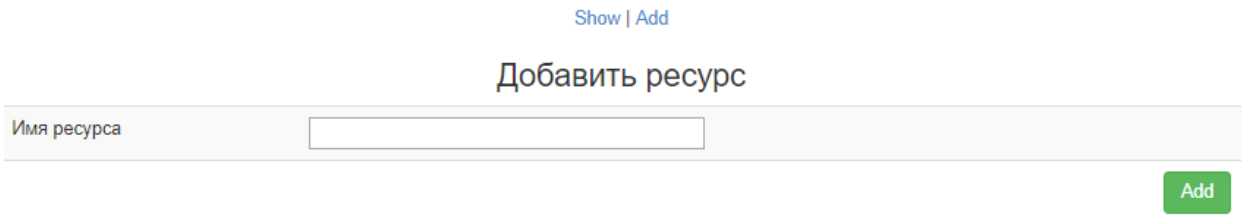


Рисунок Г.9 – Скриншот формы для добавления ресурса

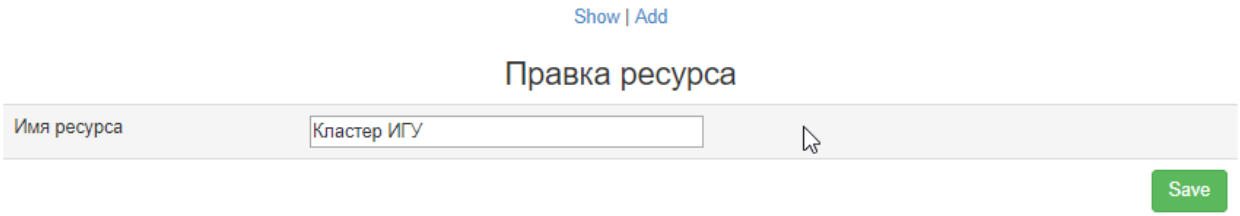
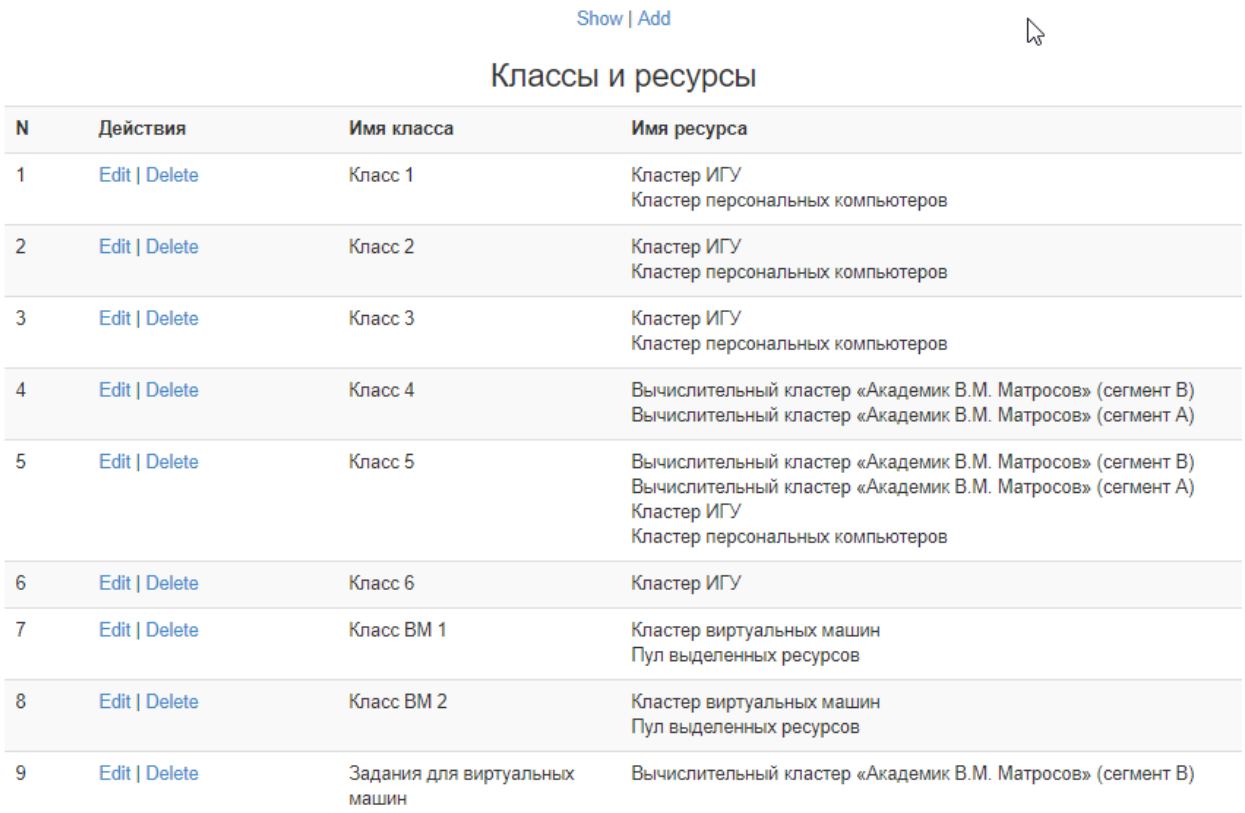


Рисунок Г.10 – Скриншот формы для редактирования ресурса

Вкладка «Классы и ресурсы» предназначена для создания, редактирования, просмотра и удаления связей между классами заданий и ресурсами, подходящими для выполнения заданий (рисунок Г.11). Вкладка содержит следующие поля:

– поле «Имя класса», предназначенное для указания содержательного имени класса;
– поле «Имя ресурса», предназначенное для указания содержательного имени ресурса.



N	Действия	Имя класса	Имя ресурса
1	Edit | Delete	Класс 1	Кластер ИГУ Кластер персональных компьютеров
2	Edit | Delete	Класс 2	Кластер ИГУ Кластер персональных компьютеров
3	Edit | Delete	Класс 3	Кластер ИГУ Кластер персональных компьютеров
4	Edit | Delete	Класс 4	Вычислительный кластер «Академик В.М. Матросов» (сегмент В) Вычислительный кластер «Академик В.М. Матросов» (сегмент А)
5	Edit | Delete	Класс 5	Вычислительный кластер «Академик В.М. Матросов» (сегмент В) Вычислительный кластер «Академик В.М. Матросов» (сегмент А) Кластер ИГУ Кластер персональных компьютеров
6	Edit | Delete	Класс 6	Кластер ИГУ
7	Edit | Delete	Класс ВМ 1	Кластер виртуальных машин Пул выделенных ресурсов
8	Edit | Delete	Класс ВМ 2	Кластер виртуальных машин Пул выделенных ресурсов
9	Edit | Delete	Задания для виртуальных машин	Вычислительный кластер «Академик В.М. Матросов» (сегмент В)

Рисунок Г.11 – Скриншот вкладки «Классы и ресурсы»

Кнопка «Show» служит для просмотра списка соответствия классов и ресурсов. ресурсов. Кнопки «Add» и «Edit» предназначены для открытия форм добавления новой (рисунок Г.12) и редактирования существующей (рисунок Г.13) связи между классами и ресурсами, которые содержат поля «Имя класса» и «Имя ресурса». Кнопка «Delete» предназначена для удаления выбранной связи между классами и ресурсами.

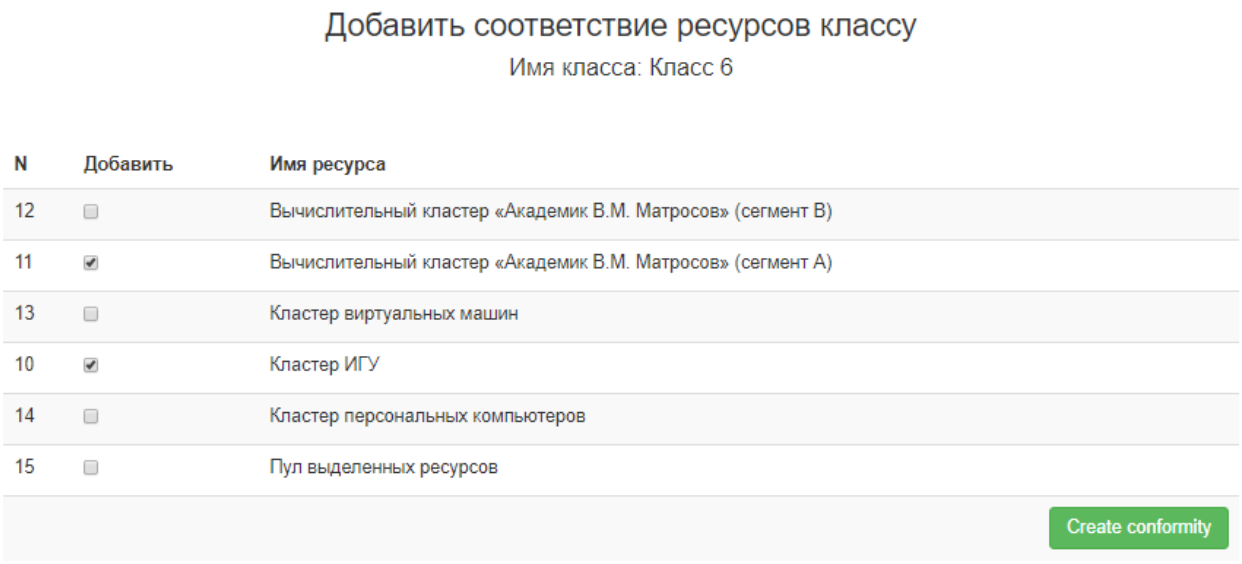


Рисунок Г.12 – Скриншот формы для добавления связи

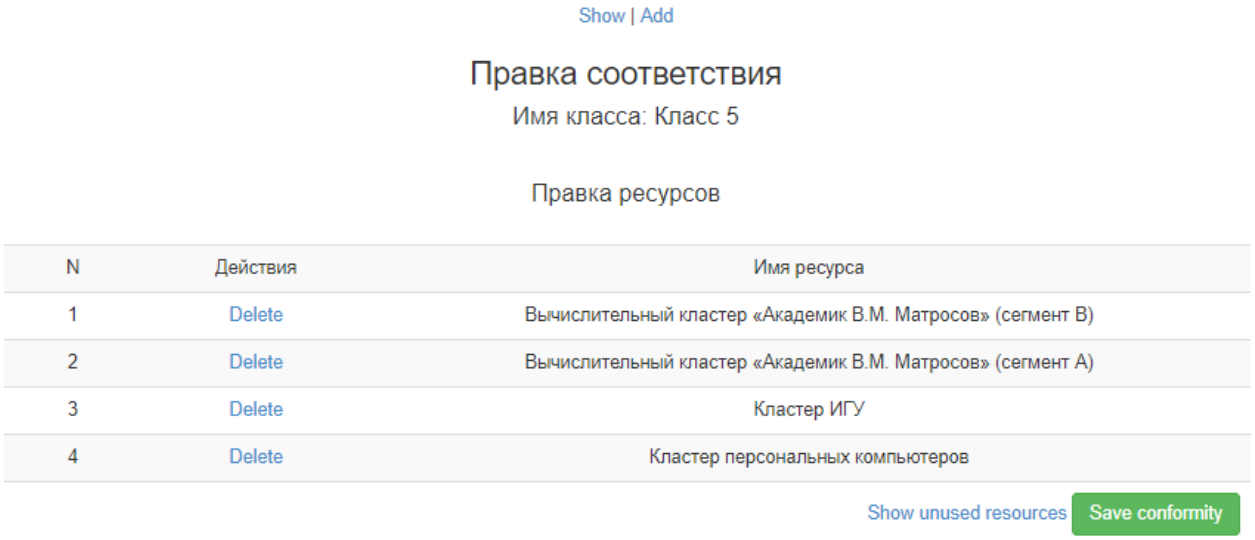


Рисунок Г.13 – Скриншот формы для редактирования связи

Приложение Д

Типовые рабочие процессы

На рисунке Д.1-Д.5 схематично приведены типовые рабочие процессы Montage, CyberShake, Epigenomics, LIGO и SIPHT.

