ISSN 2075-6836

УЧРЕЖДЕНИЕ РОССИЙСКОЙ АКАДЕМИИ НАУК
ИНСТИТУТ
КОСМИЧЕСКИХ
ИССЛЕДОВАНИЙ
РАН

ВЫЧИСЛИТЕЛЬНЫЕ ТЕХНОЛОГИИ В ЕСТЕСТВЕННЫХ НАУКАХ

ПЕРСПЕКТИВНЫЕ КОМПЬЮТЕРНЫЕ СИСТЕМЫ: УСТРОЙСТВА, МЕТОДЫ И КОНЦЕПЦИИ

ТРУДЫ СЕМИНАРА

ТАРУСА, 2–4 МАРТА 2011 г.

ПОД РЕДАКЦИЕЙ
Р. Р. НАЗИРОВА, Л. Н. ЩУРА

СЕРИЯ
МЕХАНИКА, УПРАВЛЕНИЕ И ИНФОРМАТИКА

МОСКВА
2011

УДК 004
В94

ISSN 2075-6836

Computer Technologies in Natural Science
Advanced computing: systems, methods and conceptions.
Materials of the Workshop. Tarusa, 2–4 March, 2011

Eds. R. R. Nazirov, L. N. Shchur

This issue is the collection of the papers based on the talks presented at the extended workshop "Advanced computing: systems, methods and conceptions", which was held from 2–4 March 2011, in Tarusa hotel Interkosmos of the RAS Space Research Institute. It is the sixth workshop in the series of extended workshops, devoted to the computer technologies in natural sciences. The first workshop was devoted to computer simulations of the actual problems in physics and mechanics and papers was published in the issue 1 (Proceedings of the seminar on Computing Technology in Science. Iss. 1. Computational Physics / Ed. R. R. Nazirov. M.: KDU, 2009. 288 p.), the second workshop was devoted to the discussion of the physical and medical aspects of the research of the complex nonlinear processes in the human body (Methods of Non-linear Analysis in Cardiology and Oncology: Physicist's Approaches and Clinical Practice. Iss. 2. / Ed. R. R. Nazirov. M.: KDU, 2010. 206 p.), the third workshop was devoted to the computer simulations in biology and chemistry (Materials of the seminar on Computing Technology in Science. Iss. 3. Computational Chemistry and Biology / Eds. R. R. Nazirov, L. N. Shchur. M.: KDU, 2010. 118 p.), the fourth was devoted to the discussion of the computer vision in control systems of the mobile objects (Computer vision in control systems of the mobile objects: Proceedings of the conference-workshop-2010. Iss. 4 / Ed. R. R. Nazirov. M.: KDU, 2011. 328 p.). The fifth workshop was devoted to the simulation of events of global scale (Proceedings of the seminar on Computing Technology in Science. Global scale systems / Eds. R. R. Nazirov, L. N. Shchur. M.: IKI, 2011, 148 p.). Important highlight of the workshop is the active participation of the young scientists, and more than half of the talks were done by the young speakers.

Keywords: computing systems, quantum computing, GRID, data bases, space systems, public domain software, multiagent technologies, cloud computing, program engineering, distributed computing.

Вычислительные технологии в естественных науках.
Перспективные компьютерные системы: устройства, методы и концепции.
Труды семинара. Таруса, 2–4 марта 2011 г.

Под ред. Р. Р. Назирова, Л. Н. Щура

Настоящий сборник основан на докладах, представленных на расширенном семинаре «Перспективные компьютерные системы: устройства, методы и концепции», который проходил 2–4 марта 2011 г. в Тарусе на базе гостиницы «Интеркосмос» Учреждения Российской академии наук Института космических исследований РАН. Это очередной семинар из серии расширенных семинаров, посвященных вычислительным технологиям в естественных науках. На первом семинаре рассматривались вопросы компьютерного моделирования актуальных задач физики и механики, его труды изданы в 2009 г. в выпуске 1 (Труды семинара по вычислительным технологиям в естественных науках. Вып. 1. Вычислительная физика / Под ред. Р. Р. Назирова. М.: КДУ, 2009. 288 с.); второй был посвящен обсуждению физических и медицинских аспектов исследования сложных нелинейных физических процессов в организме человека (Методы нелинейного анализа в кардиологии и онкологии: Физические подходы и клиническая практика. Вып. 2 / Под ред. Р. Р. Назирова. М.: КДУ, 2010. 206 с.); на третьем обсуждались проблемы компьютерного моделирования задач биологии и химии (Труды семинара по вычислительным технологиям в естественных науках. Вып. 3. Вычислительная химия и биология / Под ред. Р. Р. Назирова, Л. Н. Щура. М.: КДУ, 2010. 118 с.); на четвертом — проблемы технического зрения (Техническое зрение в системах управления мобильными объектами-2010: Труды научно-технической конференции-семинара. Вып. 4. / Под ред. Р. Р. Назирова. М.: КДУ, 2011. 328 с.). На пятом семинаре обсуждались вопросы моделирования событий глобальных масштабов (Вычислительные технологии в естественных науках. Системы глобального масштаба : Труды семинара / Под ред. Р. Р. Назирова, Л. Н. Щура. М.: ИКИ РАН, 2011. 148 с.). Важная особенность проводимых расширенных семинаров состоит в привлечении молодежи к активному участию в качестве докладчиков. Как правило, более половины докладов делаются молодыми исследователями.

Ключевые слова: вычислительные технологии, квантовые вычисления, грид, базы данных, космические системы, свободнораспространяемое программное обеспечение, мультиагентные технологии, облачные вычисления, программная инженерия, распределенные вычисления.

Редакторы: *Рожкова М. А.*, *Корниленко В. С.* Компьютерная верстка: *Комарова Н. Ю.*

Отдельные материалы даны в авторской редакции.

ОТ ОРГАНИЗАТОРОВ

Читателю предлагаются избранные статьи, написанные на основе приглашенных докладов на очередном расширенном семинаре по вычислительным методам в естествознании «Перспективные компьютерные системы: устройства, методы и концепции». Семинар был проведен 2–4 марта 2011 г. на базе гостиницы «Интеркосмос» Учреждения Российской академии наук Института космических исследований РАН. Он посвящался перспективам развития компьютерных систем и по существу рассматриваемых вопросов стал принципиально междисциплинарным. Были приглашены активно работающие исследователи из различных областей естественнонаучных знаний, представляющие широкий спектр географических и организационных особенностей нашей отечественной науки, представлена как вузовская, так и академическая и ведомственная наука. Акцент в предложенных докладах и в ходе активных обсуждений был сделан на все аспекты развития компьютерных систем — перспективные вычислительные устройства, методики использования вычислительных систем, концептуальные подходы к использованию вычислительных устройств и систем.

Во многих областях знаний слово «вычислительная» характеризует третий столп естествознания, дополнительный двум традиционным столпам исследования природных явлений — теории и эксперименту. Его отличительной чертой является использование вычислительных (компьютерных) технологий как основного инструмента исследований. Важность развития этого направления трудно переоценить. Происходит появление нового знания на стыке различных разделов наук при грамотном применении новейших технологий и инструментов исследования природы.

Первый семинар был посвящен компьютерному моделированию актуальных задач физики и механики, его труды изданы в сборнике [1], второй — обсуждению физических и медицинских аспектов исследования сложных нелинейных физических процессов в организме человека [2], третий — компьютерному моделированию задач биологии и химии [3], на четвертом обсуждались проблемы технического зрения [4], пятый семинар был посвящен моделированию в системах глобального масштаба [5].

Важная особенность проводимых расширенных семинаров состоит в привлечении молодежи к активному участию в качестве докладчиков. Как правило, более половины докладов делаются молодыми исследователями.

Выбор тематики семинара имеет свою интересную историю...

Представленные на семинаре доклады были разбиты на двенадцать секций. В секции «Квантовые вычисления» было заявлено три выступления. Сатанин Аркадий Михайлович (Научно-исследовательский физико-технический институт (НИФТИ ННГУ), Нижний Новгород) в докладе «Сверхпроводящие кубиты для информационных технологий» [6] познакомил слушателей с современными экспериментами по конструированию кубитов и исследованию таких свойств, как запутывание состояний и время декогерентности. Особое внимание было уделено кубитам на основе джозефсоновских контактов. Его молодой коллега Гельман Александр Иосифович (Институт прикладной физики Российской Академии наук (ИПФ РАН), Нижний Новгород) представил доклад «Квантовые траектории в диссипативной динамике единичного кубита» [7] о результатах численного моделирования временного поведения кубитов, основанного на квантовом методе Монте-Карло. Проведено качественное сравнение с результатами экспериментов, полученными группой Рахманова. Наконец, завершил работу секции доклад Аблаева Фарида Мансуровича «Квантовые ветвящиеся программы — новая парадигма модели квантовых компьютеров» [8], в котором говорилось о ветвящемся алгоритме, использование которого позволяет провести анализ сложности некоторых операций над кубитами.

Следующая секция была посвящена развитию грид-технологий в Российской Федерации. Крюков Александр Павлович (Научно-исследовательский институт ядерной физики им. Д. В. Скобельцына МГУ имени М. В. Ломоносова (НИИЯФ МГУ), Москва) в докладе «Технологические основы грид-инфраструктуры для Национальной нанотехнологической сети» [9] познакомил слушателей с работами по созданию инфраструктуры грид-сети для проводимых в РФ исследований по нанотехнологиям и с некоторыми особенностями реализации доступа к ресурсам сети. Второй доклад в этой секции — «Архитектурно-технологические аспекты эволюции Grid» [10] — был сделан молодым исследователем из Московского государственного университета Шундеевым Александром Сергеевичем, который провел экскурс в историю развития технологических преференций при построении грид.

На вечерней сессии этой секции был представлен один фундаментальный доклад «Современные параллельные СУБД и управление транзакциями» [11], в котором Кузнецов Сергей Дмитриевич (ИСП РАН, Москва) изложил свой взгляд на современное состояние в области теории параллельных СУБД и практики их применения. Более чем двухчасовая лекция была с интересом встречена слушателями и прерывалась многочисленными вопросами и обсуждениями.

Второй день семинара начался с доклада «Концептуальные подходы к созданию перспективных космических систем» [12] Алексея Александровича Романова (ОАО «Российские космические системы», Москва). В нем акцентировался переход на новую парадигму в развитии орбитальных аппаратов, которая основана на идее малых спутников и базируется на двух функциональных факторах — дифференциации функций малых спутников и интеграции их для выполнения конкретной задачи. Доклад Виктора Петрович Иванникова (ИСП РАН, Москва) «Что такое СПО» [13] был посвящен концепции использования свободно распространяемого программного обеспечения для решения нацио-

нальных задач и насыщен интересными фактами из личного профессионального опыта.

Мультиагентные технологии были основой двух докладов из Самары. Петр Олегович Скобелев познакомил слушателей с развиваемым его коллективом подходом по управлению ресурсами в реальном времени, основанным на идее конкуренции программных агентов, обслуживающих процессы и аппараты [14]. Его молодой коллега Антон Владимирович Иващенко подробно изложил применение этой концепции к управлению производственными процессами [15].

Александр Константинович Петренко познакомил слушателей с проводимыми в ИСП РАН исследованиями по программной инженерии [16].

В секции «Естественнонаучные приложения» Андрей Рыбкин из Всероссийского ядерного центра (Саров) представил доклад «Программный комплекс на базе гибридных вычислительных систем для расчета критических параметров методом Монте-Карло» [17]. Федор Сергеевич Зайцев с факультета вычислительной математики и кибернетики (ВМК) МГУ в докладе «Структура и функциональные возможности комплекса имитационного моделирования «Виртуальный токамак» рассказал о программном комплексе, созданном большим коллективом ученых из Московского государственного университета, НИИ системных исследований РАН, Института проблем управления РАН, Троицкого института инновационных и термоядерных исследований и Института ядерного синтеза, РНЦ «Курчатовский Институт» [18]. Молодой сотрудник МГУ Дмитрий Анатольевич Шульга поделился своим взглядом на перспективы развития компьютерных систем и приложением своих идей к поиску новых лекарственных препаратов [19].

В секции «Облачные вычисления» было запланировано два доклада, но по техническим причинам состоялся только одни. Вениамин Викторович Коноплев из ИКИ РАН (Москва) в докладе «Текущие и перспективные технологии виртуализации как платформа для организации облачных вычислений: сравнение и анализ» ознакомил слушателей с техническими аспектами технологии виртуализации облачных вычислений и привел сравнительный анализ производительности наиболее употребительных технических средств на разных этапах процесса облачной обработки [20].