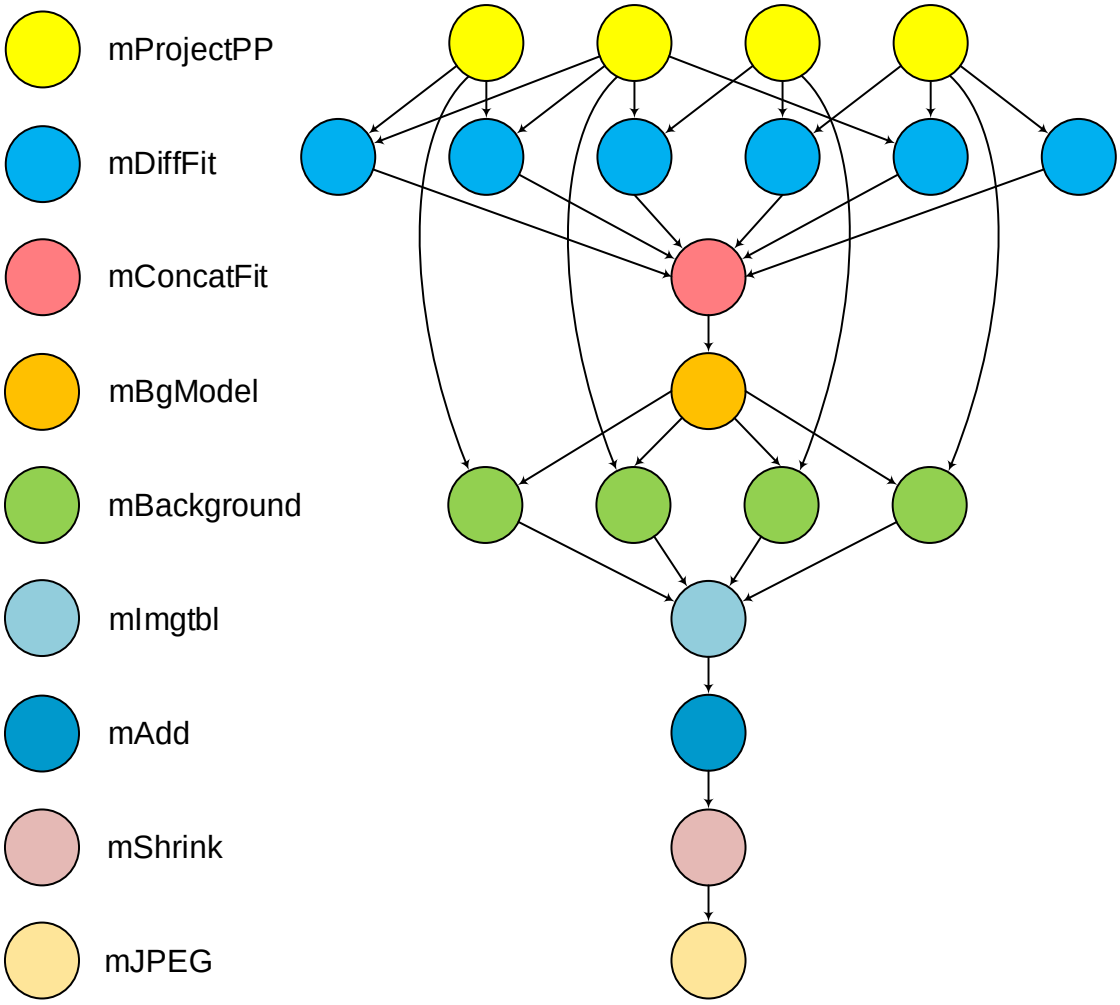


Рисунок Д.1 – Рабочий процесс Montage

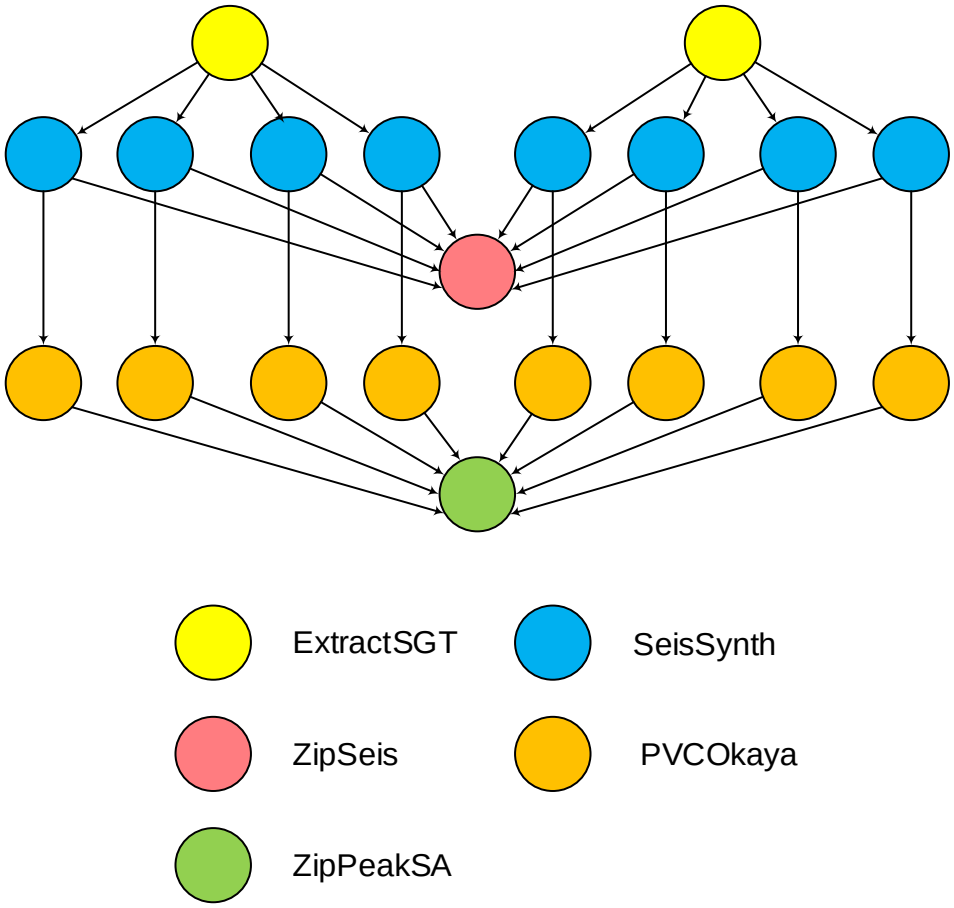


Рисунок Д.2 – Рабочий процесс CyberShake

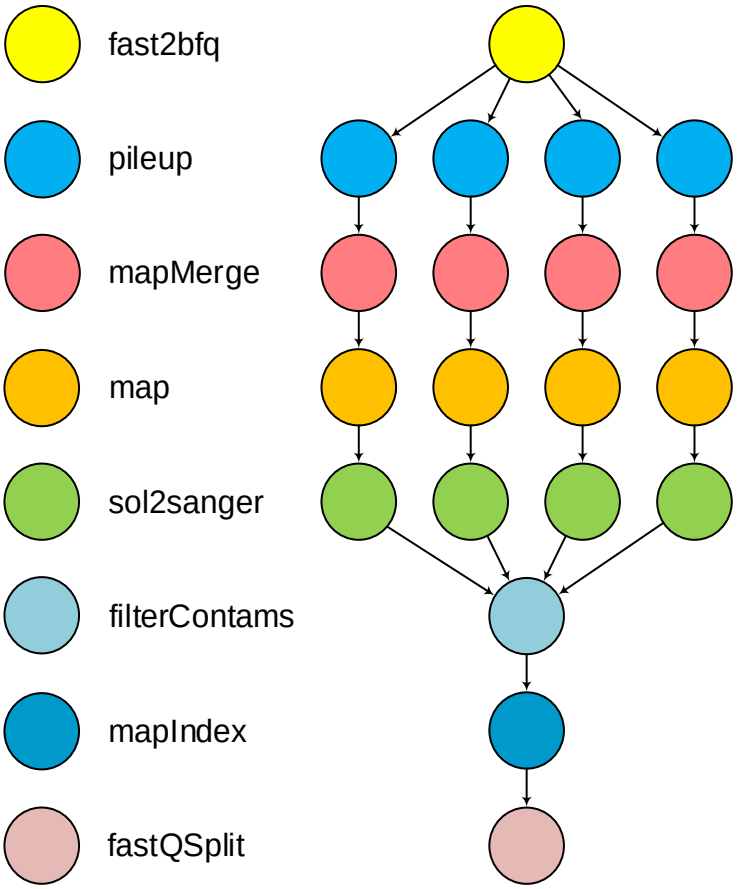


Рисунок Д.3 – Рабочий процесс Epigenomics

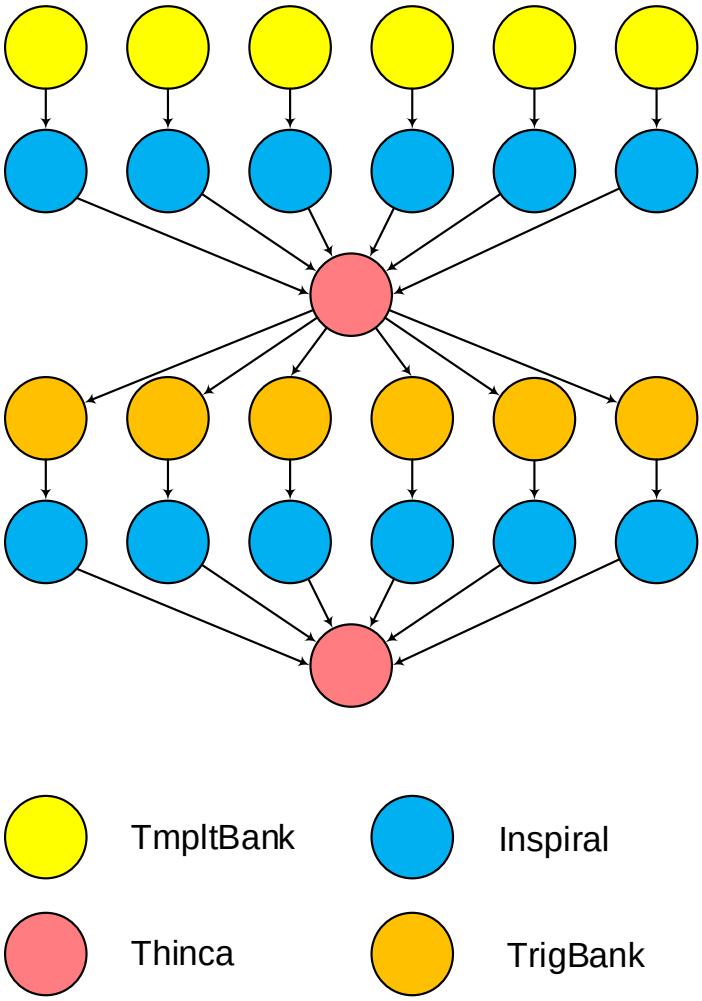


Рисунок Д.4 – Рабочий процесс LIGO

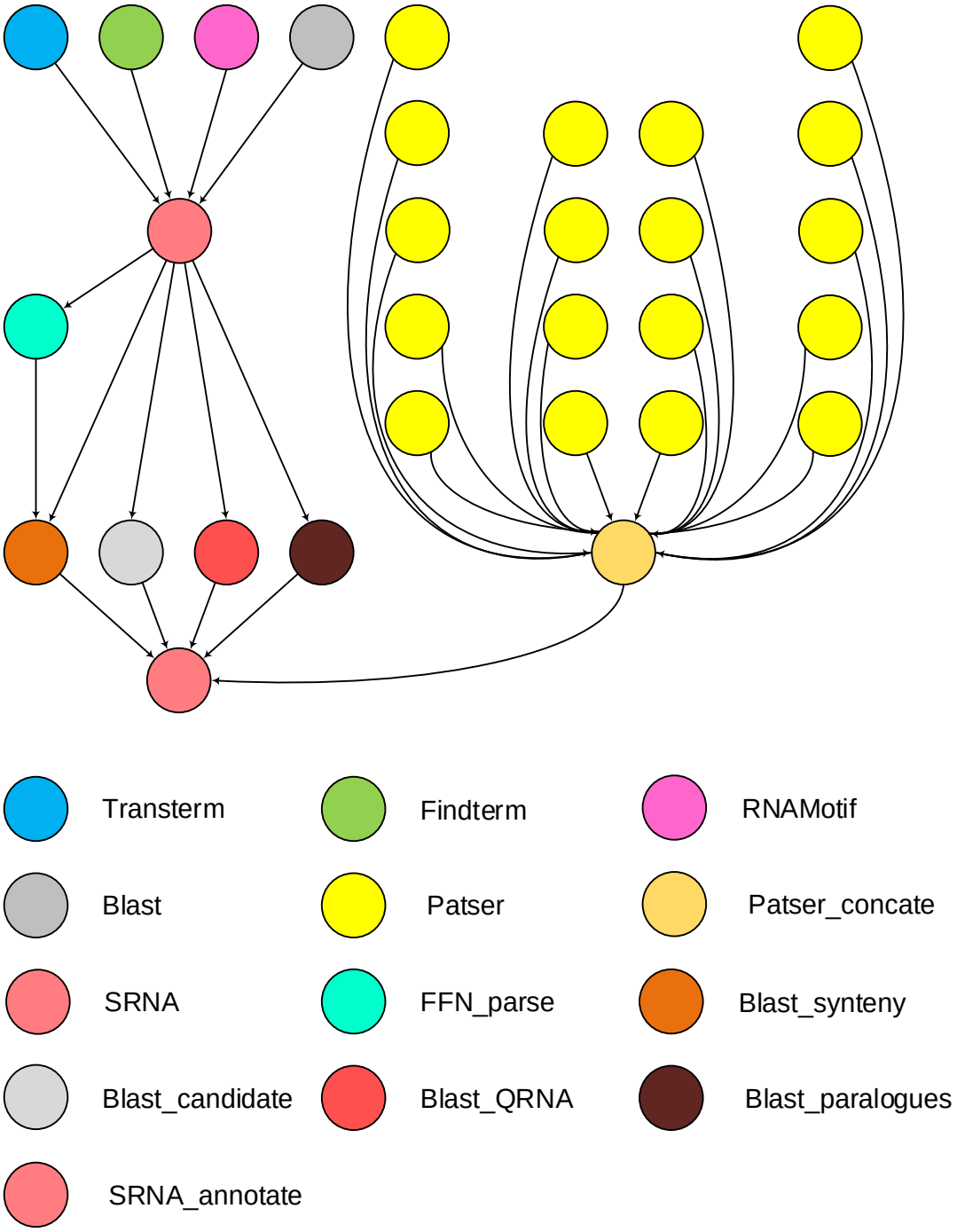


Рисунок Д.5 – Рабочий процесс SIPHT

Приложение Е

Операции рабочих процессов по уровням

На рисунке Е.1 приведены операции рабочих процессов Montage (а), CyberShake (б), Epigenomics (в), LIGO (г) и SIPHT (д) с учетом использования параллельных операций.

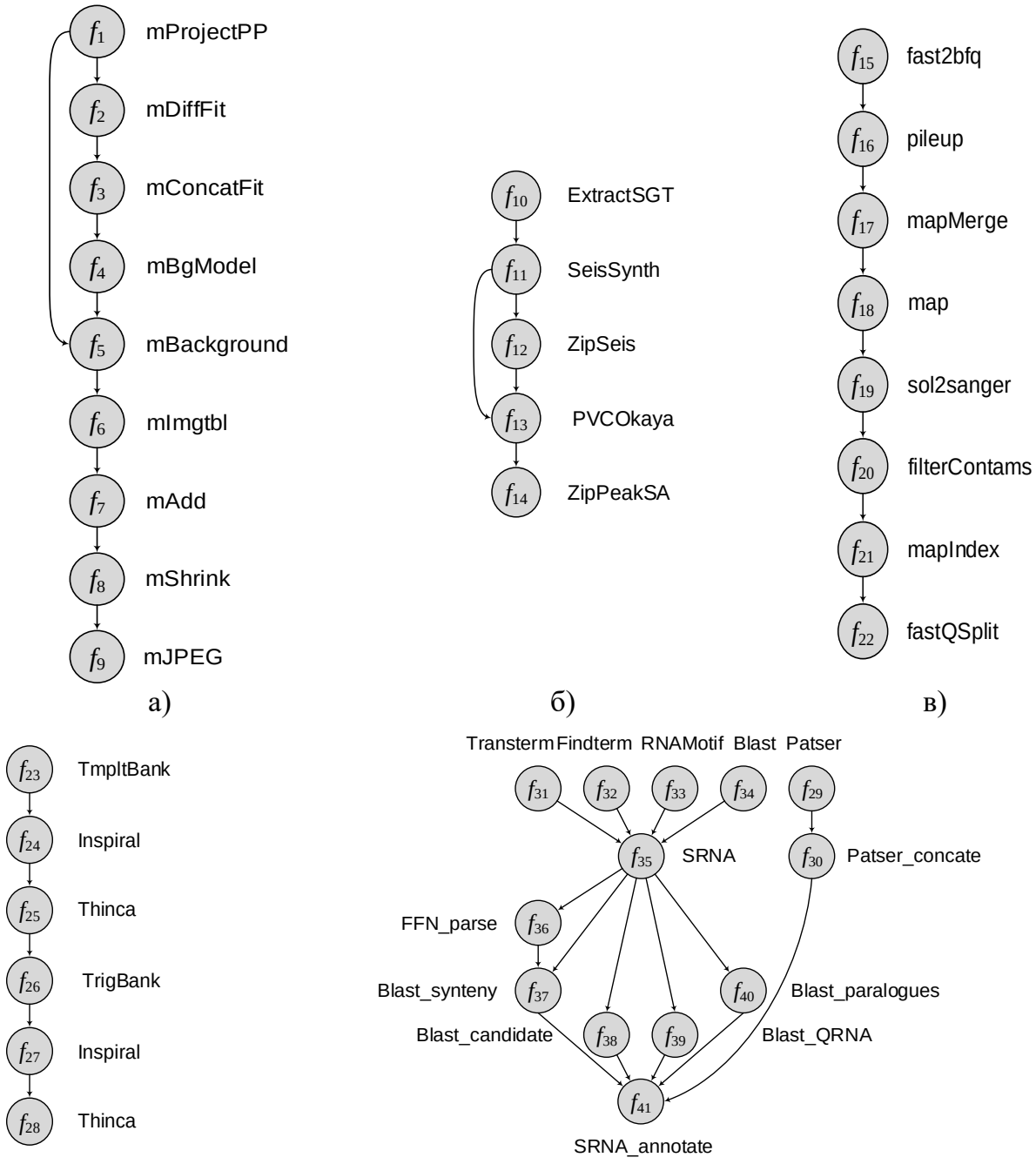


Рисунок Е.1 – Операции рабочих процессов Montage (а), CyberShake (б), Epigenomics (в), LIGO (г) и SIPHT (д)

Приложение Ж

Описание потоков рабочих процессов

Паспорт задания по выполнению рабочего процесса Montage в формате DAGMan.

```
JOB Montage1_1 Montage1_1.qs
JOB Montage1_2 Montage1_2.qs
JOB Montage1_3 Montage1_3.qs
JOB Montage1_4 Montage1_4.qs
JOB Montage1_5 Montage1_5.qs
JOB Montage1_6 Montage1_6.qs
JOB Montage1_7 Montage1_7.qs
JOB Montage1_8 Montage1_8.qs
JOB Montage1_9 Montage1_9.qs
JOB Montage1_10 Montage1_10.qs
JOB Montage1_11 Montage1_11.qs
JOB Montage1_12 Montage1_12.qs
JOB Montage1_13 Montage1_13.qs
JOB Montage1_14 Montage1_14.qs
JOB Montage1_15 Montage1_15.qs
JOB Montage1_16 Montage1_16.qs
JOB Montage1_17 Montage1_17.qs
JOB Montage1_18 Montage1_18.qs
JOB Montage1_19 Montage1_19.qs
JOB Montage1_20 Montage1_20.qs
JOB Montage1_21 Montage1_21.qs
JOB Montage1_22 Montage1_22.qs
JOB Montage1_23 Montage1_23.qs
JOB Montage1_24 Montage1_24.qs
JOB Montage1_25 Montage1_25.qs
JOB Montage1_26 Montage1_26.qs
JOB Montage1_27 Montage1_27.qs
JOB Montage1_28 Montage1_28.qs
JOB Montage1_29 Montage1_29.qs
JOB Montage1_30 Montage1_30.qs
JOB Montage1_31 Montage1_31.qs
```

```
JOB Montage1_32 Montage1_32.qs
JOB Montage1_33 Montage1_33.qs
JOB Montage1_34 Montage1_34.qs
JOB Montage1_35 Montage1_35.qs
JOB Montage1_36 Montage1_36.qs
JOB Montage1_37 Montage1_37.qs
JOB Montage1_38 Montage1_38.qs
PARENT Montage1_1 CHILD Montage1_9,Montage1_10,Montage1_27,
PARENT Montage1_2 CHILD Montage1_11,Montage1_12,Montage1_28,
PARENT Montage1_3 CHILD Montage1_13,Montage1_14,Montage1_29,
PARENT Montage1_4 CHILD Montage1_15,Montage1_16,Montage1_30,
PARENT Montage1_5 CHILD Montage1_17,Montage1_18,Montage1_31,
PARENT Montage1_6 CHILD Montage1_19,Montage1_20,Montage1_32,
PARENT Montage1_7 CHILD Montage1_21,Montage1_22,Montage1_33,
PARENT Montage1_8 CHILD Montage1_23,Montage1_24,Montage1_34,
PARENT Montage1_9 CHILD Montage1_25
PARENT Montage1_10 CHILD Montage1_25
PARENT Montage1_11 CHILD Montage1_25
PARENT Montage1_12 CHILD Montage1_25
PARENT Montage1_13 CHILD Montage1_25
PARENT Montage1_14 CHILD Montage1_25
PARENT Montage1_15 CHILD Montage1_25
PARENT Montage1_16 CHILD Montage1_25
PARENT Montage1_17 CHILD Montage1_25
PARENT Montage1_18 CHILD Montage1_25
PARENT Montage1_19 CHILD Montage1_25
PARENT Montage1_20 CHILD Montage1_25
PARENT Montage1_21 CHILD Montage1_25
PARENT Montage1_22 CHILD Montage1_25
PARENT Montage1_23 CHILD Montage1_25
PARENT Montage1_24 CHILD Montage1_25
PARENT Montage1_25 CHILD Montage1_26
PARENT Montage1_26 CHILD
Montage1_27,Montage1_28,Montage1_29,Montage1_30,Montage1_31,Mo
ntage1_32,Montage1_33,Montage1_34
PARENT Montage1_27 CHILD Montage1_35
PARENT Montage1_28 CHILD Montage1_35
PARENT Montage1_29 CHILD Montage1_35
PARENT Montage1_30 CHILD Montage1_35
```

```
PARENT Montage1_31 CHILD Montage1_35
PARENT Montage1_32 CHILD Montage1_35
PARENT Montage1_33 CHILD Montage1_35
PARENT Montage1_34 CHILD Montage1_35
PARENT Montage1_35 CHILD Montage1_36
PARENT Montage1_36 CHILD Montage1_37
PARENT Montage1_37 CHILD Montage1_38
```

Паспорт задания по выполнению рабочего процесса Montage в формате GridWay.

```
JOB Montage1_1 Montage1_1.jt
JOB Montage1_2 Montage1_2.jt
JOB Montage1_3 Montage1_3.jt
JOB Montage1_4 Montage1_4.jt
JOB Montage1_5 Montage1_5.jt
JOB Montage1_6 Montage1_6.jt
JOB Montage1_7 Montage1_7.jt
JOB Montage1_8 Montage1_8.jt
JOB Montage1_9 Montage1_9.jt
JOB Montage1_10 Montage1_10.jt
JOB Montage1_11 Montage1_11.jt
JOB Montage1_12 Montage1_12.jt
JOB Montage1_13 Montage1_13.jt
JOB Montage1_14 Montage1_14.jt
JOB Montage1_15 Montage1_15.jt
JOB Montage1_16 Montage1_16.jt
JOB Montage1_17 Montage1_17.jt
JOB Montage1_18 Montage1_18.jt
JOB Montage1_19 Montage1_19.jt
JOB Montage1_20 Montage1_20.jt
JOB Montage1_21 Montage1_21.jt
JOB Montage1_22 Montage1_22.jt
JOB Montage1_23 Montage1_23.jt
JOB Montage1_24 Montage1_24.jt
JOB Montage1_25 Montage1_25.jt
JOB Montage1_26 Montage1_26.jt
JOB Montage1_27 Montage1_27.jt
JOB Montage1_28 Montage1_28.jt
```

```
JOB Montage1_29 Montage1_29.jt
JOB Montage1_30 Montage1_30.jt
JOB Montage1_31 Montage1_31.jt
JOB Montage1_32 Montage1_32.jt
JOB Montage1_33 Montage1_33.jt
JOB Montage1_34 Montage1_34.jt
JOB Montage1_35 Montage1_35.jt
JOB Montage1_36 Montage1_36.jt
JOB Montage1_37 Montage1_37.jt
JOB Montage1_38 Montage1_38.jt
PARENT Montage1_1 CHILD Montage1_9,Montage1_10,Montage1_27,
PARENT Montage1_2 CHILD Montage1_11,Montage1_12,Montage1_28,
PARENT Montage1_3 CHILD Montage1_13,Montage1_14,Montage1_29,
PARENT Montage1_4 CHILD Montage1_15,Montage1_16,Montage1_30,
PARENT Montage1_5 CHILD Montage1_17,Montage1_18,Montage1_31,
PARENT Montage1_6 CHILD Montage1_19,Montage1_20,Montage1_32,
PARENT Montage1_7 CHILD Montage1_21,Montage1_22,Montage1_33,
PARENT Montage1_8 CHILD Montage1_23,Montage1_24,Montage1_34,
PARENT Montage1_9 CHILD Montage1_25
PARENT Montage1_10 CHILD Montage1_25
PARENT Montage1_11 CHILD Montage1_25
PARENT Montage1_12 CHILD Montage1_25
PARENT Montage1_13 CHILD Montage1_25
PARENT Montage1_14 CHILD Montage1_25
PARENT Montage1_15 CHILD Montage1_25
PARENT Montage1_16 CHILD Montage1_25
PARENT Montage1_17 CHILD Montage1_25
PARENT Montage1_18 CHILD Montage1_25
PARENT Montage1_19 CHILD Montage1_25
PARENT Montage1_20 CHILD Montage1_25
PARENT Montage1_21 CHILD Montage1_25
PARENT Montage1_22 CHILD Montage1_25
PARENT Montage1_23 CHILD Montage1_25
PARENT Montage1_24 CHILD Montage1_25
PARENT Montage1_25 CHILD Montage1_26
PARENT Montage1_26 CHILD
Montage1_27,Montage1_28,Montage1_29,Montage1_30,Montage1_31,Mo
ntage1_32,Montage1_33,Montage1_34
PARENT Montage1_27 CHILD Montage1_35
```

```
PARENT Montage1_28 CHILD Montage1_35
PARENT Montage1_29 CHILD Montage1_35
PARENT Montage1_30 CHILD Montage1_35
PARENT Montage1_31 CHILD Montage1_35
PARENT Montage1_32 CHILD Montage1_35
PARENT Montage1_33 CHILD Montage1_35
PARENT Montage1_34 CHILD Montage1_35
PARENT Montage1_35 CHILD Montage1_36
PARENT Montage1_36 CHILD Montage1_37
PARENT Montage1_37 CHILD Montage1_38
```

Паспорт задания по выполнению по выполнению модуля рабочего процесса Montage в формате DAGMan.

```
executable = /home/usr/bin/mProjectPP.exe
input = mProjectPP1.in
output = mProjectPP1.out,mProjectPP2.out,mProjectPP3.out
error = mProjectPP.err.$(cluster)
log = mProjectPP.log
universe = vanilla
queue
```

Паспорт задания по выполнению по выполнению модуля рабочего процесса Montage в формате GridWay.

```
EXECUTABLE = /home/usr/bin/mProjectPP.exe
ARGUMENTS = "$RANDOM"
INPUT_FILES = mProjectPP1.in
STDOUT_FILES = mProjectPP1.out,mProjectPP2.out,mProjectPP3.out
```

Паспорт задания по выполнению рабочего процесса CyberShake в формате DAGMan.

```
JOB CyberShake1_1 CyberShake1_1.qs
JOB CyberShake1_2 CyberShake1_2.qs
JOB CyberShake1_3 CyberShake1_3.qs
JOB CyberShake1_4 CyberShake1_4.qs
```

```
JOB CyberShake1_5 CyberShake1_5.qs
JOB CyberShake1_6 CyberShake1_6.qs
JOB CyberShake1_7 CyberShake1_7.qs
JOB CyberShake1_8 CyberShake1_8.qs
JOB CyberShake1_9 CyberShake1_9.qs
JOB CyberShake1_10 CyberShake1_10.qs
JOB CyberShake1_11 CyberShake1_11.qs
JOB CyberShake1_12 CyberShake1_12.qs
JOB CyberShake1_13 CyberShake1_13.qs
JOB CyberShake1_14 CyberShake1_14.qs
JOB CyberShake1_15 CyberShake1_15.qs
JOB CyberShake1_16 CyberShake1_16.qs
JOB CyberShake1_17 CyberShake1_17.qs
JOB CyberShake1_18 CyberShake1_18.qs
JOB CyberShake1_19 CyberShake1_19.qs
JOB CyberShake1_20 CyberShake1_20.qs
JOB CyberShake1_21 CyberShake1_21.qs
JOB CyberShake1_22 CyberShake1_22.qs
JOB CyberShake1_23 CyberShake1_23.qs
JOB CyberShake1_24 CyberShake1_24.qs
JOB CyberShake1_25 CyberShake1_25.qs
JOB CyberShake1_26 CyberShake1_26.qs
JOB CyberShake1_27 CyberShake1_27.qs
JOB CyberShake1_28 CyberShake1_28.qs
JOB CyberShake1_29 CyberShake1_29.qs
JOB CyberShake1_30 CyberShake1_30.qs
JOB CyberShake1_31 CyberShake1_31.qs
JOB CyberShake1_32 CyberShake1_32.qs
JOB CyberShake1_33 CyberShake1_33.qs
JOB CyberShake1_34 CyberShake1_34.qs
JOB CyberShake1_35 CyberShake1_35.qs
JOB CyberShake1_36 CyberShake1_36.qs
JOB CyberShake1_37 CyberShake1_37.qs
JOB CyberShake1_38 CyberShake1_38.qs
PARENT CyberShake1_1 CHILD
CyberShake1_5,CyberShake1_6,CyberShake1_7,CyberShake1_8
PARENT CyberShake1_2 CHILD
CyberShake1_9,CyberShake1_10,CyberShake1_11,CyberShake1_12
PARENT CyberShake1_3 CHILD
```

```
CyberShake1_13,CyberShake1_14,CyberShake1_15,CyberShake1_16
PARENT CyberShake1_4 CHILD
CyberShake1_17,CyberShake1_18,CyberShake1_19,CyberShake1_20
PARENT CyberShake1_5 CHILD CyberShake1_22,CyberShake1_21
PARENT CyberShake1_6 CHILD CyberShake1_23,CyberShake1_21
PARENT CyberShake1_7 CHILD CyberShake1_24,CyberShake1_21
PARENT CyberShake1_8 CHILD CyberShake1_25,CyberShake1_21
PARENT CyberShake1_9 CHILD CyberShake1_26,CyberShake1_21
PARENT CyberShake1_10 CHILD CyberShake1_27,CyberShake1_21
PARENT CyberShake1_11 CHILD CyberShake1_28,CyberShake1_21
PARENT CyberShake1_12 CHILD CyberShake1_29,CyberShake1_21
PARENT CyberShake1_13 CHILD CyberShake1_30,CyberShake1_21
PARENT CyberShake1_14 CHILD CyberShake1_31,CyberShake1_21
PARENT CyberShake1_15 CHILD CyberShake1_32,CyberShake1_21
PARENT CyberShake1_16 CHILD CyberShake1_33,CyberShake1_21
PARENT CyberShake1_17 CHILD CyberShake1_34,CyberShake1_21
PARENT CyberShake1_18 CHILD CyberShake1_35,CyberShake1_21
PARENT CyberShake1_19 CHILD CyberShake1_36,CyberShake1_21
PARENT CyberShake1_20 CHILD CyberShake1_37,CyberShake1_21
PARENT CyberShake1_22 CHILD CyberShake1_38
PARENT CyberShake1_23 CHILD CyberShake1_38
PARENT CyberShake1_24 CHILD CyberShake1_38
PARENT CyberShake1_25 CHILD CyberShake1_38
PARENT CyberShake1_26 CHILD CyberShake1_38
PARENT CyberShake1_27 CHILD CyberShake1_38
PARENT CyberShake1_28 CHILD CyberShake1_38
PARENT CyberShake1_29 CHILD CyberShake1_38
PARENT CyberShake1_30 CHILD CyberShake1_38
PARENT CyberShake1_31 CHILD CyberShake1_38
PARENT CyberShake1_32 CHILD CyberShake1_38
PARENT CyberShake1_33 CHILD CyberShake1_38
PARENT CyberShake1_34 CHILD CyberShake1_38
PARENT CyberShake1_35 CHILD CyberShake1_38
PARENT CyberShake1_36 CHILD CyberShake1_38
PARENT CyberShake1_37 CHILD CyberShake1_38
```

Паспорт задания по выполнению рабочего процесса CyberShake в формате GridWay.

```
JOB CyberShake1_1 CyberShake1_1.qs
JOB CyberShake1_2 CyberShake1_2.qs
JOB CyberShake1_3 CyberShake1_3.qs
JOB CyberShake1_4 CyberShake1_4.qs
JOB CyberShake1_5 CyberShake1_5.qs
JOB CyberShake1_6 CyberShake1_6.qs
JOB CyberShake1_7 CyberShake1_7.qs
JOB CyberShake1_8 CyberShake1_8.qs
JOB CyberShake1_9 CyberShake1_9.qs
JOB CyberShake1_10 CyberShake1_10.qs
JOB CyberShake1_11 CyberShake1_11.qs
JOB CyberShake1_12 CyberShake1_12.qs
JOB CyberShake1_13 CyberShake1_13.qs
JOB CyberShake1_14 CyberShake1_14.qs
JOB CyberShake1_15 CyberShake1_15.qs
JOB CyberShake1_16 CyberShake1_16.qs
JOB CyberShake1_17 CyberShake1_17.qs
JOB CyberShake1_18 CyberShake1_18.qs
JOB CyberShake1_19 CyberShake1_19.qs
JOB CyberShake1_20 CyberShake1_20.qs
JOB CyberShake1_21 CyberShake1_21.qs
JOB CyberShake1_22 CyberShake1_22.qs
JOB CyberShake1_23 CyberShake1_23.qs
JOB CyberShake1_24 CyberShake1_24.qs
JOB CyberShake1_25 CyberShake1_25.qs
JOB CyberShake1_26 CyberShake1_26.qs
JOB CyberShake1_27 CyberShake1_27.qs
JOB CyberShake1_28 CyberShake1_28.qs
JOB CyberShake1_29 CyberShake1_29.qs
JOB CyberShake1_30 CyberShake1_30.qs
JOB CyberShake1_31 CyberShake1_31.qs
JOB CyberShake1_32 CyberShake1_32.qs
JOB CyberShake1_33 CyberShake1_33.qs
JOB CyberShake1_34 CyberShake1_34.qs
JOB CyberShake1_35 CyberShake1_35.qs
JOB CyberShake1_36 CyberShake1_36.qs
```

```
JOB CyberShake1_37 CyberShake1_37.qs
JOB CyberShake1_38 CyberShake1_38.qs
PARENT CyberShake1_1 CHILD
CyberShake1_5,CyberShake1_6,CyberShake1_7,CyberShake1_8
PARENT CyberShake1_2 CHILD
CyberShake1_9,CyberShake1_10,CyberShake1_11,CyberShake1_12
PARENT CyberShake1_3 CHILD
CyberShake1_13,CyberShake1_14,CyberShake1_15,CyberShake1_16
PARENT CyberShake1_4 CHILD
CyberShake1_17,CyberShake1_18,CyberShake1_19,CyberShake1_20
PARENT CyberShake1_5 CHILD CyberShake1_22,CyberShake1_21
PARENT CyberShake1_6 CHILD CyberShake1_23,CyberShake1_21
PARENT CyberShake1_7 CHILD CyberShake1_24,CyberShake1_21
PARENT CyberShake1_8 CHILD CyberShake1_25,CyberShake1_21
PARENT CyberShake1_9 CHILD CyberShake1_26,CyberShake1_21
PARENT CyberShake1_10 CHILD CyberShake1_27,CyberShake1_21
PARENT CyberShake1_11 CHILD CyberShake1_28,CyberShake1_21
PARENT CyberShake1_12 CHILD CyberShake1_29,CyberShake1_21
PARENT CyberShake1_13 CHILD CyberShake1_30,CyberShake1_21
PARENT CyberShake1_14 CHILD CyberShake1_31,CyberShake1_21
PARENT CyberShake1_15 CHILD CyberShake1_32,CyberShake1_21
PARENT CyberShake1_16 CHILD CyberShake1_33,CyberShake1_21
PARENT CyberShake1_17 CHILD CyberShake1_34,CyberShake1_21
PARENT CyberShake1_18 CHILD CyberShake1_35,CyberShake1_21
PARENT CyberShake1_19 CHILD CyberShake1_36,CyberShake1_21
PARENT CyberShake1_20 CHILD CyberShake1_37,CyberShake1_21
PARENT CyberShake1_22 CHILD CyberShake1_38
PARENT CyberShake1_23 CHILD CyberShake1_38
PARENT CyberShake1_24 CHILD CyberShake1_38
PARENT CyberShake1_25 CHILD CyberShake1_38
PARENT CyberShake1_26 CHILD CyberShake1_38
PARENT CyberShake1_27 CHILD CyberShake1_38
PARENT CyberShake1_28 CHILD CyberShake1_38
PARENT CyberShake1_29 CHILD CyberShake1_38
PARENT CyberShake1_30 CHILD CyberShake1_38
PARENT CyberShake1_31 CHILD CyberShake1_38
PARENT CyberShake1_32 CHILD CyberShake1_38
PARENT CyberShake1_33 CHILD CyberShake1_38
PARENT CyberShake1_34 CHILD CyberShake1_38
```

```
PARENT CyberShake1_35 CHILD CyberShake1_38
PARENT CyberShake1_36 CHILD CyberShake1_38
PARENT CyberShake1_37 CHILD CyberShake1_38
```

Паспорт задания по выполнению по выполнению модуля рабочего процесса CyberShake в формате DAGMan.

```
executable = /home/usr/bin/ExtractSGT.exe
input = ExtractSGT1.in
output =
ExtractSGT1.out,ExtractSGT2.out,ExtractSGT3.out,ExtractSGT4.ou
t
error = ExtractSGT.err.$(cluster)
log = ExtractSGT.log
universe = vanilla
queue
```

Паспорт задания по выполнению по выполнению модуля рабочего процесса CyberShake в формате GridWay.

```
EXECUTABLE = /home/usr/bin/ExtractSGT.exe
ARGUMENTS = "$RANDOM"
INPUT_FILES = ExtractSGT1.in
STDOUT_FILES =
ExtractSGT1.out,ExtractSGT2.out,ExtractSGT3.out,ExtractSGT4.ou
t
```

Паспорт задания по выполнению рабочего процесса Epigenomics в формате DAGMan.

```
JOB Epigenomics1_1 Epigenomics1_1.qs
JOB Epigenomics1_2 Epigenomics1_2.qs
JOB Epigenomics1_3 Epigenomics1_3.qs
JOB Epigenomics1_4 Epigenomics1_4.qs
JOB Epigenomics1_5 Epigenomics1_5.qs
JOB Epigenomics1_6 Epigenomics1_6.qs
JOB Epigenomics1_7 Epigenomics1_7.qs
```

```
JOB Epigenomics1_8 Epigenomics1_8.qs
JOB Epigenomics1_9 Epigenomics1_9.qs
JOB Epigenomics1_10 Epigenomics1_10.qs
JOB Epigenomics1_11 Epigenomics1_11.qs
JOB Epigenomics1_12 Epigenomics1_12.qs
JOB Epigenomics1_13 Epigenomics1_13.qs
JOB Epigenomics1_14 Epigenomics1_14.qs
JOB Epigenomics1_15 Epigenomics1_15.qs
JOB Epigenomics1_16 Epigenomics1_16.qs
JOB Epigenomics1_17 Epigenomics1_17.qs
JOB Epigenomics1_18 Epigenomics1_18.qs
JOB Epigenomics1_19 Epigenomics1_19.qs
JOB Epigenomics1_20 Epigenomics1_20.qs
JOB Epigenomics1_21 Epigenomics1_21.qs
JOB Epigenomics1_22 Epigenomics1_22.qs
JOB Epigenomics1_23 Epigenomics1_23.qs
JOB Epigenomics1_24 Epigenomics1_24.qs
JOB Epigenomics1_25 Epigenomics1_25.qs
JOB Epigenomics1_26 Epigenomics1_26.qs
JOB Epigenomics1_27 Epigenomics1_27.qs
JOB Epigenomics1_28 Epigenomics1_28.qs
JOB Epigenomics1_29 Epigenomics1_29.qs
JOB Epigenomics1_30 Epigenomics1_30.qs
JOB Epigenomics1_31 Epigenomics1_31.qs
JOB Epigenomics1_32 Epigenomics1_32.qs
JOB Epigenomics1_33 Epigenomics1_33.qs
JOB Epigenomics1_34 Epigenomics1_34.qs
JOB Epigenomics1_35 Epigenomics1_35.qs
JOB Epigenomics1_36 Epigenomics1_36.qs
JOB Epigenomics1_37 Epigenomics1_37.qs
JOB Epigenomics1_38 Epigenomics1_38.qs
JOB Epigenomics1_39 Epigenomics1_39.qs
JOB Epigenomics1_40 Epigenomics1_40.qs
JOB Epigenomics1_41 Epigenomics1_41.qs
JOB Epigenomics1_42 Epigenomics1_42.qs
JOB Epigenomics1_43 Epigenomics1_43.qs
JOB Epigenomics1_44 Epigenomics1_44.qs
JOB Epigenomics1_45 Epigenomics1_45.qs
JOB Epigenomics1_46 Epigenomics1_46.qs
```

```
JOB Epigenomics1_47 Epigenomics1_47.qs
JOB Epigenomics1_48 Epigenomics1_48.qs
JOB Epigenomics1_49 Epigenomics1_49.qs
JOB Epigenomics1_50 Epigenomics1_50.qs
JOB Epigenomics1_51 Epigenomics1_51.qs
JOB Epigenomics1_52 Epigenomics1_52.qs
PARENT Epigenomics1_1 CHILD
Epigenomics1_2,Epigenomics1_3,Epigenomics1_4,Epigenomics1_5,Ep
igenomics1_6,Epigenomics1_7,Epigenomics1_8,Epigenomics1_9,Epig
enomics1_10,Epigenomics1_11,Epigenomics1_12,Epigenomics1_13
PARENT Epigenomics1_2 CHILD Epigenomics1_14
PARENT Epigenomics1_3 CHILD Epigenomics1_15
PARENT Epigenomics1_4 CHILD Epigenomics1_16
PARENT Epigenomics1_5 CHILD Epigenomics1_17
PARENT Epigenomics1_6 CHILD Epigenomics1_18
PARENT Epigenomics1_7 CHILD Epigenomics1_19
PARENT Epigenomics1_8 CHILD Epigenomics1_20
PARENT Epigenomics1_9 CHILD Epigenomics1_21
PARENT Epigenomics1_10 CHILD Epigenomics1_22
PARENT Epigenomics1_11 CHILD Epigenomics1_23
PARENT Epigenomics1_12 CHILD Epigenomics1_24
PARENT Epigenomics1_13 CHILD Epigenomics1_25
PARENT Epigenomics1_14 CHILD Epigenomics1_26
PARENT Epigenomics1_15 CHILD Epigenomics1_27
PARENT Epigenomics1_16 CHILD Epigenomics1_28
PARENT Epigenomics1_17 CHILD Epigenomics1_29
PARENT Epigenomics1_18 CHILD Epigenomics1_30
PARENT Epigenomics1_19 CHILD Epigenomics1_31
PARENT Epigenomics1_20 CHILD Epigenomics1_32
PARENT Epigenomics1_21 CHILD Epigenomics1_33
PARENT Epigenomics1_22 CHILD Epigenomics1_34
PARENT Epigenomics1_23 CHILD Epigenomics1_35
PARENT Epigenomics1_24 CHILD Epigenomics1_36
PARENT Epigenomics1_25 CHILD Epigenomics1_37
PARENT Epigenomics1_26 CHILD Epigenomics1_38
PARENT Epigenomics1_27 CHILD Epigenomics1_39
PARENT Epigenomics1_28 CHILD Epigenomics1_40
PARENT Epigenomics1_29 CHILD Epigenomics1_41
PARENT Epigenomics1_30 CHILD Epigenomics1_42
```

```
PARENT Epigenomics1_31 CHILD Epigenomics1_43
PARENT Epigenomics1_32 CHILD Epigenomics1_44
PARENT Epigenomics1_33 CHILD Epigenomics1_45
PARENT Epigenomics1_34 CHILD Epigenomics1_46
PARENT Epigenomics1_35 CHILD Epigenomics1_47
PARENT Epigenomics1_36 CHILD Epigenomics1_48
PARENT Epigenomics1_37 CHILD Epigenomics1_49
PARENT Epigenomics1_38,Epigenomics1_39,Epigenomics1_40,
Epigenomics1_41,Epigenomics1_42,Epigenomics1_43,Epigenomics1_4
4,Epigenomics1_45,Epigenomics1_46,Epigenomics1_47,Epigenomics1
_48,Epigenomics1_49 CHILD Epigenomics1_50
PARENT Epigenomics1_50 CHILD Epigenomics1_51
PARENT Epigenomics1_51 CHILD Epigenomics1_52
```