В секции прикладная математика было заслушано три доклада из Санкт-Петербургского государственного университета. В своем докладе «Рандомизация процесса получения данных как средство „обогащения“ наблюдений и ускорения обработки» Граничин Олег Николаевич остановился на теоретических аспектах ускорения обработки данных наблюдений [21]. Его молодой коллега Олег П. Исаев в докладе «Аппаратная реализация на базе FPGA парадигмы Compressive Sensing для захвата изображений» подробно изложил аспекты практической реализации указанных идей, в частности в беспилотных летательных аппаратах [22]. Молодая сотрудница ИКИ РАН Татьяна Викторовна Полякова в докладе «Разработка точных и реалистичных цифровых МКЭ-моделей в задачах дентальной имплантации» рассказала о разрабатываемом ею подходе по моделированию некоторых практических задач челюстной хирургии с помощью методов конечных элементов [23].

Доклад Наталии Юрьевны Комаровой «О подготовке рукописей статей к печати» был сделан с целью улучшения качества предоставляемого авторами мате-

риала для публикации и ознакомления докладчиков с правилами оформления текста (включая формулы, таблицы, библиографию и т. п.) и иллюстраций [24].

Применениям новых информационных технологий в медицине были посвящены два выступления из Нижегородского государственного университета. Виктор Павлович Гергель в докладе «Создание специализированных высокопроизводительных компьютеров на базе новейших графических процессоров для применения в технологиях медицинской визуализации и телемедицины» изложил подход к решению сложных задач визуализации в медицине, развиваемый на первом в России факультете вычислительной математики [25]. Его молодой коллега Григорий Осипов рассказал о проблемах моделирования сердечной активности в докладе «Моделирование сердечной активности с использованием суперкомпьютерных технологий» [26].

Последней по счету, но важнейшей по востребованности, была секция «Распределенные технологии», на которую были вынесены два доклада молодых сотрудников. Алексей Анатольевич Пойда (ИКИ РАН) в докладе «Вычисление в беспроводных сетях» изложил подход своей группы к решению проблемы проведения вычислений в распределенных мобильных сетях с автоматической конфигурацией топологии [27]. Более «приземленное» решение предлагается в подходе сотрудников НИВЦ МГУ, изложенном Сергеем Игоревичем Соболевым в докладе «Технология распределенных вычислений X-Com: возможности, задачи, направления развития» [28], основу его составляет развиваемое в НИВЦ оригинальное программное обеспечение X-Com.

Мультимедийные версии докладов доступны на странице семинара [29].

По окончанию семинара состоялась дискуссия, результатом которой было принятие обращения участников к представителям структур управления наукой о необходимости организации конкурса по финансированию научных исследований в области неэволюционного развития суперкомпьютерных вычислений [30].

Семинар прошел в обстановке научных обсуждений, коллегиальности и большого интереса участников. Как отметил один из авторитетнейших участников семинара, «Обстановка семинара была как в старое доброе время на кухне».

[1] Труды семинара по вычислительным технологиям в естественных науках. Вып. 1. Вычислительная физика / Под ред. Р. Р. Назирова. М.: КДУ, 2009. 288 с.

[2] Методы нелинейного анализа в кардиологии и онкологии: Физические подходы и клиническая практика. Вып. 2 / Под ред. Р. Р. Назирова. М. КДУ, 2010. 206 с.

[3] Труды семинара по вычислительным технологиям в естественных науках. Вып. 3. Вычислительная химия и биология / Под ред. Р. Р. Назирова, Л. Н. Щура. М.: КДУ, 2010. 118 с.

[4] Техническое зрение в системах управления мобильными объектами-2010: Тр. научно-технич. конф.-семинара. Вып. 4. / Под ред. Р. Р. Назирова. М.: КДУ, 2011. 328 с.

[5] Вычислительные технологии в естественных науках. Системы глобального масштаба : Труды семинара / Под ред. Р. Р. Назирова, Л. Н. Щура. М.: ИКИ РАН, 2011. 148 с.

[6] *А. М. Сатанин*. Сверхпроводящие кубиты для информационных технологий.

[7] *А. И. Гельман*. Квантовые траектории в диссипативной динамике единичного кубита.

[8] *Ф. М. Аблаев*. Квантовые ветвящиеся программы — новая парадигма модели квантовых компьютеров.

[9] *А. П. Крюков*. Технологические основы грид-инфраструктуры для Национальной нанотехнологической сети.

[10] *А. С. Шундеев*. Архитектурно-технологические аспекты эволюции Grid.

[11] *С. Д. Кузнецов*. Современные параллельные СУБД и управление транзакциями.

[12] *А. А. Романов*. Концептуальные подходы к созданию перспективных космических систем.

[13] *В. П. Иванников*. Что такое СПО.

[14] *П. О. Скобелев*. Мультиагентные технологии для управления ресурсами в реальном времени.

[15] *А. В. Иващенко*. Мультиагентная система для управления производством в реальном времени.

[16] *А. В. Петренко*. Реинжиниринг программной инженерии.

[17] *А. Рыбкин*. Программный комплекс на базе гибридных вычислительных систем для расчета критических параметров методом Монте-Карло.

[18] *Ф. С. Зайцев*. Структура и функциональные возможности комплекса имитационного моделирования «Виртуальный токамак».

[19] *Д. А. Шульга*. Эволюция компьютерных систем: новые возможности и вызовы.

[20] *В. В. Коноплев*. Текущие и перспективные технологии виртуализации как платформа для организации облачных вычислений: сравнение и анализ.

[21] *О. Н. Граничин*. Рандомизация процесса получения данных как средство «обогащения» наблюдений и ускорения обработки.

[22] *О. П. Исаев*. Аппаратная реализация на базе FPGA парадигмы Compressive Sensing для захвата изображений.

[23] *Т. В. Полякова*. Разработка точных и реалистичных цифровых МКЭ-моделей в задачах дентальной имплантации.

[24] *Н. Ю. Комарова*. О подготовке рукописей статей к печати.

[25] *В. П. Гергель*. Создание специализированных высокопроизводительных компьютеров на базе новейших графических процессоров для применения в технологиях медицинской визуализации и телемедицины.