Паспорт задания по выполнению рабочего процесса Epigenomics в формате GridWay.

```
JOB Epigenomics1_1 Epigenomics1_1.jt
JOB Epigenomics1_2 Epigenomics1_2.jt
JOB Epigenomics1_3 Epigenomics1_3.jt
JOB Epigenomics1_4 Epigenomics1_4.jt
JOB Epigenomics1_5 Epigenomics1_5.jt
JOB Epigenomics1_6 Epigenomics1_6.jt
JOB Epigenomics1_7 Epigenomics1_7.jt
JOB Epigenomics1_8 Epigenomics1_8.jt
JOB Epigenomics1_9 Epigenomics1_9.jt
JOB Epigenomics1_10 Epigenomics1_10.jt
JOB Epigenomics1_11 Epigenomics1_11.jt
JOB Epigenomics1_12 Epigenomics1_12.jt
JOB Epigenomics1_13 Epigenomics1_13.jt
JOB Epigenomics1_14 Epigenomics1_14.jt
JOB Epigenomics1_15 Epigenomics1_15.jt
JOB Epigenomics1_16 Epigenomics1_16.jt
JOB Epigenomics1_17 Epigenomics1_17.jt
JOB Epigenomics1_18 Epigenomics1_18.jt
JOB Epigenomics1_19 Epigenomics1_19.jt
JOB Epigenomics1_20 Epigenomics1_20.jt
JOB Epigenomics1_21 Epigenomics1_21.jt
JOB Epigenomics1_22 Epigenomics1_22.jt
```

```
JOB Epigenomics1_23 Epigenomics1_23.jt
JOB Epigenomics1_24 Epigenomics1_24.jt
JOB Epigenomics1_25 Epigenomics1_25.jt
JOB Epigenomics1_26 Epigenomics1_26.jt
JOB Epigenomics1_27 Epigenomics1_27.jt
JOB Epigenomics1_28 Epigenomics1_28.jt
JOB Epigenomics1_29 Epigenomics1_29.jt
JOB Epigenomics1_30 Epigenomics1_30.jt
JOB Epigenomics1_31 Epigenomics1_31.jt
JOB Epigenomics1_32 Epigenomics1_32.jt
JOB Epigenomics1_33 Epigenomics1_33.jt
JOB Epigenomics1_34 Epigenomics1_34.jt
JOB Epigenomics1_35 Epigenomics1_35.jt
JOB Epigenomics1_36 Epigenomics1_36.jt
JOB Epigenomics1_37 Epigenomics1_37.jt
JOB Epigenomics1_38 Epigenomics1_38.jt
JOB Epigenomics1_39 Epigenomics1_39.jt
JOB Epigenomics1_40 Epigenomics1_40.jt
JOB Epigenomics1_41 Epigenomics1_41.jt
JOB Epigenomics1_42 Epigenomics1_42.jt
JOB Epigenomics1_43 Epigenomics1_43.jt
JOB Epigenomics1_44 Epigenomics1_44.jt
JOB Epigenomics1_45 Epigenomics1_45.jt
JOB Epigenomics1_46 Epigenomics1_46.jt
JOB Epigenomics1_47 Epigenomics1_47.jt
JOB Epigenomics1_48 Epigenomics1_48.jt
JOB Epigenomics1_49 Epigenomics1_49.jt
JOB Epigenomics1_50 Epigenomics1_50.jt
JOB Epigenomics1_51 Epigenomics1_51.jt
JOB Epigenomics1_52 Epigenomics1_52.jt
PARENT Epigenomics1_1 CHILD
Epigenomics1_2,Epigenomics1_3,Epigenomics1_4,Epigenomics1_5,Ep
igenomics1_6,Epigenomics1_7,Epigenomics1_8,Epigenomics1_9,Epig
enomics1_10,Epigenomics1_11,Epigenomics1_12,Epigenomics1_13
PARENT Epigenomics1_2 CHILD Epigenomics1_14
PARENT Epigenomics1_3 CHILD Epigenomics1_15
PARENT Epigenomics1_4 CHILD Epigenomics1_16
PARENT Epigenomics1_5 CHILD Epigenomics1_17
PARENT Epigenomics1_6 CHILD Epigenomics1_18
```

```
PARENT Epigenomics1_7 CHILD Epigenomics1_19
PARENT Epigenomics1_8 CHILD Epigenomics1_20
PARENT Epigenomics1_9 CHILD Epigenomics1_21
PARENT Epigenomics1_10 CHILD Epigenomics1_22
PARENT Epigenomics1_11 CHILD Epigenomics1_23
PARENT Epigenomics1_12 CHILD Epigenomics1_24
PARENT Epigenomics1_13 CHILD Epigenomics1_25
PARENT Epigenomics1_14 CHILD Epigenomics1_26
PARENT Epigenomics1_15 CHILD Epigenomics1_27
PARENT Epigenomics1_16 CHILD Epigenomics1_28
PARENT Epigenomics1_17 CHILD Epigenomics1_29
PARENT Epigenomics1_18 CHILD Epigenomics1_30
PARENT Epigenomics1_19 CHILD Epigenomics1_31
PARENT Epigenomics1_20 CHILD Epigenomics1_32
PARENT Epigenomics1_21 CHILD Epigenomics1_33
PARENT Epigenomics1_22 CHILD Epigenomics1_34
PARENT Epigenomics1_23 CHILD Epigenomics1_35
PARENT Epigenomics1_24 CHILD Epigenomics1_36
PARENT Epigenomics1_25 CHILD Epigenomics1_37
PARENT Epigenomics1_26 CHILD Epigenomics1_38
PARENT Epigenomics1_27 CHILD Epigenomics1_39
PARENT Epigenomics1_28 CHILD Epigenomics1_40
PARENT Epigenomics1_29 CHILD Epigenomics1_41
PARENT Epigenomics1_30 CHILD Epigenomics1_42
PARENT Epigenomics1_31 CHILD Epigenomics1_43
PARENT Epigenomics1_32 CHILD Epigenomics1_44
PARENT Epigenomics1_33 CHILD Epigenomics1_45
PARENT Epigenomics1_34 CHILD Epigenomics1_46
PARENT Epigenomics1_35 CHILD Epigenomics1_47
PARENT Epigenomics1_36 CHILD Epigenomics1_48
PARENT Epigenomics1_37 CHILD Epigenomics1_49
PARENT Epigenomics1_38,Epigenomics1_39,Epigenomics1_40,
Epigenomics1_41,Epigenomics1_42,Epigenomics1_43,Epigenomics1_4
4,Epigenomics1_45,Epigenomics1_46,Epigenomics1_47,Epigenomics1
_48,Epigenomics1_49 CHILD Epigenomics1_50
PARENT Epigenomics1_50 CHILD Epigenomics1_51
PARENT Epigenomics1_51 CHILD Epigenomics1_52
```

Паспорт задания по выполнению по выполнению модуля рабочего процесса Epigenomics в формате DAGMan.

```
executable = /home/usr/bin/fastQSplit.exe
input = fastQSplit.in
output =
fastQSplit1.out,fastQSplit2.out,fastQSplit3.out,fastQSplit4.ou
t,fastQSplit5.out,fastQSplit6.out,fastQSplit7.out,fastQSplit8.
out,fastQSplit9.out,fastQSplit10.out,fastQSplit11.out,fastQSpl
it12.out
error = fastQSplit.err.$(cluster)
log = fastQSplit.log
universe = vanilla
queue
```

Паспорт задания по выполнению по выполнению модуля рабочего процесса Epigenomics в формате DAGMan.

```
EXECUTABLE = /home/usr/bin/fastQSplit.exe
ARGUMENTS = "$RANDOM"
INPUT_FILES = fastQSplit.in
STDOUT_FILES =
fastQSplit1.out,fastQSplit2.out,fastQSplit3.out,fastQSplit4.ou
t,fastQSplit5.out,fastQSplit6.out,fastQSplit7.out,fastQSplit8.
out,fastQSplit9.out,fastQSplit10.out,fastQSplit11.out,fastQSpl
it12.out
```

Паспорт задания по выполнению рабочего процесса LIGO в формате DAGMan.

```
JOB LIGO1_1 LIGO1_1.qs
JOB LIGO1_2 LIGO1_2.qs
JOB LIGO1_3 LIGO1_3.qs
JOB LIGO1_4 LIGO1_4.qs
JOB LIGO1_5 LIGO1_5.qs
JOB LIGO1_6 LIGO1_6.qs
```

```
PARENT LIGO1_1 CHILD LIGO1_2
PARENT LIGO1_2 CHILD LIGO1_3
PARENT LIGO1_3 CHILD LIGO1_4
PARENT LIGO1_4 CHILD LIGO1_5
PARENT LIGO1_5 CHILD LIGO1_6
```

Паспорт задания по выполнению рабочего процесса LIGO в формате GridWay.

```
JOB LIGO1_1 LIGO1_1.jt
JOB LIGO1_2 LIGO1_2.jt
JOB LIGO1_3 LIGO1_3.jt
JOB LIGO1_4 LIGO1_4.jt
JOB LIGO1_5 LIGO1_5.jt
JOB LIGO1_6 LIGO1_6.jt
PARENT LIGO1_1 CHILD LIGO1_2
PARENT LIGO1_2 CHILD LIGO1_3
PARENT LIGO1_3 CHILD LIGO1_4
PARENT LIGO1_4 CHILD LIGO1_5
PARENT LIGO1_5 CHILD LIGO1_6
```

Паспорт задания по выполнению по выполнению модуля рабочего процесса LIGO в формате DAGMan.

```
executable = /home/usr/bin/TmpltBank.exe
input = TmpltBank1.in
output = TmpltBank1.out
error = TmpltBank.err.$(cluster)
log = TmpltBank.log
universe = vanilla
queue
```

Паспорт задания по выполнению по выполнению модуля рабочего процесса LIGO в формате GridWay.

```
EXECUTABLE = /home/usr/bin/TmpltBank.exe
ARGUMENTS = "$RANDOM"
```

```
INPUT_FILES = TmpltBank1.in
STDOUT_FILES = TmpltBank1.out
```

Паспорт задания по выполнению рабочего процесса SIPHT в формате DAGMan.

```
JOB SIPHT1_1 SIPHT1_1.qs
JOB SIPHT1_2 SIPHT1_2.qs
JOB SIPHT1_3 SIPHT1_3.qs
JOB SIPHT1_4 SIPHT1_4.qs
JOB SIPHT1_5 SIPHT1_5.qs
JOB SIPHT1_6 SIPHT1_6.qs
JOB SIPHT1_7 SIPHT1_7.qs
JOB SIPHT1_8 SIPHT1_8.qs
JOB SIPHT1_9 SIPHT1_9.qs
JOB SIPHT1_10 SIPHT1_10.qs
JOB SIPHT1_11 SIPHT1_11.qs
JOB SIPHT1_12 SIPHT1_12.qs
JOB SIPHT1_13 SIPHT1_13.qs
JOB SIPHT1_14 SIPHT1_14.qs
JOB SIPHT1_15 SIPHT1_15.qs
JOB SIPHT1_16 SIPHT1_16.qs
PARENT SIPHT1_1,SIPHT1_2,SIPHT1_3,SIPHT1_4 CHILD SIPHT1_9
PARENT SIPHT1_5,SIPHT1_6,SIPHT1_7,SIPHT1_8 CHILD SIPHT1_10
PARENT SIPHT1_9 CHILD
SIPHT1_11,SIPHT1_12,SIPHT1_13,SIPHT1_14,SIPHT1_15
PARENT SIPHT1_10 CHILD SIPHT1_16
PARENT SIPHT1_11 CHILD SIPHT1_12
PARENT SIPHT1_12,SIPHT1_13,SIPHT1_14,SIPHT1_15 CHILD SIPHT1_16
```

Паспорт задания по выполнению рабочего процесса SIPHT в формате GridWay.

```
JOB SIPHT1_1 SIPHT1_1.jt
JOB SIPHT1_2 SIPHT1_2.jt
JOB SIPHT1_3 SIPHT1_3.jt
JOB SIPHT1_4 SIPHT1_4.jt
JOB SIPHT1_5 SIPHT1_5.jt
```

```
JOB SIPHT1_6 SIPHT1_6.jt
JOB SIPHT1_7 SIPHT1_7.jt
JOB SIPHT1_8 SIPHT1_8.jt
JOB SIPHT1_9 SIPHT1_9.jt
JOB SIPHT1_10 SIPHT1_10.jt
JOB SIPHT1_11 SIPHT1_11.jt
JOB SIPHT1_12 SIPHT1_12.jt
JOB SIPHT1_13 SIPHT1_13.jt
JOB SIPHT1_14 SIPHT1_14.jt
JOB SIPHT1_15 SIPHT1_15.jt
JOB SIPHT1_16 SIPHT1_16.jt
PARENT SIPHT1_1,SIPHT1_2,SIPHT1_3,SIPHT1_4 CHILD SIPHT1_9
PARENT SIPHT1_5,SIPHT1_6,SIPHT1_7,SIPHT1_8 CHILD SIPHT1_10
PARENT SIPHT1_9 CHILD
SIPHT1_11,SIPHT1_12,SIPHT1_13,SIPHT1_14,SIPHT1_15
PARENT SIPHT1_10 CHILD SIPHT1_16
PARENT SIPHT1_11 CHILD SIPHT1_12
PARENT SIPHT1_12,SIPHT1_13,SIPHT1_14,SIPHT1_15 CHILD SIPHT1_16
```

Паспорт задания по выполнению по выполнению модуля рабочего процесса SIPHT в формате DAGMan.

```
executable = /home/usr/bin/Transterm.exe
input = Transterm.in
output = Transterm.out
error = Transterm.err.$(cluster)
log = Transterm.log
universe = vanilla
queue
```

Паспорт задания по выполнению по выполнению модуля рабочего процесса SIPHT в формате GridWay.

```
EXECUTABLE   = /home/usr/bin/Transterm.exe
ARGUMENTS = "$RANDOM"
INPUT_FILES = Transterm.in
STDOUT_FILES = Transterm.out
```

Приложение З

Характеристики потоков заданий

В таблице З.1 приведены общие характеристики потоков рабочих процессов Montage, CyberShake, Epigenomics, LIGO и SIPHT. Детализированная информация по выполнению каждой операции рабочих процессов на 1 ядре узла пула 1 приведена в таблицах З.2-З.6.

Таблица З.1 – Характеристики потоков рабочих процессов

Рабочий процесс	Число запусков модулей	Чтение данных, Гбайт	Запись данных, Гбайт
Montage	1380	1.5345	0.7760
CyberShake	1680	0.8229	0.2531
Epigenomics	1560	0.7921	0.4756
LIGO	1680	1.0804	0.0001
SIPHT	960	1.6525	1.4805

Таблица З.2 – Операции и модули рабочего процесса Montage: 1 ядро узла пула 1

Операция	Модуль	Число запусков модуля	Среднее время выполнения, с	Чтение данных, Гбайт	Запись данных, Гбайт
f_1	mProjectPP	300	3.63487	0.00198	0.00790
f_2	mDiffFit	600	1.52089	0.01615	0.00065
f_3	mConcatFit	30	286.54462	0.00188	0.00121
f_4	mBgModel	30	768.91913	0.00149	0.00013
f_5	mBackground	300	3.59002	0.00819	0.00789
f_6	mImgtbl	30	5.65319	0.00149	0.00009
f_7	mAdd	30	564.84861	1.07672	0.75729
f_8	mShrink	30	132.52509	0.40187	0.00050
f_9	mJPEG	30	1.48885	0.02471	0.00036

Таблица 3.2 – Операции и модули рабочего процесса Montage: 1 ядро узла пула 1 (продолжение)

Операция	Модуль	Среднее время чтения данных, с	Среднее время записи данных, с	Среднее время получения данных, с	Среднее время отправки данных, с
f_1	mProjectPP	0.01634	0.06871	0.01978	0.07902
f_2	mDiffFit	0.13346	0.00566	0.16148	0.00651
f_3	mConcatFit	0.01551	0.01055	0.01877	0.01213
f_4	mBgModel	0.01235	0.00112	0.01494	0.00129
f_5	mBackground	0.06773	0.06858	0.08195	0.07886
f_6	mImgtbl	0.01227	0.00082	0.01485	0.00094
f_7	mAdd	8.89850	6.58515	10.76718	7.57292
f_8	mShrink	3.32123	0.00434	4.01869	0.00499
f_9	mJPEG	0.20418	0.00315	0.24706	0.00362

Таблица 3.3 – Операции и модули рабочего процесса CyberShake: 1 ядро узла пула 1

Операция	Модуль	Число запусков модуля	Среднее время выполнения, с	Чтение данных, Гбайт	Запись данных, Гбайт
f_{10}	ExtractSGT	180	110.69955	0.17131	0.15220
f_{11}	SeisSynth	720	79.63815	0.53435	0.00001
f_{12}	ZipSeis	30	266.00113	0.11614	0.09866
f_{13}	PVCOkaya	720	0.59500	0.00002	0.00000
f_{14}	ZipPeakSA	30	195.87631	0.00107	0.00224

Таблица 3.3 – Операции и модули рабочего процесса CyberShake: 1 ядро узла пула 1 (продолжение)

Операция	Модуль	Среднее время чтения данных, с	Среднее время записи данных, с	Среднее время получения данных, с	Среднее время отправки данных, с
f_{10}	ExtractSGT	1.41582	1.32345	1.71315	1.52197
f_{11}	SeisSynth	4.41608	0.00010	5.34346	0.00011
f_{12}	ZipSeis	0.95985	0.85792	1.16141	0.98661
f_{13}	PVCOkaya	0.00015	0.00004	0.00018	0.00004
f_{14}	ZipPeakSA	0.00883	0.01951	0.01069	0.02244

Таблица 3.4 – Операции и модули рабочего процесса Epigenomics: 1 ядро узла пула 1

Операция	Модуль	Число запусков модуля	Среднее время выполнения, с	Чтение данных, Гбайт	Запись данных, Гбайт
f_{15}	mProjectPP	30	1.31432	0.00982	0.00215
f_{16}	mDiffFit	360	49.54635	0.14826	0.08197
f_{17}	mConcatFit	360	9.89969	0.02701	0.02610
f_{18}	mBgModel	360	177.93751	0.13552	0.00091
f_{19}	mBackground	360	0.46072	0.01286	0.00987
f_{20}	mImgtbl	30	2.21997	0.01297	0.01293
f_{21}	mAdd	30	38.49469	0.20909	0.10502
f_{22}	mShrink	30	30.17885	0.23659	0.23660

Таблица 3.4 – Операции и модули рабочего процесса Epigenomics: 1 ядро узла пула 1 (продолжение)