[26] *Г. Осипов*. Моделирование сердечной активности с использованием суперкомпьютерных технологий.

[27] *А. А. Пойда*. Вычисление в беспроводных сетях.

[28] *С. И. Соболев*. Технология распределенных вычислений X-Com: возможности, задачи, направления развития.

[29] Мультимедийные версии докладов. Режим доступа: http://www.iki.rssi.ru/seminar/2011030204/.

[30] Резолюция участников расширенного семинара «Перспективные компьютерные системы: устройства, методы и концепции», «Интеркосмос», Таруса, 2–4 марта 2011 года.

Обсудив в ходе семинара актуальные вопросы развития перспективных компьютерных систем, участники семинара сошлись в следующем мнении. Для устойчивого развития различных областей деятельности необходима поддержка на конкурсной основе научных исследований и разработок в области перспективных компьютерных систем. Следует обратить особое внимание на поддержку проведения исследований и разработок по неэволюционному развитию компьютерных систем.

Организаторы семинара

Назиров Р. Р. — доктор технических наук, профессор ИКИ РАН

Щур Л. Н. — доктор физико-математических наук, профессор ИТФ им. Л. Д. Ландау РАН

УДК 537.86

УПРАВЛЕНИЕ НАСЕЛЕННОСТЯМИ КУБИТОВ ПУТЕМ ВОЗДЕЙСТВИЯ ЭЛЕКТРОМАГНИТНЫХ ИМПУЛЬСОВ

А. В. Швецов [1], А. М. Сатанин [1], А. И. Гельман [2]

[1] *Нижегородский государственный университет им. Н. И. Лобачевского (ННГУ им. Н. И. Лобачевского)*
[2] *Учреждение Российской академии наук Институт прикладной физики (ИПФ РАН), Москва*

Изучается возможность создания периодической модуляции населенности в волноводной линии, содержащей последовательность идентичных джозефсоновских переходов, каждый из которых при низких уровнях возбуждения представляет собой двухуровневую систему — кубит. Показано, что дисперсионные характеристики среды зависят от состояний кубитов, которые можно инициализировать путем воздействия электромагнитных импульсов, распространяющихся в волноводной линии. Аналитически и численно продемонстрировано, что встречные импульсы позволяют осуществить накачку системы и создать периодическую модуляцию населенностей кубитов с периодом, равным половине длины волны импульсов. Получаемые квантовые структуры проявляют свойства фотонных кристаллов. Детально рассмотрены условия эффективного возбуждения кубитов в такой системе.

Работа выполнена при поддержке РФФИ (проект № 11-02-97058-р_поволжье_а).

Ключевые слова: метаматериалы, джозефсоновские переходы, кубит.

ВВЕДЕНИЕ

В последнее время огромный интерес вызывают исследования искусственных сред, обладающих необычными электродинамическими свойствами [Agranovich et al., 2004; Zharov et al., 2003; Shvets, 2003]. Наиболее известный пример подобных сред — это вещества с отрицательным показателем преломления или среды Веселаго [Veselago, 2003]. Современные нанотехнологии позволяют создавать композиционные материалы, содержащие включения разных масштабов и свойств, которые обладают заданными электродинамическими характеристиками (диэлектрическими и магнитными проницаемостями), коренным образом отличающимися от характеристик компонент [Zhang et al., 2006]. Подобные искусственные среды или метаматериалы уже широко используются для создания безотражательных покрытий и различных оптических элементов (адаптивных линз, перестраиваемых зеркал, конвертеров и т. д.) [Ziolkowski et al., 2006; Korobkin et al., 2006]. Другое направление исследований связано с разработкой метаматериалов на основе активных сред, например, содержащих сверхпроводящие джозефсоновские переходы или джозефсоновские генераторы [Barone, Paterno, 1982; Likharev, 1986; Ricci et al., 2005; Du et al., 2006]. Если к джозефсоновскому

Швецов Александр Владимирович — научный сотрудник, кандидат физико-математических наук, e-mail: alexshdze@mail.ru.

Сатанин Аркадий Михайлович — доктор физико-математических наук, профессор, e-mail: sarkady@mail.ru.

Гельман Александр Иосифович — младший научный сотрудник, кандидат физико-математических наук, e-mail: gelman@appl.sci-nnov.ru

переходу приложить напряжение, то он может генерировать электромагнитное излучение в области терагерцевого диапазона. Как известно, излучение в этом диапазоне имеет очень большое значение для практических применений, поэтому джозефсоновские среды могут быть использованы для создания генераторов терагерцевого излучения, фильтров, детекторов и волноводов [Barone, Paterno,, 1982; Savel'ev et al., 2005; Yampol'skii et al., 2007]. Поскольку джозефсоновские переходы обладают нелинейной индуктивностью, в настоящее время они уже используются для создания нелинейных метаматериалов [Lazarides, Tsironis, 2007].

В данной работе изучаются принципиально новые среды — квантовые метаматериалы [Rakhmanov et al., 2008]. Если изготовить джозефсоновские переходы с малыми емкостями, то при низких температурах у них эффективно будет населено (возбуждено) несколько энергетических уровней и динамические свойства переходов будут описываться уравнениями квантовой механики [Rakhmanov et al., 2008; Nakamura et al., 2002]. В свою очередь, физические свойства таких материалов также будут зависеть от квантового состояния, в котором находятся сверхпроводящие элементы [Lazarides, Tsironis, 2007; Rakhmanov et al., 2008]. Главное отличие таких метаматериалов от классических состоит в том, что в них можно создавать суперпозицию состояний, которые будут сохранять когерентность на временах, меньших времени релаксации квантовых элементов среды (составляющих для отдельных джозефсоновских переходов микросекунды). Предполагая время декогерентности существенно большим времени распространения сигналов, мы можем описывать квантовые метаматериалы волновой функцией. Чтобы записать или считать информацию, можно использовать когерентное классическое поле, которое позволит осуществить управление средой по аналогии с оптикой, используя pump-probe методику. Иными словами, естественно использовать традиционный подход, развитый в оптике при описании взаимодействия излучения с атомами или молекулами, когда электромагнитное поле описывается квазиклассически, а джозефсоновская подсистема — квантовыми уравнениями. В частности, можно попытаться применить методы Раби или Рамси, подавая на среду одиночные (Раби) импульсы или их последовательность (Рамси) для активации и диагностики среды. Однако применение подобных методов для квантовых метаматериалов не развито.