Операция	Модуль	Среднее время чтения данных, с	Среднее время записи данных, с	Среднее время получения данных, с	Среднее время отправки данных, с
f_{15}	mProjectPP	0.08114	0.01873	0.09818	0.02154
f_{16}	mDiffFit	1.22526	0.71281	1.48257	0.81973
f_{17}	mConcatFit	0.22320	0.22696	0.27008	0.26100
f_{18}	mBgModel	1.12003	0.00794	1.35524	0.00913
f_{19}	mBackground	0.10627	0.08583	0.12859	0.09870
f_{20}	mImgtbl	0.10720	0.11247	0.12971	0.12934
f_{21}	mAdd	1.72805	0.91321	2.09094	1.05019
f_{22}	mShrink	1.95527	2.05741	2.36588	2.36603

Таблица 3.5 – Операции и модули рабочего процесса LIGO: 1 ядро узла пула 1

Операция	Модуль	Число запусков модуля	Среднее время выполнения, с	Чтение данных, Гбайт	Запись данных, Гбайт
f_{23}	TmpltBank	360	84.61269	0.53972	0.00002
f_{24}	Inspiral	360	80.50396	0.54017	0.00004
f_{25}	Thinca	30	0.06964	0.00050	0.00003
f_{26}	TrigBank	360	0.09416	0.00004	0.00001
f_{27}	Inspiral	360	80.50396	0.54017	0.00004
f_{28}	Thinca	30	0.06964	0.00050	0.00003

Таблица 3.5 – Операции и модули рабочего процесса LIGO: 1 ядро узла пула 1 (продолжение)

Операция	Модуль	Среднее время чтения данных, с	Среднее время записи данных, с	Среднее время получения данных, с	Среднее время отправки данных, с
f_{23}	TmpltBank	4.46053	0.00015	5.39724	0.00018
f_{24}	Inspiral	4.46424	0.00037	5.40173	0.00043
f_{25}	Thinca	0.00415	0.00029	0.00502	0.00034
f_{26}	TrigBank	0.00032	0.00008	0.00039	0.00009
f_{27}	Inspiral	4.46424	0.00037	5.40173	0.00043
f_{28}	Thinca	0.00415	0.00029	0.00502	0.00034

Таблица 3.6 – Операции и модули рабочего процесса SIPHT: 1 ядро узла пула 1

Операция	Модуль	Число запусков модуля	Среднее время выполнения, с	Чтение данных, Гбайт	Запись данных, Гбайт
f_{29}	Patser	600	0.46066	0.00264	0.00004
f_{30}	Patser_concate	30	0.30493	0.00011	0.00014
f_{31}	Transterm	30	13.34913	0.00288	0.00000
f_{32}	Findterm	30	244.04318	0.01482	0.37010
f_{33}	RNAMotif	30	10.78713	0.00281	0.00007
f_{34}	Blast	30	1357.57752	0.78922	0.55185
f_{35}	SRNA	30	5.27608	0.04687	0.00132
f_{36}	FFN_parse	30	0.42270	0.00491	0.00243
f_{37}	Blast_synteny	30	1.51824	0.00169	0.00040
f_{38}	Blast_candidate	30	0.48704	0.00027	0.00038
f_{39}	Blast_QRNA	30	180.83542	0.78576	0.55374
f_{40}	Blast_paralogues	30	0.52507	0.00012	0.00001
f_{41}	SRNA_annotate	30	0.18925	0.00041	0.00006

Таблица 3.6 – Операции и модули рабочего процесса SIPHT: 1 ядро узла пула 1 (продолжение)

Операция	Модуль	Среднее время чтения данных, с	Среднее время записи данных, с	Среднее время получения данных, с	Среднее время отправки данных, с
f_{29}	Patser	0.02184	0.00031	0.02643	0.00035
f_{30}	Patser_concate	0.00092	0.00123	0.00112	0.00142
f_{31}	Transterm	0.02381	0.00000	0.02881	0.00000
f_{32}	Findterm	0.12248	3.21826	0.14821	3.70100
f_{33}	RNAMotif	0.02322	0.00064	0.02809	0.00073
f_{34}	Blast	6.52245	4.79871	7.89217	5.51851
f_{35}	SRNA	0.38732	0.01151	0.46866	0.01324
f_{36}	FFN_parse	0.04059	0.02114	0.04912	0.02431
f_{37}	Blast_synteny	0.01395	0.00347	0.01688	0.00399
f_{38}	Blast_candidate	0.00226	0.00329	0.00273	0.00378
f_{39}	Blast_QRNA	6.49385	4.81511	7.85756	5.53737
f_{40}	Blast_paralogues	0.00099	0.00006	0.00120	0.00007
f_{41}	SRNA_annotate	0.00335	0.00048	0.00406	0.00055

Детализированная информация по выполнению каждой операции рабочих процессов на 1 ядре узла пула 2 приведена в таблицах 3.7-3.11.

Таблица 3.7 – Операции и модули рабочего процесса Montage: 1 ядро узла пула 2

Операция	Модуль	Число запусков модуля	Среднее время выполнения, с	Чтение данных, Гбайт	Запись данных, Гбайт
f_1	mProjectPP	300	3.05	1.98	7.90
f_2	mDiffFit	600	1.28	16.15	0.65
f_3	mConcatFit	30	240.70	1.88	1.21

Таблица 3.7 – Операции и модули рабочего процесса Montage: 1 ядро узла пула 2 (продолжение)

f_4	mBgModel	30	645.89	1.49	0.13
f_5	mBackground	300	3.02	8.19	7.89
f_6	mImgtbl	30	4.75	1.49	0.09
f_7	mAdd	30	474.47	1076.72	757.29
f_8	mShrink	30	111.32	401.87	0.50
f_9	mJPEG	30	1.25	24.71	0.36
Операция	Модуль	Среднее время чтения данных, с	Среднее время записи данных, с	Среднее время получения данных, с	Среднее время отправки данных, с
f_1	mProjectPP	0.01465	0.06585	0.00198	0.00790
f_2	mDiffFit	0.11962	0.00542	0.01615	0.00065
f_3	mConcatFit	0.01391	0.01011	0.00188	0.00121
f_4	mBgModel	0.01107	0.00107	0.00149	0.00013
f_5	mBackground	0.06070	0.06572	0.00819	0.00789
f_6	mImgtbl	0.01100	0.00078	0.00149	0.00009
f_7	mAdd	7.97569	6.31077	1.07672	0.75729
f_8	mShrink	2.97681	0.00416	0.40187	0.00050
f_9	mJPEG	0.18301	0.00302	0.02471	0.00036

Таблица 3.8 – Операции и модули рабочего процесса CyberShake: 1 ядро узла пула 2

Операция	Модуль	Число запусков модуля	Среднее время выполнения, с	Чтение данных, Гбайт	Запись данных, Гбайт
f_{10}	ExtractSGT	180	92.99	171.31	152.20
f_{11}	SeisSynth	720	66.90	534.35	0.01
f_{12}	ZipSeis	30	223.44	116.14	98.66
f_{13}	PVCOkaya	720	0.50	0.02	0.00

Таблица 3.8 – Операции и модули рабочего процесса CyberShake: 1 ядро узла пула 2 (продолжение)

Операция	Модуль	Число запусков модуля	Среднее время выполнения, с	Чтение данных, Гбайт	Запись данных, Гбайт
f_{14}	ZipPeakSA	30	164.54	1.07	2.24
Операция	Модуль	Среднее время чтения данных, с	Среднее время записи данных, с	Среднее время получения данных, с	Среднее время отправки данных, с
f_{10}	ExtractSGT	1.26900	1.26831	0.17131	0.15220
f_{11}	SeisSynth	3.95812	0.00009	0.53435	0.00001
f_{12}	ZipSeis	0.86031	0.82218	0.11614	0.09866
f_{13}	PVCOkaya	0.00013	0.00004	0.00002	0.00000
f_{14}	ZipPeakSA	0.00792	0.01870	0.00107	0.00224

Таблица 3.9 – Операции и модули рабочего процесса Epigenomics: 1 ядро узла пула 2

Операция	Модуль	Число запусков модуля	Среднее время выполнения, с	Чтение данных, Гбайт	Запись данных, Гбайт
f_{15}	mProjectPP	30	1.10	9.82	2.15
f_{16}	mDiffFit	360	41.62	148.26	81.97
f_{17}	mConcatFit	360	8.32	27.01	26.10
f_{18}	mBgModel	360	149.47	135.52	0.91
f_{19}	mBackground	360	0.39	12.86	9.87
f_{20}	mImgtbl	30	1.86	12.97	12.93
f_{21}	mAdd	30	32.34	209.09	105.02
f_{22}	mShrink	30	25.35	236.59	236.60

Таблица 3.9 – Операции и модули рабочего процесса Epigenomics: 1 ядро узла пула 2 (продолжение)

Операция	Модуль	Среднее время чтения данных, с	Среднее время записи данных, с	Среднее время получения данных, с	Среднее время отправки данных, с
f_{15}	mProjectPP	0.07273	0.01795	0.00982	0.00215
f_{16}	mDiffFit	1.09820	0.68311	0.14826	0.08197
f_{17}	mConcatFit	0.20006	0.21750	0.02701	0.02610
f_{18}	mBgModel	1.00388	0.00761	0.13552	0.00091
f_{19}	mBackground	0.09525	0.08225	0.01286	0.00987
f_{20}	mImgtbl	0.09608	0.10778	0.01297	0.01293
f_{21}	mAdd	1.54885	0.87516	0.20909	0.10502
f_{22}	mShrink	1.75250	1.97169	0.23659	0.23660

Таблица 3.10 – Операции и модули рабочего процесса LIGO: 1 ядро узла пула 2

Операция	Модуль	Число запусков модуля	Среднее время выполнения, с	Чтение данных, Гбайт	Запись данных, Гбайт
f_{23}	TmpltBank	360	71.07	539.72	0.02
f_{24}	Inspiral	360	67.62	540.17	0.04
f_{25}	Thinca	30	0.06	0.50	0.03
f_{26}	TrigBank	360	0.08	0.04	0.01
f_{27}	Inspiral	360	67.62	540.17	0.04
f_{28}	Thinca	30	0.06	0.50	0.03
Операция	Модуль	Среднее время чтения данных, с	Среднее время записи данных, с	Среднее время получения данных, с	Среднее время отправки данных, с
f_{23}	TmpltBank	3.99795	0.00015	0.53972	0.00002
f_{24}	Inspiral	4.00128	0.00036	0.54017	0.00004
f_{25}	Thinca	0.00372	0.00028	0.00050	0.00003

Таблица 3.10 – Операции и модули рабочего процесса LIGO: 1 ядро узла пула 2 (продолжение)

Операция	Модуль	Среднее время чтения данных, с	Среднее время записи данных, с	Среднее время получения данных, с	Среднее время отправки данных, с
f_{26}	TrigBank	0.00029	0.00007	0.00004	0.00001
f_{27}	Inspiral	4.00128	0.00036	0.54017	0.00004
f_{28}	Thinca	0.00372	0.00028	0.00050	0.00003

Таблица 3.11 – Операции и модули рабочего процесса SIPHT: 1 ядро узла пула 2

Операция	Модуль	Число запусков модуля	Среднее время выполнения, с	Чтение данных, Гбайт	Запись данных, Гбайт
f_{29}	Patser	120	0.39	2.64	0.04
f_{30}	Patser_concate	30	0.26	0.11	0.14
f_{31}	Transterm	30	11.21	2.88	0.00
f_{32}	Findterm	30	205.00	14.82	370.10
f_{33}	RNAMotif	30	9.06	2.81	0.07
f_{34}	Blast	30	1140.37	789.22	551.85
f_{35}	SRNA	30	4.43	46.87	1.32
f_{36}	FFN_parse	30	0.36	4.91	2.43
f_{37}	Blast_synteny	30	1.28	1.69	0.40
f_{38}	Blast_candidate	30	0.41	0.27	0.38
f_{39}	Blast_QRNA	30	151.90	785.76	553.74
f_{40}	Blast_paralogues	30	0.44	0.12	0.01
f_{41}	SRNA_annotate	30	0.16	0.41	0.06

Таблица 3.11 – Операции и модули рабочего процесса SIPHT: 1 ядро узла пула 2 (продолжение)

Операция	Модуль	Среднее время чтения данных, с	Среднее время записи данных, с	Среднее время получения данных, с	Среднее время отправки данных, с
f_{29}	Patser	0.01957	0.00030	0.00264	0.00004
f_{30}	Patser_concate	0.00083	0.00118	0.00011	0.00014
f_{31}	Transterm	0.02134	0.00000	0.00288	0.00000
f_{32}	Findterm	0.10978	3.08417	0.01482	0.37010
f_{33}	RNAMotif	0.02081	0.00061	0.00281	0.00007
f_{34}	Blast	5.84605	4.59876	0.78922	0.55185
f_{35}	SRNA	0.34716	0.01103	0.04687	0.00132
f_{36}	FFN_parse	0.03638	0.02026	0.00491	0.00243
f_{37}	Blast_synteny	0.01250	0.00332	0.00169	0.00040
f_{38}	Blast_candidate	0.00203	0.00315	0.00027	0.00038
f_{39}	Blast_QRNA	5.82042	4.61448	0.78576	0.55374
f_{40}	Blast_paralogues	0.00089	0.00006	0.00012	0.00001
f_{41}	SRNA_annotate	0.00301	0.00046	0.00041	0.00006

Приложение И

Веб-интерфейс инструментальных средств разработки РППП

Спецификации параметров предметной области, модулей и схемы решения задачи РППП для решения задачи уточнения направлений развития энергетического комплекса Вьетнама приведены ниже.

```
<?xml version="1.0" encoding="UTF-8"?>
<parameters>
     <param>
          <name>input</name>
          <type>file</type>
          <filename>input.xml</filename>
     </param>
     <param>
          <name>k</name>
          <type>file</type>
          <filename>k.txt</filename>
     </param>
     <param>
          <name>p</name>
          <type>file</type>
          <filename>p.txt</filename>
     </param>
     <param>
          <name>variants</name>
          <type>filelist</type>
          <filepattern>input%1.xml</filepattern>
     </param>
     <param>
          <name>results</name>
          <type>filelist</type>
          <filepattern>result%1.xml</filepattern>
     </param>
     <param>
          <name>out</name>
          <type>file</type>
          <filename>out.xml</filename>
     </param>
</parameters>

<module>
     <info>
          <fullname>Prepare</fullname>
          <comment>Module for preapare base scenario</comment>
          <type>App</type>
          <os>Linux</os>
```

```
     </info>
     <commands>
          <start><![CDATA[
./prepare ${output:p} ${output:k} ${output:input}
]]></start>
     </commands>
     <parameters>
          <output>
               <param>
                    <name>input</name>
                    <type>file</type>
                    <filename>input.xml</filename>
               </param>
               <param>
                    <name>k</name>
                    <type>file</type>
                    <filename>k.txt</filename>
               </param>
               <param>
                    <name>p</name>
                    <type>file</type>
                    <filename>p.txt</filename>
               </param>
          </output>
     </parameters>
</module>


<module>
     <info>
          <fullname>Decompose</fullname>
          <comment>Module for decompose data</comment>
          <type>Application</type>
          <os>Linux</os>
     </info>
     <commands>
          <start stdout="s.out" stderr="s.err"><![CDATA[
               ./decompose -input_file input.xml -k ${input:k} \
               -output_dir output -p ${input:p}
          ]]></start>
     </commands>
     <parameters>
          <input>
               <param>
                    <name>input</name>
                    <type>file</type>
                    <filename>input.xml</filename>
               </param>
               <param>
                    <name>k</name>
                    <type>file</type>
                    <filename>k.txt</filename>
               </param>
```

```
                <param>
                    <name>p</name>
                    <type>file</type>
                    <filename>p.txt</filename>
                </param>
            </input>
            <output>
                <param>
                    <name>variants</name>
                    <type>filelist</type>
                    <filepattern>input%1.xml</filepattern>
                </param>
            </output>
        </parameters>
</module>

<module>
        <info>
            <fullname>Solver</fullname>
            <type>Application</type>
            <os>Windows</os>
        </info>
        <commands>
            <start><![CDATA[
                failsets.exe -input_file ${input:variant} \
                -output_file ${output:result}
            ]]></start>
        </commands>
        <parameters>
            <input>
                <param>
                    <name>variant</name>
                    <type>file</type>
                    <filename>input.xml</filename>
                </param>
            </input>
            <output>
                <param>
                    <name>result</name>
                    <type>file</type>
                    <filename>result.xml</filename>
                </param>
            </output>
        </parameters>
</module>

<module>
        <info>
            <fullname>Analyze</fullname>
            <type>Application</type>
            <os>Linux</os>
```

```
    </info>
    <commands>
        <start><![CDATA[
            ./analyze     -input_dir     ./output     -input_file
        ${intput:input} \
            -output_file ${output:out}
        ]]></start>
    </commands>
    <parameters>
        <input>
            <param>
                <name>results</name>
                <type>filelist</type>
                <filepattern>result%1.xml</filepattern>
            </param>

            <param>
                <name>input</name>
                <type>file</type>
                <filename>input.xml</filename>
            </param>
        </input>
        <output>
            <param>
                <name>out</name>
                <type>file</type>
                <filename>out.xml</filename>
            </param>
        </output>
    </parameters>
</module>


<process>
    <stages controlpoints='1'>
        <stage>
            <module name='prepare'/>
        </stage>

        <stage>
            <module name='decompose'/>
        </stage>
        <stage>
            <listmodule name='solver'/>
        </stage>
        <stage>
            <module name='analyse'/>
        </stage>
    </stages>
</process>
```

Веб-интерфейс инструментального комплекса DISCOMP. На рисунках И.1-И.9 представлены скриншоты веб-интерфейса инструментального комплекса DISCOMP, используемого в процессе разработки и применения РППП для решения задачи уточнения направлений развития энергетического комплекса Вьетнама.