В настоящее время отсутствует теория, описывающая управление функцией отклика метаматериалов путем изменения квантовых состояний образующих элементов. Можно отметить работу [Rakhmanov et al., 2008], в которой обсуждалась возможность создания статических и медленно вращающихся структур в волноводной линии, заполненной джозефсоновскими переходами. В работе [Rakhmanov et al., 2008] также утверждается, что на основе упомянутой волноводной линии путем периодического изменения квантового состояния сверхпроводящих элементов может быть создан квантовый фотонный кристалл, однако метод динамической модуляции не разработан. Нерешенными также остаются проблемы создания периодической модуляции населенности в этой волноводной линии и анализа записанной информации.

В настоящей работе изучена возможность управления состояниями среды путем воздействия на нее электромагнитных импульсов. Рассмотрен активный участок волноводной линии, заполненный джозефсоновскими переходами.

Предполагается, что он идеально сопряжен с пассивными участками, которые служат для соединения активной среды с генератором сигналов и соответствующими приемными устройствами. Далее будет показано, что при помощи сильных электромагнитных импульсов, распространяющихся навстречу друг другу, можно формировать периодическую модуляцию населенности в активной области и последняя проявляет свойства квантовых фотонных кристаллов. Периодом и величиной модуляции можно эффективно управлять, изменяя частоту и амплитуду импульсов.

МОДЕЛЬ И ОСНОВНЫЕ УРАВНЕНИЯ

Волноводная линия, изучаемая в данной работе, изображена на рис. 1 (см. с. 12). Активная часть линии образована джозефсоновскими переходами — сверхпроводящими элементами (S), — которые соединены со сверхпроводящими линиями оксидными перемычками. Эта часть линии вполне аналогична системе, рассмотренной в работе [Rakhmanov et al., 2008]. Для связи с внешними устройствами служат пассивные участки линии, заполненные нормальными элементами (N), параметры которых подобраны так, чтобы обеспечить идеальное сопряжение. Области $n < 0$ и $n > N$ содержат мостики из нормального металла, где n определяет номер мостика. Пусть направление распространения импульсов совпадает с осью Z. Обозначим векторный потенциал между n-м и $n+1$-м мостиками как A_{xn} (в силу геометрии задачи векторный потенциал имеет лишь проекцию на ось X). Динамика участка волноводной линии со сверхпроводящими мостиками может быть представлена в лагранжевой форме:

$$L_{SC} = \sum_{n=1}^{N} \left[\begin{array}{c} \dfrac{E_J}{2\omega_J^2}\left(\left(\dfrac{2\pi D\dot{A}_{xn}}{\Phi_0} + \dot{\varphi}_n \right)^2 + \left(\dfrac{2\pi D\dot{A}_{xn}}{\Phi_0} - \dot{\varphi}_n \right)^2 \right) + \\ + E_J \left(\cos\left(\dfrac{2\pi D\dot{A}_{xn}}{\Phi_0} + \dot{\varphi}_n \right) + \cos\left(\dfrac{2\pi D\dot{A}_{xn}}{\Phi_0} - \dot{\varphi}_n \right) \right) + \\ + \dfrac{DL_0 W\dot{A}_{xn}^2}{8\pi c^2} - \dfrac{DL_0 W}{8\pi}\left(\dfrac{A_{x,n+1} - A_{xn}}{L_0} \right)^2 + 2I_n\dot{\varphi}_n \end{array} \right], \qquad (1)$$

где E_j, ω_J^2 — энергия и частота Джозефсона соответственно, $E_J = \Phi_0 I_c / 2\pi c$, $\omega_J^2 = 2eI_c/\hbar C$; I_c — критический ток; C — емкость джозефсоновских переходов; D — расстояние между сверхпроводящими линиями; Φ_0 — квант магнитного потока; φ_n — фаза сверхпроводящего мостика с номером n; W — толщина джозефсоновского перехода; c — скорость света; I_n — ток, текущий через мостик с номером n. Первое слагаемое в (1) связано с емкостью джозефсоновских переходов; второе определяет свободную энергию барьеров; третье и четвертое описывают энергию электромагнитного поля. Здесь мы полагаем, что ширина мостиков мала по сравнению с расстоянием между ними, а векторный потенциал не зависит от координаты в области n-го перехода.

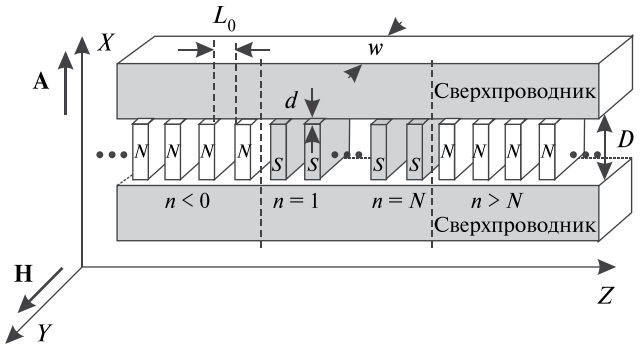


Рис. 1. Волноводная линия, содержащая джозефсоновские переходы: D — расстояние между сверхпроводящими линиями; L_0 — расстояние между соседними сверхпроводящими мостиками. Мостики с номерами $n < 0$ и $n > N$ сделаны из нормального металла; мостики с номерами $0 < n < N$ — сверхпроводящие. Каждый сверхпроводящий мостик формирует туннельные контакты со сверхпроводящими линиями. Толщина контактов — d. Электромагнитный импульс распространяется вдоль оси Z; **A** — векторный потенциал; **H** — магнитное поле электромагнитной волны

Лагранжиан пассивной области волноводной линии (содержащей мостики из нормального металла) равен

$$L_{NC} = \sum_{n<0,n>N} \left[\frac{CD^2 \dot{A}_{xn}^2}{c^2} + \frac{DL_0 W \dot{A}_{xn}^2}{8\pi c^2} - \frac{DL_0 W}{8\pi} \left(\frac{A_{x,n+1} - A_{xn}}{L_0} \right)^2 \right]. \quad (2)$$

Из сравнения выражений для энергий электромагнитного поля в области сверхпроводящих мостиков (1) и мостиков из нормального металла (2) следует, что согласование импедансов обеспечено.

В дальнейшем нас будут интересовать только низколежащие возбуждения джозефсоновских переходов. Рассматривая в (1) лишь члены, содержащие фазу φ_n, можно записать гамильтониан двойного джозефсоновского перехода, взаимодействующего с полем, в виде:

$$\hat{H}_n = -\frac{(\hbar\omega_J)^2}{E_J} \left(\frac{\partial}{\partial\varphi_n} \right)^2 - 2(E_J \cos(\varphi_n) - I_n \varphi_n) + \hat{V}_n, \quad (3)$$

$$\hat{V}_n = 2E_J (1 - \cos(a_n)) \cos(\varphi_n), \quad (4)$$

где $\hat{V}_n$ — оператор взаимодействия кубита с электромагнитной волной; a_n — потенциал электромагнитного поля (безразмерный векторный потенциал), $a_n = 2\pi D A_{xn} / \Phi_0$, который будем измерять в a_0 (такое значение поля, при котором $A_{xn} = \Phi_0 / 2\pi D$).

Как видно, первые два слагаемых в (3) можно интерпретировать как гамильтониан частицы в одномерной яме, взаимодействующей с полем волны (V_n — оператор взаимодействия). Предполагаем, что температура среды низкая настолько, что можно рассматривать лишь два нижних энергетических уровня

в этой яме. Оператор $\hat{H}_n$, действующий на подпространстве двух нижних состояний, будем называть гамильтонианом кубита с номером n. Каждый мостик формирует, таким образом, двухуровневую систему, а описанная ранее структура является последовательностью идентичных кубитов. Каждый кубит описывается своей волновой функцией ψ_n, а энергия возбужденного состояния равна $\hbar\varepsilon$. Предположение о малости температур означает $\hbar\varepsilon \gg k_B T$, кроме того, будем полагать, что электромагнитное поле не возбуждает высоколежащие уровни, так что волновая функция представляет собой суперпозицию основного $|0\rangle$ и первого возбужденного $|1\rangle$ состояний:

$$|\psi_n(t)\rangle = c_0(n,t)|0\rangle \exp\left(\frac{i\varepsilon t}{2}\right) + c_1(n,t)|1\rangle \exp\left(-\frac{i\varepsilon t}{2}\right). \tag{5}$$

Уравнение Шредингера для волновой функции ψ_n сводится к системе уравнений для коэффициентов c_0 и c_1:

$$i\hbar \dot{c}_\alpha(n,t) = \sum_{\beta=0,1} V_{n,\alpha\beta}(t)\, c_\beta(n,t), \quad (\alpha,\beta) = 0,1. \tag{6}$$

В случае, когда электромагнитное поле слабое, $a_n \ll 1$, оператор взаимодействия (4) может быть приближенно записан в виде:

$$\hat{V}_n = E_J a_n^2 \cos(\varphi_n). \tag{7}$$

Используя лагранжиан (1), получаем уравнения для поля волны в области сверхпроводящих мостиков $0 \leqslant n \leqslant N$:

$$\ddot{a}_n - \upsilon^2 (a_{n+1} + a_{n-1} - 2a_n) = -\chi \sin(a_n), \tag{8}$$

где

$$\upsilon^2 = \frac{W\Phi_0^2}{32\pi^3 L_0 D}\left(\frac{E_J}{\omega_J^2} + \frac{\Phi_0^2 L_0 W}{32\pi^3 c^2 D}\right)^{-1},$$

$$\chi = E_J \left(\frac{E_J}{\omega_J^2} + \frac{\Phi_0^2 L_0 W}{32\pi^3 c^2 D}\right)^{-1} \langle \psi_n | \cos(\varphi_n) | \psi_n \rangle.$$

Уравнение для поля волны в областях с мостиками из нормального металла ($n < 0$ и $n > N$) можно получить из (8), если положить $\chi = 0$. В случае, когда $a_n \ll 1$, в уравнении (8) $\sin(a_n) \approx a_n$ и уравнение (8) для непрерывной среды можно записать в виде:

$$\frac{\partial^2 a}{\partial t^2} - \tilde{\upsilon}^2 \frac{\partial^2 a}{\partial z^2} = -\chi(z,t)\, a, \tag{9}$$

где

$$\tilde{\upsilon} = \upsilon L_0,$$

$$\chi(z,t) = E_J \left(\frac{E_J}{\omega_J^2} + \frac{\Phi_0^2 L_0 W}{32\pi^3 c^2 D}\right)^{-1} \langle \psi(z,t) | \cos(\varphi) | \psi(z,t) \rangle,$$

$$\chi_{\alpha\beta} = E_J \left(\frac{E_J}{\omega_J^2} + \frac{\Phi_0^2 L_0 W}{32\pi^3 c^2 D} \right)^{-1} \left\langle \alpha \left| \cos(\varphi) \right| \beta \right\rangle.$$

В дальнейшем мы используем уравнение (9) для теоретического анализа в квазилинейном приближении.

Система связанных уравнений (6) и (8) полностью описывает распространение электромагнитных волн в волноводной линии. В дальнейшем, при ее численном решении, для описания эволюции квантовых систем будет использоваться схема Кэли и метод канонических преобразований для численного решения волнового уравнения.

МЕТОД СОЗДАНИЯ ПЕРИОДИЧЕСКОЙ МОДУЛЯЦИИ НАСЕЛЕННОСТИ В ВОЛНОВОДНОЙ ЛИНИИ С ДЖОЗЕФСОНОВСКИМИ ПЕРЕХОДАМИ

Обсудим теперь возможность создания «квантового фотонного кристалла» в рассматриваемой цепочке джозефсоновских переходов. Покажем, что если состояния кубитов периодически модулированы в пространстве с периодом L_m, то волноводная линия, заполненная джозефсоновскими переходами, будет проявлять свойства фотонных кристаллов. Предварительно отметим, что если джозефсоновская подсистема находится в нормальном (не сверхпроводящем состоянии), то логично положить $\chi = 0$ в уравнении (8) и общее решение можно записать в виде модулированных волновых пакетов, распространяющихся в противоположных направлениях:

$$a_n(t) = \sum_k A_k^+ \exp(i(knL_0 - \omega(k)t)) + A_k^- \exp(i(knL_0 + \omega(k)t)), \qquad (10)$$

где $\omega(k) = 2\upsilon \left| \sin(kL_0/2) \right|$. При этом, в случае непрерывной среды, когда $L_0 \to 0$, как обычно, от суммирования в (10) необходимо перейти к интегрированию по первой зоне Бриллюэна, а $\omega(k)$ заменить на $\omega = \tilde{\upsilon} k$.