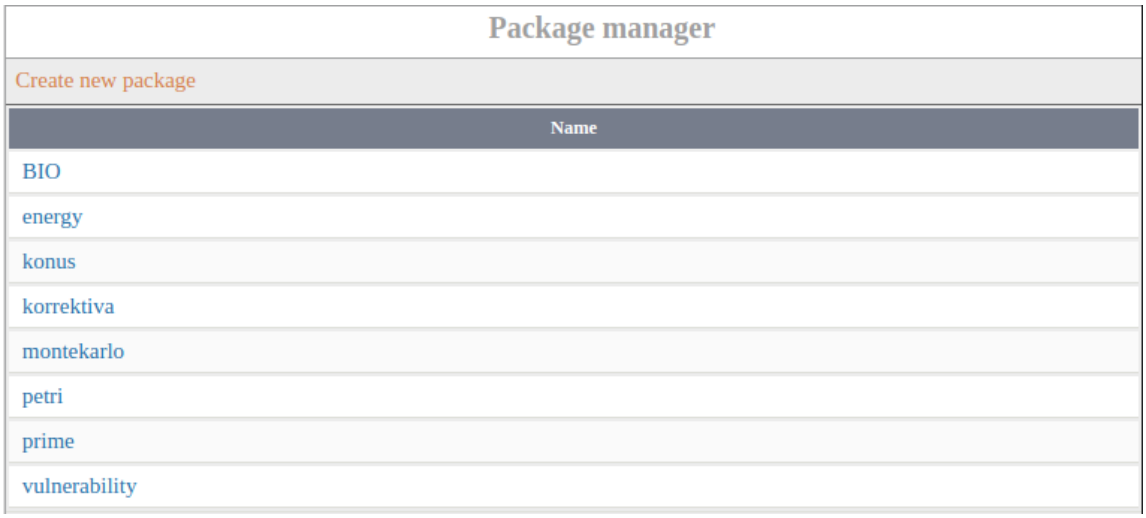


Рисунок И.1 – Меню DISCOMP для работы с РППП

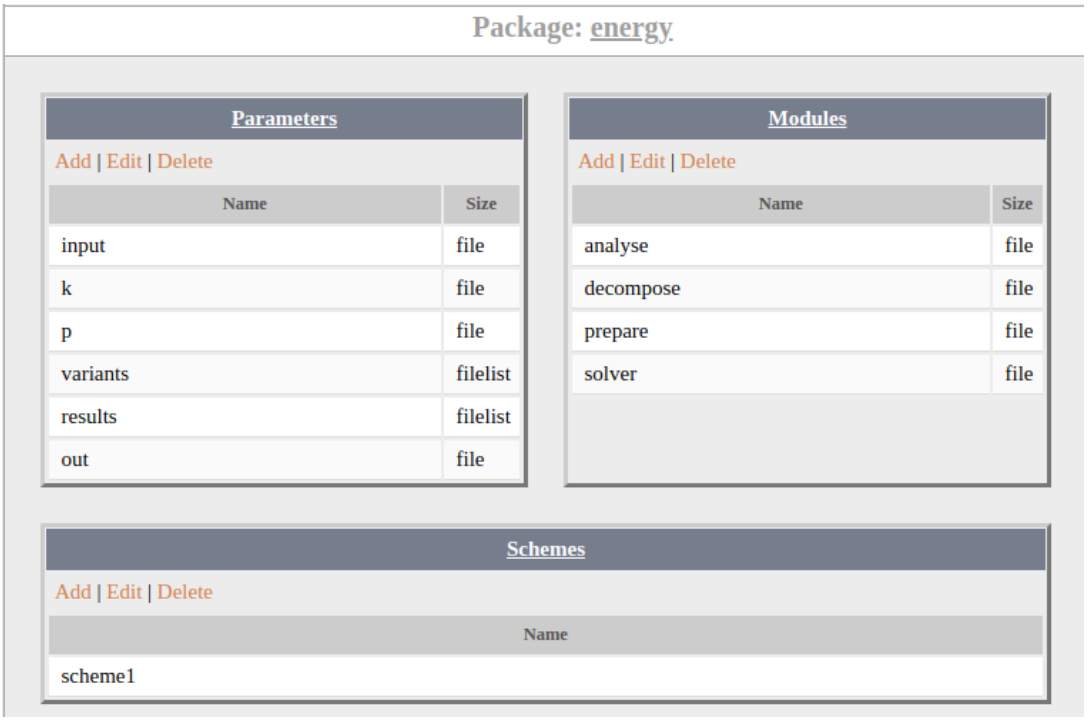


Рисунок И.2 – Списки объектов предметной области РППП для решения задачи уточнения направлений развития энергетического комплекса Вьетнама

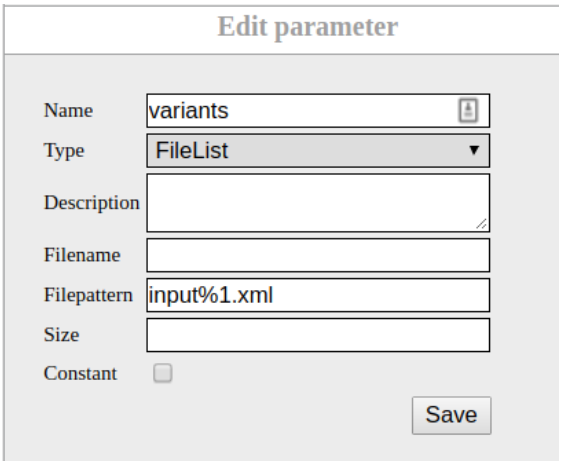


Рисунок И.3 – Спецификация параметра

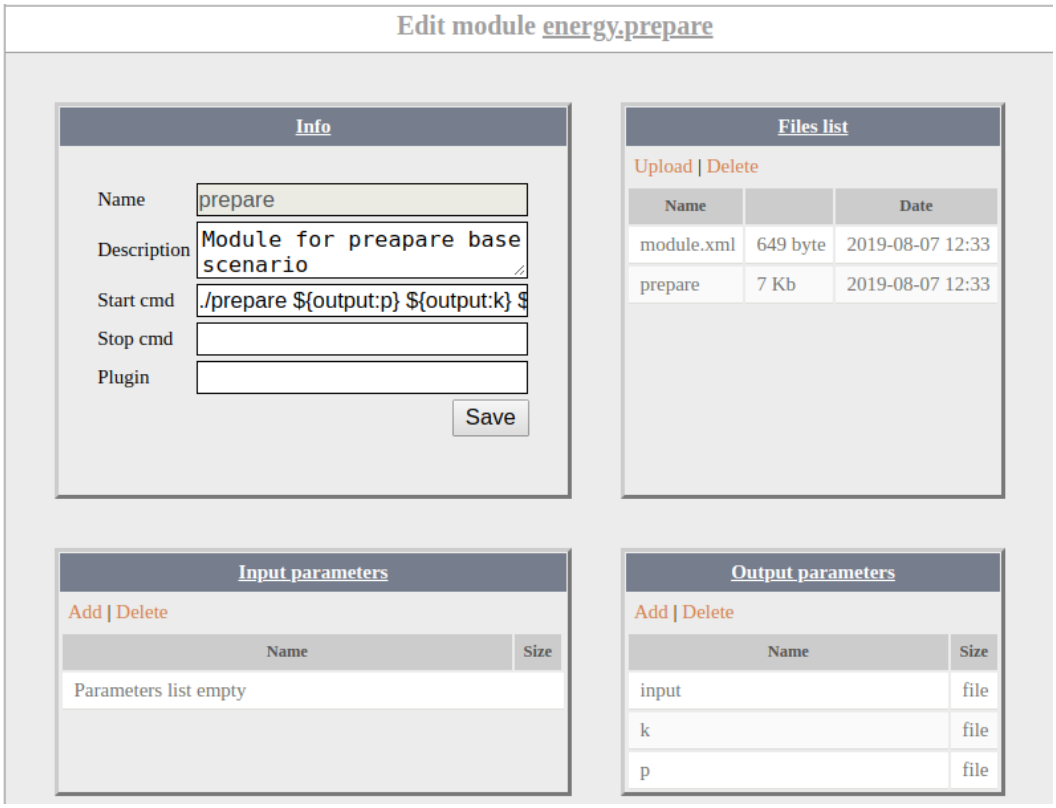


Рисунок И.4 – Спецификация модуля prepare (модуля m_1)

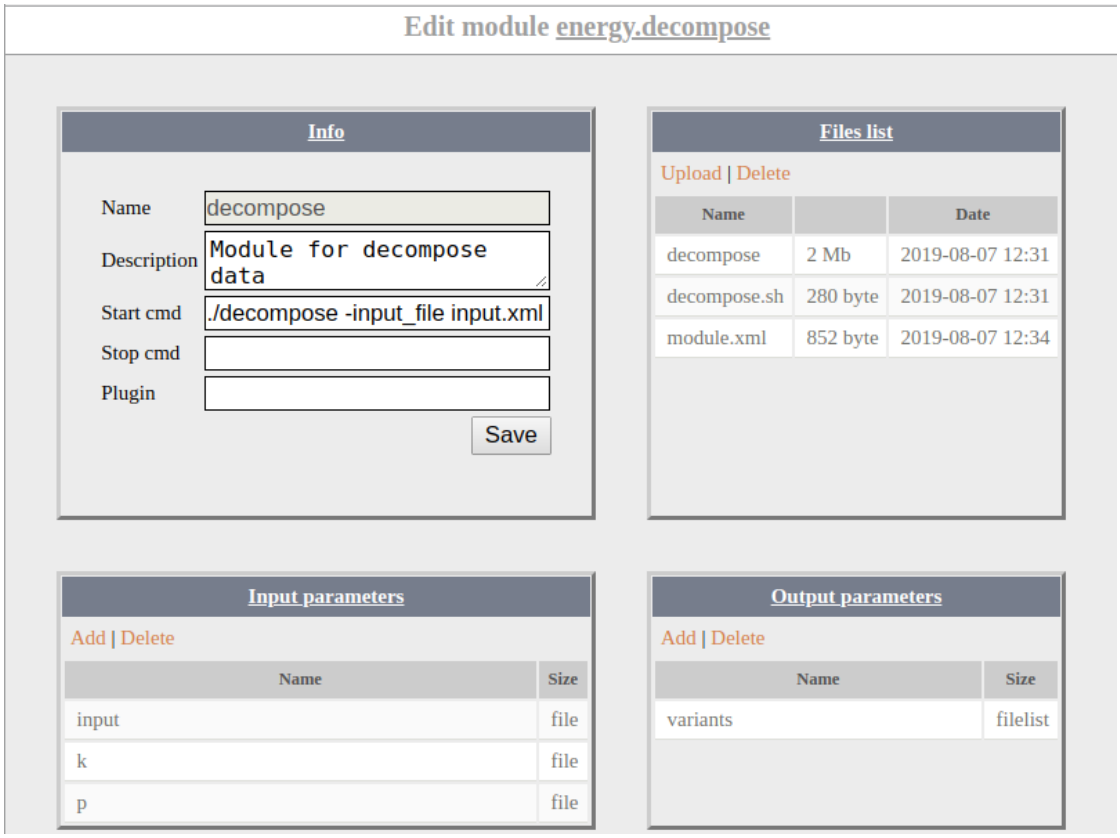


Рисунок И.5 Спецификация модуля decompose (модуля m_2)

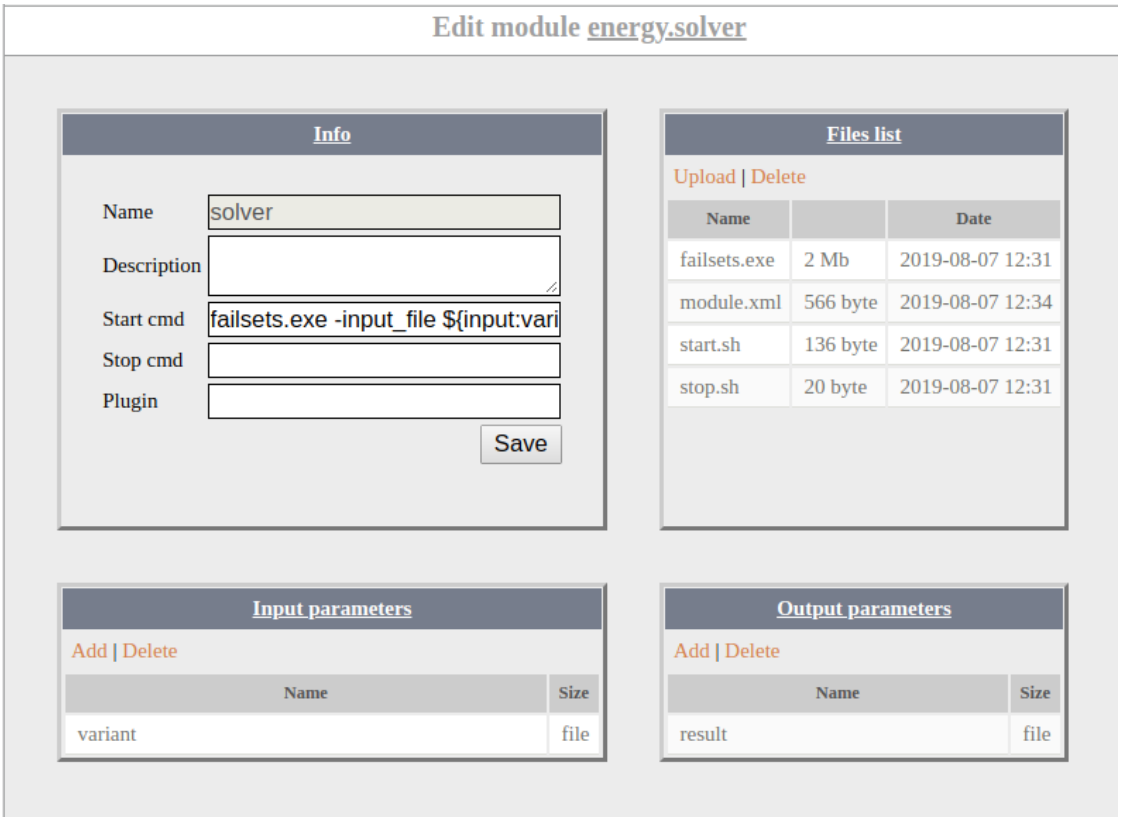


Рисунок И.6 – Спецификация модуля solver (модуля m_3)

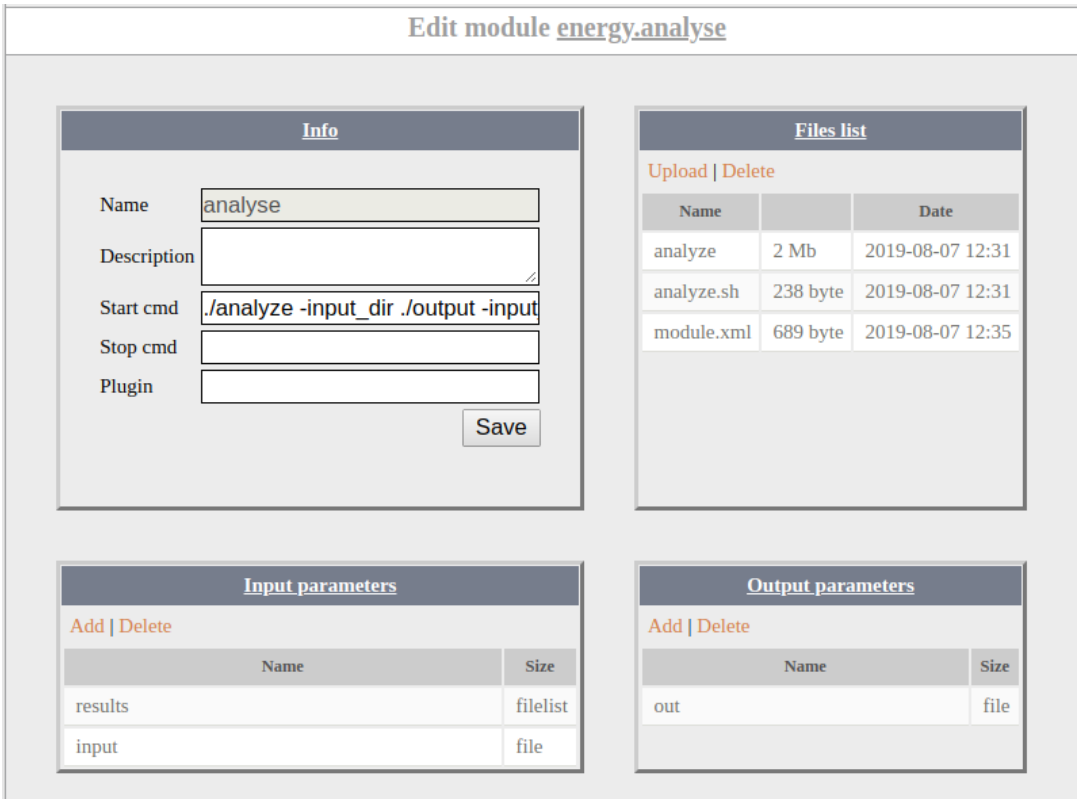


Рисунок И.7 – Спецификация модуля analyze (модуля m_4)

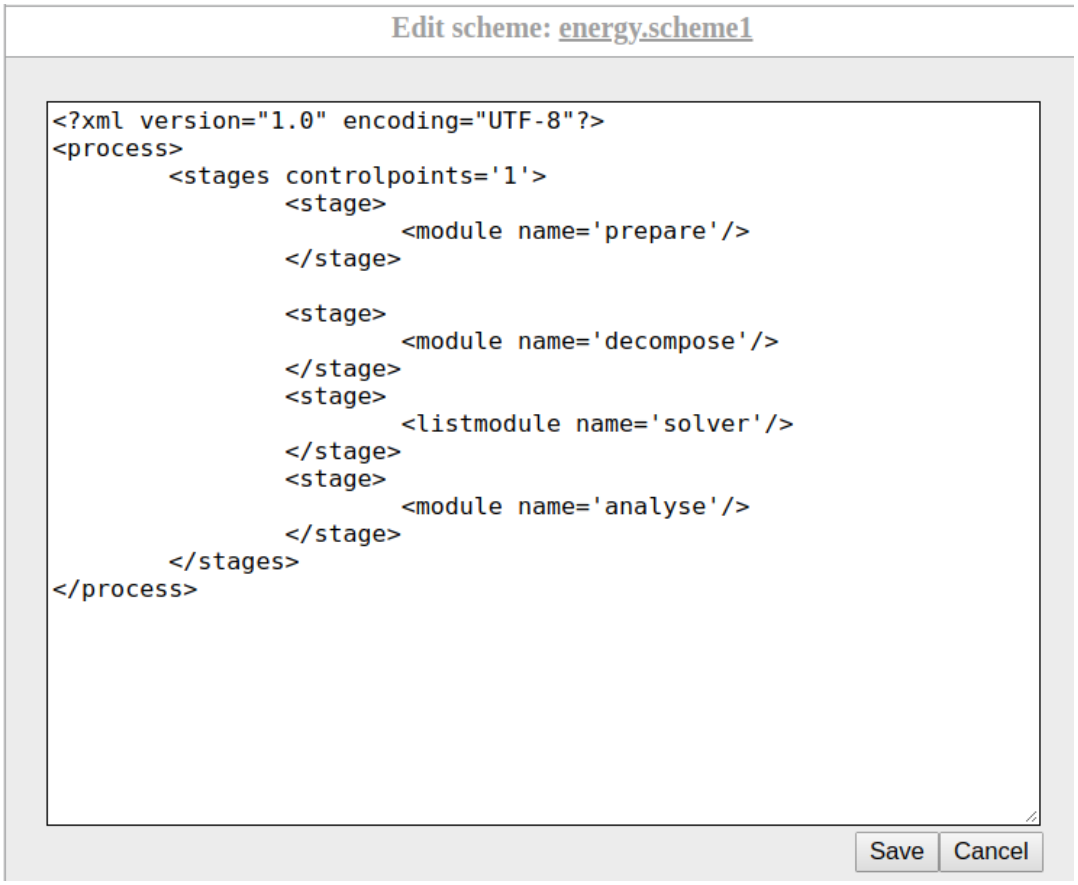
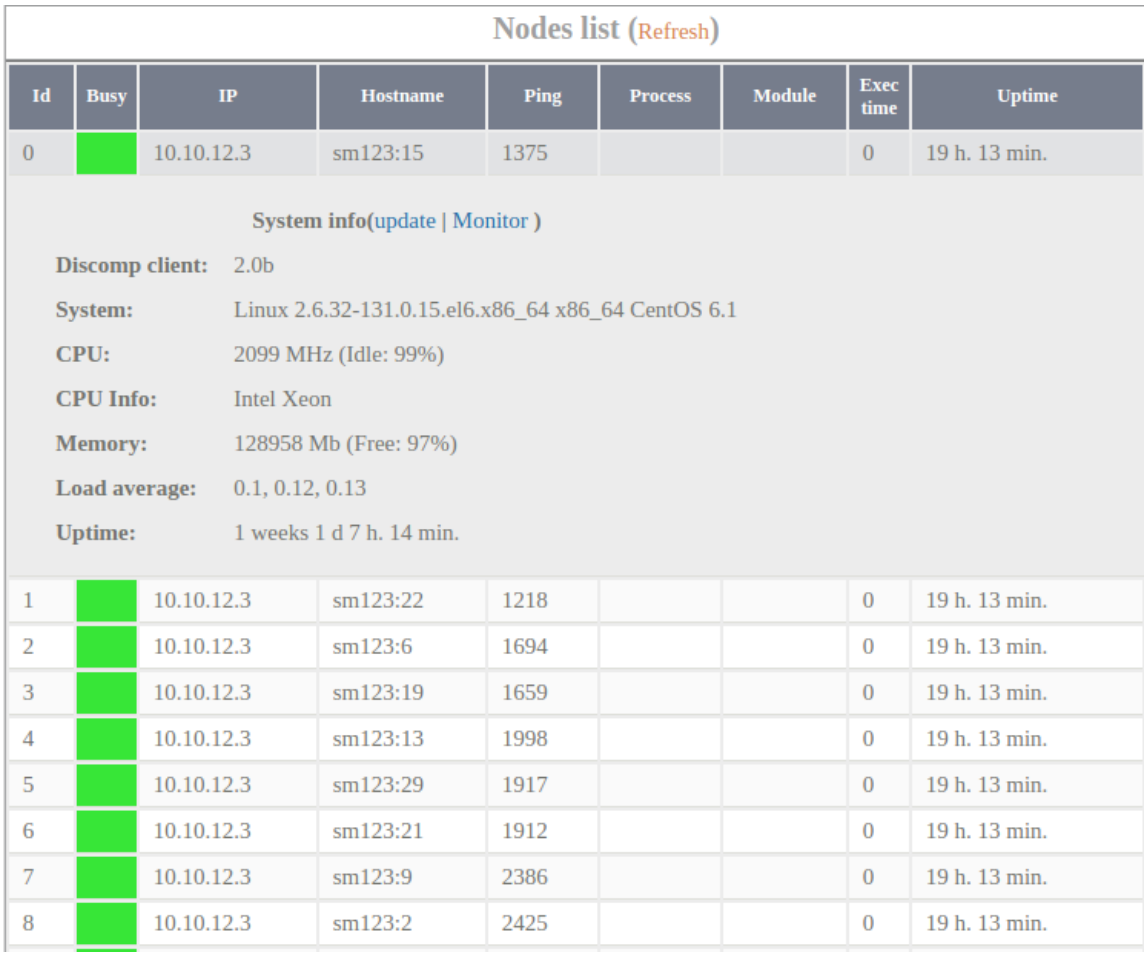