Возможность управлять откликом среды, не изменяя ее микроструктуру, представляется крайне интересной. Может быть предложено несколько подходов для создания периодической модуляции населенности в джозефсоновских структурах. В настоящей работе предлагается использовать для этой цели электромагнитные импульсы, распространяющиеся в волноводной линии, заполненной джозефсоновскими переходами, навстречу друг другу и осуществляющие накачку системы. Наша идея базируется на хорошо известном явлении интерференции волн. Также как два когерентных источника могут создавать интерференционную картину на поверхности воды, в данном случае два электромагнитных импульса благодаря интерференции могут создать периодическую модуляцию населенности кубитов.

Для того чтобы эффективно возбуждать систему кубитов, положим, что удвоенная частота импульса ω близка к частоте перехода между основным и возбужденным состояниями кубита ε ($2\omega \approx \varepsilon$) [Rabi, 1937; Fain, Khanin, 2003; Allen, Eberly, 1975]. Это следует из того факта, что в описываемой системе оператор

взаимодействия (7) квадратично зависит от амплитуды волны и, следовательно, связывает уровни кубита посредством двухфотонного поглощения. Кроме того, предположим, что ширина импульсов l удовлетворяет неравенству

$$\frac{2\pi\tilde{\upsilon}}{l} \ll \omega. \tag{11}$$

В этом случае для описания эволюции состояний кубитов справедливо резонансное приближение [Rabi, 1937; Fain, Khanin, 2003]. Таким образом, уравнение (6) дает:

$$\begin{aligned} i\hbar\dot{c}_0(z,t) &= a^2(z,t)\, d_{00}c_0 + a^2(z,t)\, d_{01}c_1 \exp(-i\varepsilon t), \\ i\hbar\dot{c}_1(z,t) &= a^2(z,t)\, d_{10}c_0 \exp(i\varepsilon t) + a^2(z,t)\, d_{11}c_1, \end{aligned} \tag{12}$$

где матричные элементы $d_{\alpha\beta} = E_J \left\langle \alpha \middle| \cos(\varphi_n) \middle| \beta \right\rangle$ и $z = nL_0$. Величина поля в области n-го кубита складывается из значений полей первого и второго импульсов в этой области: $a_n = a_n^{(1)} + a_n^{(2)}$. Мы будем рассматривать функции $a_n^{(1)}(t)$ и $a_n^{(2)}(t)$ как заданные, это справедливо, если импульсы достаточно сильные и передается лишь незначительная часть их энергии системе кубитов, что и предполагается. Иными словами, в случае довольно сильных импульсов решение (9) может быть записано в виде

$$\begin{aligned} a^{(1)}(z,t) &= \exp\left(-\frac{(z-\upsilon t)^2}{l^2}\right)\left(A_1 \exp[i(kz-\omega t)] + \text{к. с.}\right), \\ a^{(2)}(z,t) &= \exp\left(-\frac{(z+\upsilon t)^2}{l^2}\right)\left(A_2 \exp[i(kz+\omega t+\varphi_0)] + \text{к. с.}\right), \end{aligned} \tag{13}$$

где A_1 и A_2 — амплитуды первого и второго импульсов соответственно (введенные здесь A_1 и A_2 — действительны); φ_0 — начальная фаза второго импульса. Не ставя перед собой задачу найти точное значение коэффициентов c_0, c_1 после прохождения импульсов через структуру, но получить некоторые общие свойства их пространственной зависимости, мы будем использовать квазимонохроматическое приближение (11), т. е. полагать, что амплитуда импульсов слабо меняется во времени:

$$\begin{aligned} c_0(z,t) &= \left[R_1 \exp\left(i\sqrt{\left|\Omega\right|^2 + \frac{\gamma^2}{4}} \cdot t \right) + R_2 \exp\left(-i\sqrt{\left|\Omega\right|^2 + \frac{\gamma^2}{4}} \cdot t \right) \right] \exp\left[i\left(\frac{\gamma}{2} - \Delta_0\right) t \right], \\ \gamma &= \Delta + (\Delta_0 - \Delta_1), \\ \Delta_0 &= \frac{2d_{00}}{\hbar}\left(A_1^2 + A_2^2 + 2A_1A_2\cos(2kz+\varphi_0)\right), \\ \Delta_1 &= \frac{2d_{11}}{\hbar}\left(A_1^2 + A_2^2 + 2A_1A_2\cos(2kz+\varphi_0)\right), \end{aligned} \tag{14}$$

где Δ — отстройка от резонанса, $\Delta = 2\omega - \varepsilon$; R_1, R_2 — константы, зависящие от начального состояния. Пусть в начальный момент времени $t = 0$ состояние

кубита с номером n' определялось коэффициентами $c_0 = 1$, $c_1 = 0$, тогда для $c_1(n', t)$ получаем:

$$\tilde{n}_1(n', t) = \frac{-i\Omega}{\sqrt{|\Omega|^2 + \frac{\gamma^2}{4}}} \sin\left(\sqrt{|\Omega|^2 + \frac{\gamma^2}{4}} \cdot t\right) \exp\left[-i\left(\frac{\gamma}{2} + \Delta_1\right)t\right]. \quad (15)$$

Таким образом, если считать, что $\gamma = 0$, то c_1 изменяется периодически с частотой Раби Ω [Fain, Khanin, 2003], а частота Раби Ω определяется выражением:

$$\Omega(z) = \frac{d_{01}}{\hbar}\left| A_1^2 \exp\left[-i(2kz + \varphi_0)\right] + A_2^2 \exp\left[i(2kz + \varphi_0)\right] + 2A_1A_2 \right|, \quad (16)$$