Рисунок И.8 – Спецификация схемы решения задачи



Nodes list (Refresh)

Id	Busy	IP	Hostname	Ping	Process	Module	Exec time	Uptime
0		10.10.12.3	sm123:15	1375			0	19 h. 13 min.

System info(update | Monitor)

Discomp client:	2.0b
System:	Linux 2.6.32-131.0.15.el6.x86_64 x86_64 CentOS 6.1
CPU:	2099 MHz (Idle: 99%)
CPU Info:	Intel Xeon
Memory:	128958 Mb (Free: 97%)
Load average:	0.1, 0.12, 0.13
Uptime:	1 weeks 1 d 7 h. 14 min.

1		10.10.12.3	sm123:22	1218			0	19 h. 13 min.
2		10.10.12.3	sm123:6	1694			0	19 h. 13 min.
3		10.10.12.3	sm123:19	1659			0	19 h. 13 min.
4		10.10.12.3	sm123:13	1998			0	19 h. 13 min.
5		10.10.12.3	sm123:29	1917			0	19 h. 13 min.
6		10.10.12.3	sm123:21	1912			0	19 h. 13 min.
7		10.10.12.3	sm123:9	2386			0	19 h. 13 min.
8		10.10.12.3	sm123:2	2425			0	19 h. 13 min.

Рисунок И.9 – Информация об узлах ГРВС

Спецификации параметров предметной области, операций, модулей и схемы решения задачи РППП для решения задачи выявления критических объектов газотранспортной сети России.

```
<?xml version="1.0" encoding="UTF-8"?>
   <list></list>
  </parameter>
  <parameter>
   <name>z2</name>
   <type></type>
   <extention>txt</extention>
   <list></list>
  </parameter>
  <parameter>
   <name>z3</name>
   <type></type>
   <extention>txt</extention>
   <list></list>
```

```
 </parameter>
 <parameter>
  <name>z4</name>
  <type></type>
  <extention>txt</extention>
  <list>n_0_s</list>
 </parameter>
 <parameter>
  <name>z5</name>
  <type></type>
  <extention>txt</extention>
  <list>n_0_s</list>
 </parameter>
 <parameter>
  <name>z6</name>
  <type></type>
  <extention>txt</extention>
  <list></list>
 </parameter>
 <parameter>
  <name>z7</name>
  <type></type>
  <extention>txt</extention>
  <list></list>
 </parameter>
 <parameter>
  <name>n</name>
  <type></type>
  <extention>txt</extention>
  <list></list>
 </parameter>
 ...
</project>

 <operation>
  <name>f3</name>
  <input>z1,z2,z3>z4,n</input>
  <condition>1</condition>
  <while>0</while>
  <type>module</type>
  <module>m3</module>
  <list></list>
 </operation>
 <operation>
  <name>f4</name>
  <input>z1,z2,z3,z4>z5</input>
  <condition>1</condition>
  <while>0</while>
  <type>module</type>
  <module>m4</module>
  <list>n_0_s</list>
 </operation>
```

```
<operation>
 <name>f5</name>
 <input>z1,z2,z3,z5>z6</input>
 <condition>1</condition>
 <while>0</while>
 <type>module</type>
 <module>m5</module>
 <list>n_0_s</list>
</operation>
<operation>
 <name>f6</name>
 <input>z1,z2,z3,z6>z7</input>
 <condition>1</condition>
 <while>0</while>
 <type>module</type>
 <module>m6</module>
 <list></list>
</operation>

<module>
 <name>m3</name>
 <parameter>z1,z2,z3>z4,n</parameter>
 <signatura>m3 -z1 %1 -z2 %2 -z3 %3 -z4 %4 -N %5</signatura>
 <cores>32</cores>
 <walltime>10</walltime>
 <type>node</type>
 <repo>dd dd</repo>
 <server>
  <servername>localhost</servername>
  <folder>/home/user/p1/</folder>
 </server>
</module>
<module>
 <name>m4</name>
 <parameter>z1,z2,z3,z4>z5</parameter>
 <signatura>m4 -z1 %1 -z2 %2 -z3 %3 -z4 %4 -z5 %5</signatura>
 <cores>32</cores>
 <walltime>10</walltime>
 <type>node</type>
 <repo>dd dd</repo>
 <server>
  <servername>localhost</servername>
  <folder>/home/user/p1/</folder>
 </server>
</module>
<module>
 <name>m5</name>
 <parameter>z1,z2,z3,z5>z6</parameter>
 <signatura>m5 -z1 %1 -z2 %2 -z3 %3 -z5 %4 -z6 %5 </signatura>
 <cores>32</cores>
 <walltime>10</walltime>
 <type>node</type>
```

```
   <repo>dd dd</repo>
   <server>
    <servername>localhost</servername>
    <folder>/home/user/p1/</folder>
   </server>
  </module>
  <module>
   <name>m6</name>
   <parameter>z1,z2,z3,z6[]>z7</parameter>
   <signatura>m6 -z1 %1 -z2 %2 -z3 %3 -z6 %4 -z7 %5 </signatura>
   <cores>32</cores>
   <walltime>10</walltime>
   <type>node</type>
   <repo>dd dd</repo>
   <server>
    <servername>localhost</servername>
    <folder>/home/user/p1/</folder>
   </server>
  </module>

  <task>
   <name>t1</name>
   <parameter>z1,z2,z3>z7</parameter>
   <plan_type>1</plan_type>
   <plan>f3,f4,f5,f6</plan>
  </task>
 <project>
  <name>P1</name>
  <parameter>
   <name>z1</name>
   <type></type>
   <extention>txt</extention>
```

*Веб-интерфейс инструментального комплекса Orlando To*ols. На рисунках И.10-И.15 представлены скриншоты веб-интерфейса инструментального комплекса Orlando Tools, используемого в процессе разработки и применения РППП для решения задачи выявления критических объектов газотранспортной сети России.

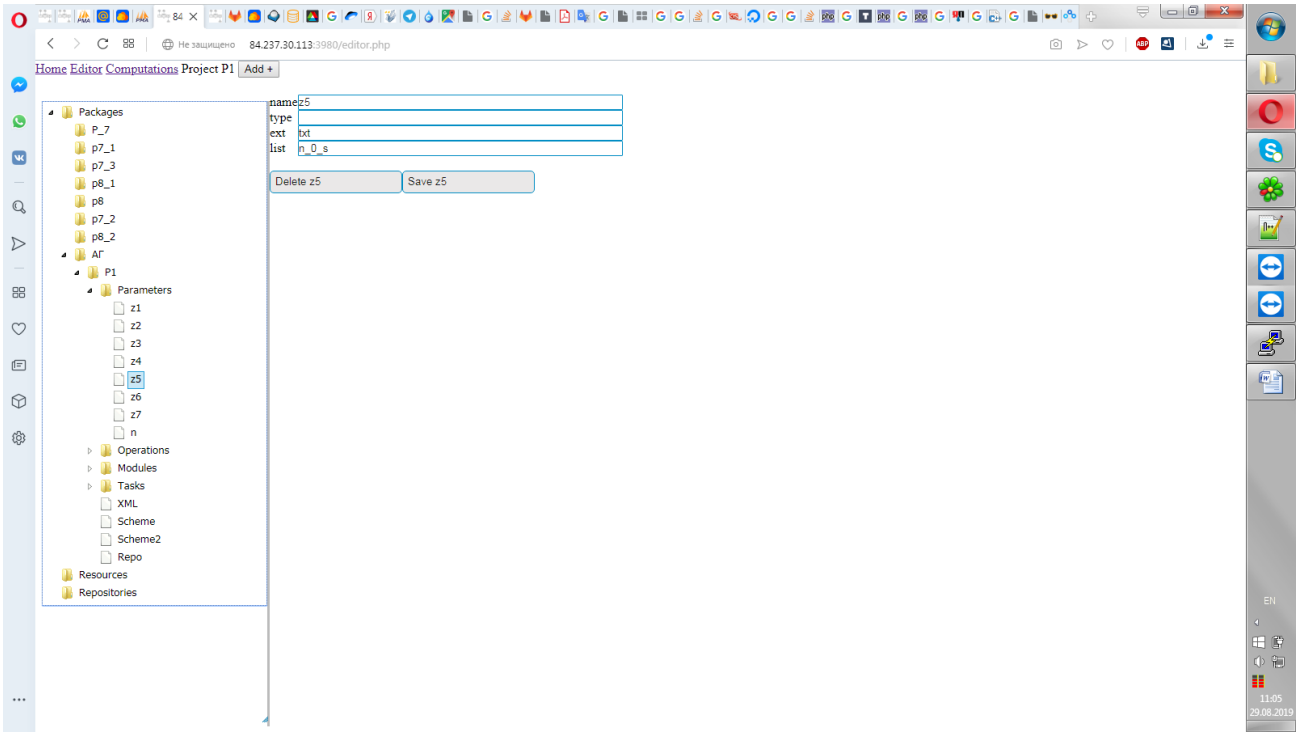


Рисунок И.10 – Форма для редактирования параметра

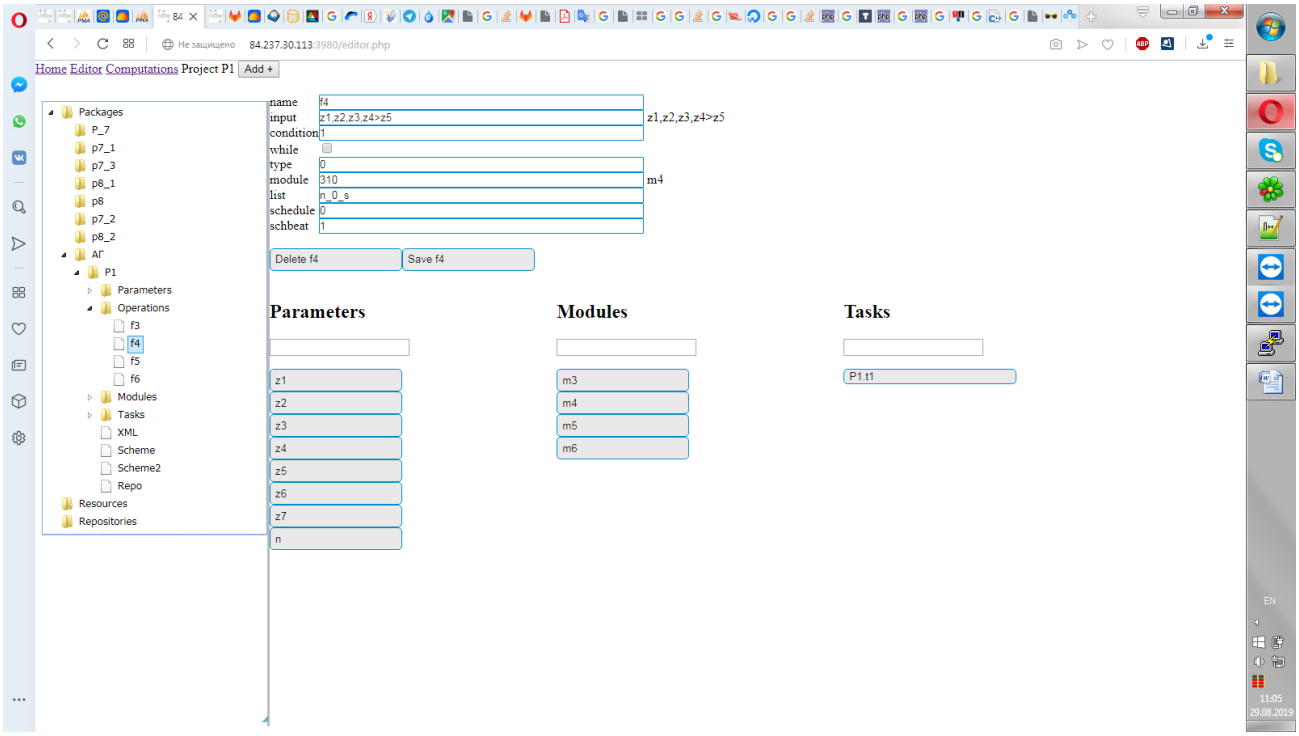


Рисунок И.11 – Форма для редактирования операции

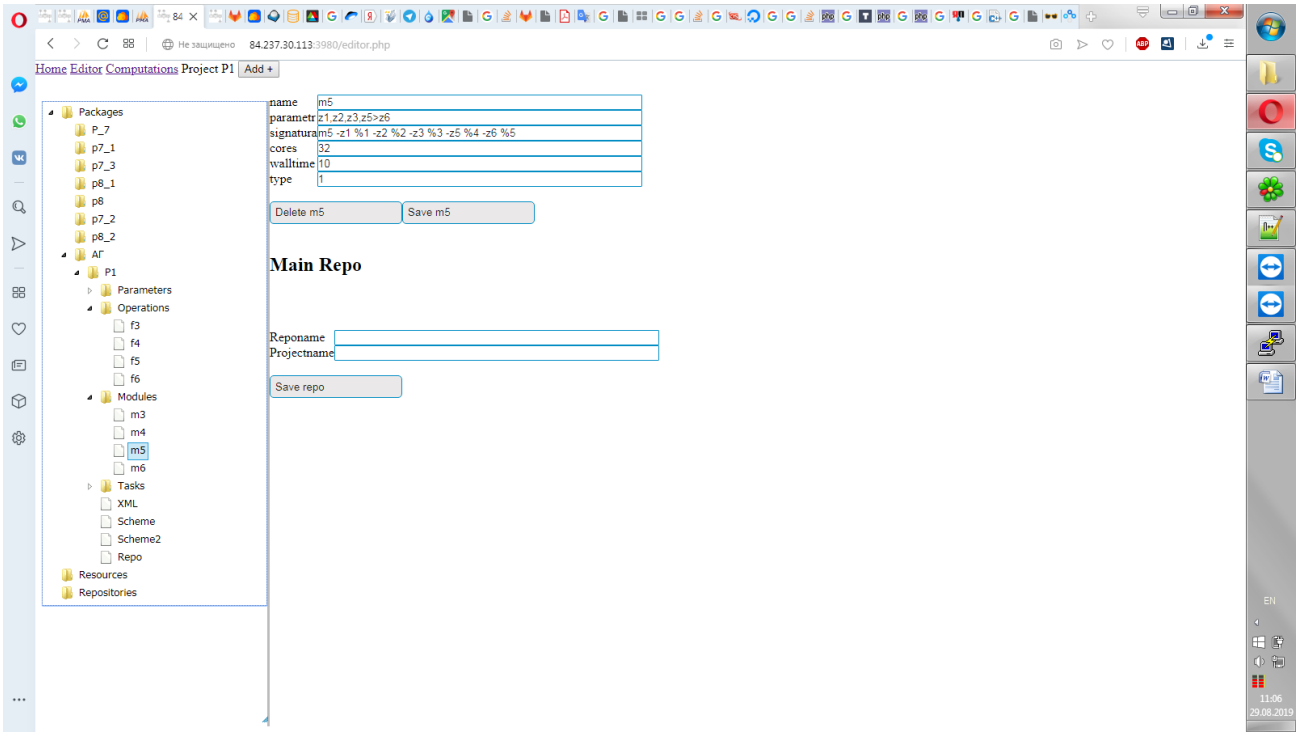


Рисунок И.12 – Форма редактирования модуля

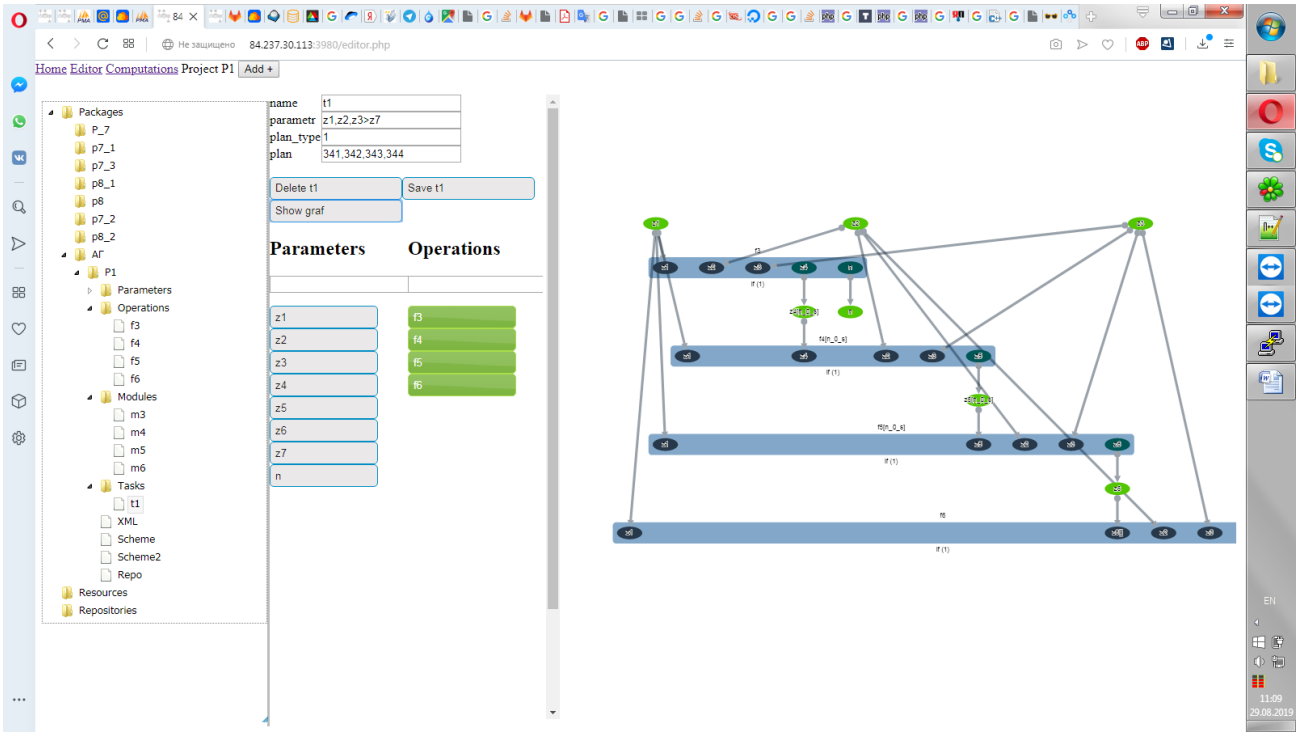


Рисунок И.13 – Форма редактирования схемы решения задачи

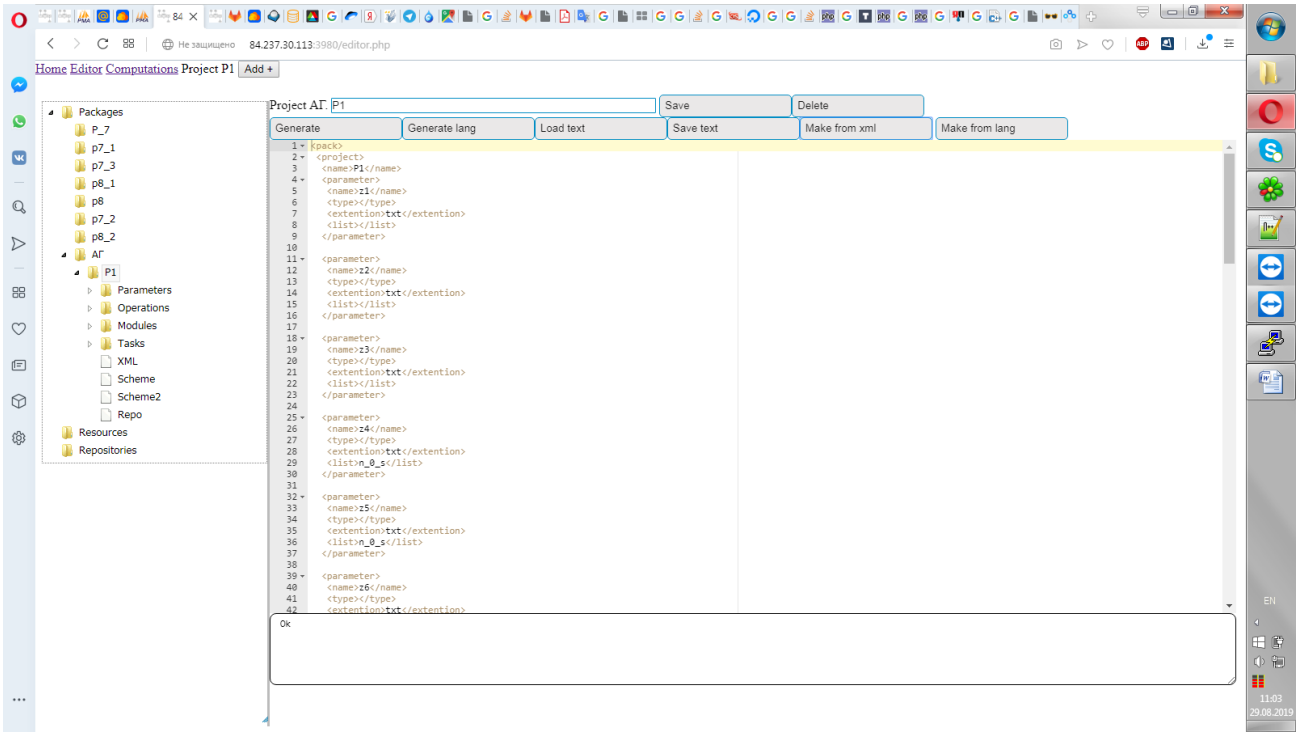


Рисунок И.14 – Форма для работы с вычислительной моделью в текстовом виде

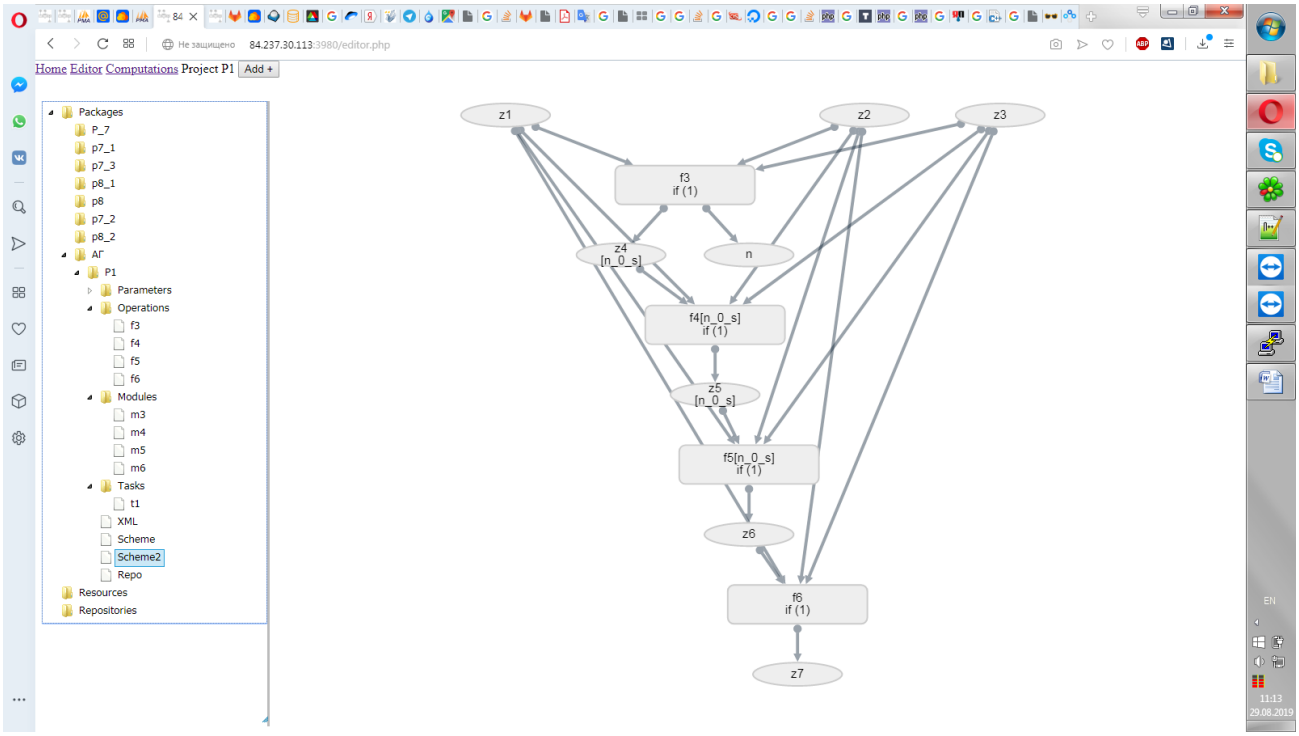


Рисунок И.15 – Форма для работы с вычислительной моделью в графическом виде

Приложение К

Объекты концептуальной модели логистического складского комплекса

Таблица К.1 – Параметры

Параметр	Содержательное имя параметра	Вид параметра	Список параметров
v_1	Наименование клиента	Простой	–
v_2	Наименования груза	Простой	–
v_3	Вес груза	Простой	–
v_4	Объем груза	Простой	–
v_5	Товарно-транспортная накладная	Составной	v_1, v_2, v_3, v_4
v_6	Вид тары	Простой	–
v_7	Вид отгрузки/разгрузки (внавал или в паллетах)	Простой	–
v_8	Температура хранения	Простой	–
v_9	Номер камеры	Простой	–
v_{10}	Число грузчиков	Простой	–
v_{11}	Число электропогрузчиков	Простой	–
v_{12}	Возможность штабелирования	Простой	–
v_{13}	Число паллет	Простой	–
v_{14}	Вес паллеты	Простой	–
v_{15}	Требуется дозаморозка	Простой	–
v_{16}	Наличие транспортной тары	Простой	–
v_{17}	Акт приемки/отгрузки	Составной	v_7, v_9, v_{12}, v_{32}
v_{18}	Ветеринарный осмотр пройден	Простой	–
v_{19}	Партионные признаки	Простой	–
v_{20}	Количество брака	Простой	–
v_{21}	Параметры бракованного груза	Простой	–
v_{22}	Заказ поставлен на хранение	Простой	–
v_{23}	Вес груза в камере	Простой	–

Таблица К.1 – Параметры (продолжение)

Параметр	Содержательное имя параметра	Вид параметра	Список параметров
v_{24}	Площадь груза в камере	Простой	–
v_{25}	Приходная накладная	Простой	–
v_{26}	Остатки груза по клиенту	Простой	–
v_{27}	Количество отходов тары на утилизацию	Простой	–
v_{28}	Количество отходов тары на реализацию	Простой	–
v_{29}	Виды отходов тары	Простой	–
v_{30}	Отходы транспортной тары	Составной	v_{27}, v_{28}
v_{31}	Заказ собран	Простой	–
v_{32}	Расходная накладная	Простой	–
v_{33}	Общие остатки груза по клиенту	Простой	–
v_{34}	Число дней хранения	Простой	–
v_{35}	Стоимость хранения груза по клиенту	Простой	–
v_{36}	Акт инвентаризации	Простой	–
v_{37}	Акт о технологической оттайке камеры	Простой	–
v_{38}	Есть заявка на разгрузку	Простой	–
v_{39}	Есть заявка на отгрузку	Простой	–
v_{40}	Есть заявка на инвентаризацию	Простой	–
v_{41}	Есть заявка на технологическую оттайку камеры	Простой	–
v_{42}	Есть заявка на расчет оплаты за хранение груза	Простой	–
v_{43}	Разгрузка завершена	Простой	–
v_{44}	Груз оформлен	Простой	–
v_{45}	Товар выгружен	Простой	–

Таблица К.1 – Параметры (продолжение)

Параметр	Содержательное имя параметра	Вид параметра	Список параметров
v_{46}	Требуется ветеринарный контроль	Простой	–
v_{47}	Ветеринарный контроль пройден	Простой	–
v_{48}	Груз упакован	Простой	–
v_{49}	Дозаморозка выполнена	Простой	–

Таблица К.2 – Операции

Операция	Содержательное имя операции	Входные параметры операции	Выходные параметры операции
w_3	Приход заявки на разгрузку	–	v_5, v_6, v_7, v_8, v_9, v_{16}, v_{19}
w_4	Определение характеристик разгрузки	v_5, v_6, v_7	v_{10}, v_{11}, v_{12}, v_{15}, v_{46}
w_5	Разгрузка груза внавал	v_5, v_6, v_{10}, v_{11}	v_{17}, v_{45}
w_6	Разгрузка груза в паллетах	v_5, v_{11}	v_{13}, v_{14}, v_{17}, v_{45}
w_7	Ветеринарный осмотр груза	v_5	v_{18}, v_{47}
w_8	Отбраковка и сотрировка груза	v_5, v_{19}	v_{20}, v_{21}
w_9	Упаковка груза	v_5, v_{20}, v_{21}	v_{48}
w_{10}	Дозаморозка груза	v_5	v_{49}
w_{11}	Паллетирование груза	v_5, v_{21}	v_{13}, v_{14}
w_{12}	Установка груза на хранение	v_5, v_8, v_9, v_{12}, v_{13}, v_{14}	v_{22}
w_{13}	Оформление приемки груза	v_5, v_{17}	v_{23}, v_{24}, v_{25}, v_{26}, v_{44}
w_{14}	Сбор отходов товарной тары	v_5	v_{30}

Таблица К.2 – Операции (продолжение)

Операция	Содержательное имя операции	Входные параметры операции	Выходные параметры операции
w_{15}	Приход заявки на отгрузку	–	v_5, v_6, v_7, v_9, v_{19}
w_{16}	Определение характеристик отгрузки	v_5, v_6, v_7	v_{10}, v_{11}
w_{17}	Сбор заказа	v_5, v_9, v_{10}, v_{11}, v_{19}	v_{31}
w_{18}	Отгрузка груза внавал	v_5, v_6, v_{10}, v_{11}	v_{17}
w_{19}	Отгрузка груза в паллетах	v_5, v_{11}	v_{17}
w_{20}	Оформление отгрузки груза	v_5, v_{17}	v_{23}, v_{24}, v_{26}, v_{32}
w_{21}	Технологическая оттайка камеры	v_9, v_{13}	v_{37}
w_{22}	Хранение груза	v_1, v_{33}, v_{34}	v_{35}
w_{23}	Инвентаризация груза	v_1, v_{33}	v_{36}

Таблица К.3 – Продукции

Продукция	Логическое выражение в левой части продукции, представленное булевой формулой	Операция в правой части продукции
pr_1	v_{38}	w_3
pr_2	v_{38}	w_4
pr_3	$\bar{v}_7 v_{38}$	w_5
pr_4	$v_7 v_{38}$	w_6
pr_5	$v_{38} v_{45} v_{46}$	w_7
pr_6	$\bar{v}_7 v_{18} v_{38} v_{45} (\bar{v}_{46} \vee v_{47})$	w_8
pr_7	$\bar{v}_7 v_{18} v_{38} v_{45}$	w_9
pr_8	$v_{15} v_{18} v_{38} v_{45} v_{48}$	w_{10}
pr_9	$\bar{v}_7 (\bar{v}_{15} \vee v_{49}) v_{18} v_{38} v_{45}$	w_{11}

Таблица К.3 – Продукции (продолжение)

Продукция	Логическое выражение в левой части продукции, представленное булевой формулой	Операция в правой части продукции
pr_{10}	$(\bar{v}_{15} \vee v_{49})v_{18}v_{38}v_{45}$	w_{12}
pr_{11}	$v_{18}v_{22}v_{38}v_{45}$	w_{13}
pr_{12}	$v_{16}v_{18}v_{38}v_{44}v_{45}$	w_{14}
pr_{13}	v_{39}	w_{15}
pr_{14}	v_{39}	w_{16}
pr_{15}	v_{39}	w_{17}
pr_{16}	$\bar{v}_{7}v_{31}v$	w_{18}
pr_{17}	$v_{7}v_{31}v_{39}$	w_{19}
pr_{18}	v_{39}	w_{20}
pr_{19}	v_{41}	w_{21}
pr_{20}	v_{42}	w_{22}
pr_{21}	v_{40}	w_{23}

Приложение Л

Варианты плановых и случайных заявок клиентов на обслуживание

Таблица Л.1 – Варианты плановых заявок

№ варианта	V, т	Операция	t_1	t_2, ч	n_1	n_2	n_2
1	55	погрузка	03:30	4	2	4	2
	3.8	отгрузка	10:00	1	2	2	1
	19	погрузка	13:00	2	2	4	4
	7.5	погрузка	17:30	2	1	2	2
	4	отгрузка	21:00	1	2	2	1
2	18	отгрузка	08:00	2	2	4	2
	39	отгрузка	13:00	4	2	4	2
	3.5	погрузка	18:30	1	2	2	1
	60	отгрузка	22:00	4	2	4	4
3	19.5	погрузка	07:00	2	2	4	2
	57	отгрузка	9:30	4	2	4	2
	4	погрузка	14:00	1	2	2	1
	17	погрузка	18:00	2	2	4	2
	20	отгрузка	22:00	2	2	4	2
	16	погрузка	23:00	2	2	4	2
4	18	погрузка	05:30	2	2	4	2
	60	погрузка	08:00	4	2	4	2
	12	отгрузка	13:00	2	2	2	1
	3	отгрузка	19:00	1	2	2	1
5	60	погрузка	03:30	4	2	4	2
	20	погрузка	07:00	2	2	4	2
	4	погрузка	18:30	1	2	2	1
…	…						

Таблица Л.2 – Варианты случайных заявок

№ варианта	Интенсивность поступления заявок	Вероятность заявки на погрузку/отгрузку груза весом 1, 1.5, 5, 10, 20, 60 т	n_1	n_2	n_2
1	30 ± 15	0.10, 0.15, 0.15, 0.15, 0.20, 0.25	6	12	12
2	30 ± 15	0.10, 0.15, 0.15, 0.15, 0.20, 0.25	4	10	10
3	30 ± 15	0.10, 0.15, 0.15, 0.15, 0.20, 0.25	3	6	6
4	30 ± 15	0.10, 0.15, 0.15, 0.15, 0.20, 0.25	6	4	4
5	30 ± 15	0.21, 0.24, 0.19, 0.16, 0.11, 0.09	4	12	12
6	30 ± 15	0.21, 0.24, 0.19, 0.16, 0.11, 0.09	3	10	10
7	30 ± 15	0.21, 0.24, 0.19, 0.16, 0.11, 0.09	6	6	6
8	30 ± 15	0.21, 0.24, 0.19, 0.16, 0.11, 0.09	4	4	4
9	40 ± 10	0.14, 0.16, 0.14, 0.16, 0.15, 0.25	3	12	12
10	40 ± 10	0.14, 0.16, 0.14, 0.16, 0.15, 0.25	6	10	10
11	40 ± 10	0.14, 0.16, 0.14, 0.16, 0.15, 0.25	4	6	6
12	40 ± 10	0.14, 0.16, 0.14, 0.16, 0.15, 0.25	3	4	4
13	40 ± 10	0.20, 0.25, 0.23, 0.10, 0.10, 0.12	6	12	12
14	40 ± 10	0.20, 0.25, 0.23, 0.10, 0.10, 0.12	4	10	10
15	40 ± 10	0.20, 0.25, 0.23, 0.10, 0.10, 0.12	3	6	6
16	40 ± 10	0.20, 0.25, 0.23, 0.10, 0.10, 0.12	6	4	4
…	…				

В таблицах Л.1 и Л.2 использованы следующие обозначения:

- V – вес груза в тоннах;
- t_1 – время поступления заявки;
- t_2 – планируемое время обслуживания заявки;
- n_1 – число кладовщиков;
- n_2 – число грузчиков;
- n_3 – число электропогрузчиков.