или для случая равных амплитуд импульсов $A_1 = A_2 = A$:

$$\Omega(z) = \frac{2d_{01}A^2}{\hbar}\left(1 + \cos(2kz + \varphi_0)\right).$$

Последнее означает, что частота Раби есть периодическая функция координаты z с периодом π/k, т. е. $\Omega(z) = \Omega(z + \pi/k)$, что непосредственно следует из (16). Так, если для одних точек $\cos(2kz + \varphi_0) = 1$ и значение частоты Раби максимально $\Omega = d_{01}(A_1 + A_2)^2/\hbar$, то для других точек $\cos(2kz + \varphi_0) = -1$ и частота Раби равна $\Omega = d_{01}(A_1 - A_2)^2/\hbar$, а двухуровневые системы, находящиеся в этих точках, возбуждаются менее эффективно и в частном случае двух одинаковых импульсов $A_1 = A_2$ не возбуждаются вообще. Продолжая анализ уравнения (15), следует отметить, что даже при наличии отстройки от резонанса пространственная периодичность накачки не исчезает. Действительно, величина

$$\frac{\gamma^2}{4} + |\Omega|^2 = \frac{\left(\Delta - \frac{2(d_{11} - d_{00})}{\hbar}\left(A_1^2 + A_2^2 + 2A_1A_2\cos(2kz + \varphi_0)\right)\right)^2}{4} + \frac{4|d_{01}|^2}{\hbar^2}\left(\begin{array}{c}\left(1 + \cos(2kz + \varphi_0)\right)^2 A_1^2A_2^2 + \left(1 + \cos(2kz + \varphi_0)\right)(A_1 - A_2)\left(A_1^2 + A_2^2\right)A_2 + \\ + \frac{(A_1 - A_2)^2\left(A_1^2 - 4A_1A_2 + A_2^2\right)}{4}\end{array}\right)$$

является периодической функцией z и, соответственно, c_1 также представляет собой периодическую функцию z. Для наглядности приведем ту же самую величину для случая равных амплитуд импульсов $A_1 = A_2$ и точного резонанса $\Delta = 0$:

$$\frac{\gamma^2}{4} + |\Omega|^2 = \frac{4A^2}{\hbar^2}\left((d_{11} - d_{00})^2 + |d_{01}|^2\right)\left(1 + \cos(2kz + \varphi_0)\right)^2.$$

Полученная зависимость частоты Раби от координаты указывает путь создания периодической модуляции населенности в линии с джозефсоновскими переходами. На рис. 2 (см. с. 17) изображены в начальный момент времени два импульса, локализованные в нормальных областях. После прохождения импуль-

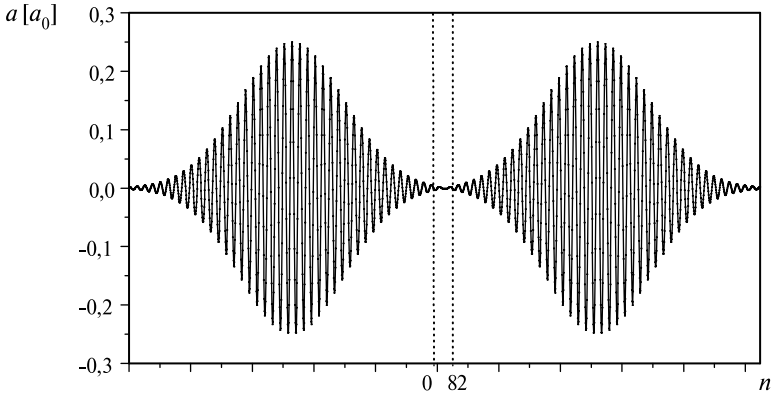


Рис. 2. Импульсы электромагнитного поля, локализованные в начальный момент времени в нормальных областях. Область сверхпроводящих мостиков располагается посередине. Параметры импульсов: скорость $\tilde{\upsilon} = c$; частота $\omega = \omega_J/2$; волновое число $k = 2\pi/(25L_0)$; ширина импульсов $l = 240L_0$; амплитуда импульсов $A = 0{,}25$. Матричный элемент $d_{01} = 0{,}1\hbar/4$; $\chi_{00} = 0{,}1\omega_J^2$; $\chi_{11} = 0{,}3\omega_J^2$; $c_0(n, t = 0) = 1$; $c_1(n, t = 0) = 0$. Общая длина области расчетов $\tilde{L} = 2048L_0$; число ячеек — 2048. Отстройка $\Delta = 0{,}03\omega_J$

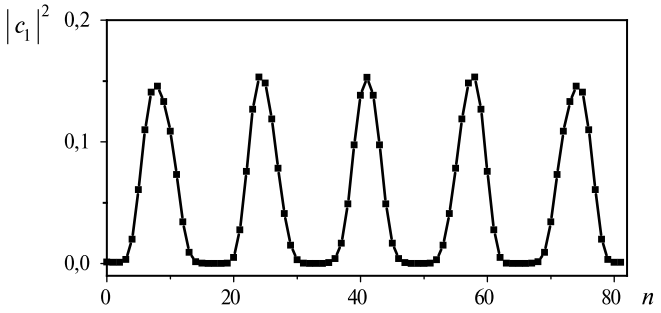


Рис. 3. Возникновение периодической модуляции населенности уровней кубитов вследствие прохождения через активную область двух встречных импульсов. Апериодичность связана с конечной шириной накачивающих импульсов. Длина активной области — $82L_0$

сами области сверхпроводящих мостиков там возникает периодическая модуляция населенности уровней кубитов, как это показано на рис. 3. Небольшое искажение периодичности связано с немонохроматичностью волн и может быть устранено, если использовать более широкие импульсы. Из рис. 3 видно, что период модуляции для указанных параметров примерно равен 16,5 размерам ячеек, что в пересчете в единицы длины равно $16{,}5L_0$. Это значение хорошо совпадает с величиной периода модуляции по оси z, следующей из формулы (16), согласно которой $\Omega(z) = \Omega(z + \pi/k)$, где $\pi/k \approx 16{,}2L_0$ (с учетом того, что в активной области $k \approx \left(\omega^2 - \chi_{00}\right)^{1/2} \Big/ \tilde{\upsilon} \approx 1{,}194$, $\chi_{00} = 0{,}1$). Отметим также, что для указанных параметров начальных импульсов величина $(\Delta_1 - \Delta_0) \approx 0{,}03$, таким образом, была взята отстройка $\Delta = 0{,}03$, чтобы обеспечить равенство $\gamma \approx 0$ (см. уравнение (14)). При условии $\Delta = 0$ эффективное возбуждение системы кубитов не наблюдается.

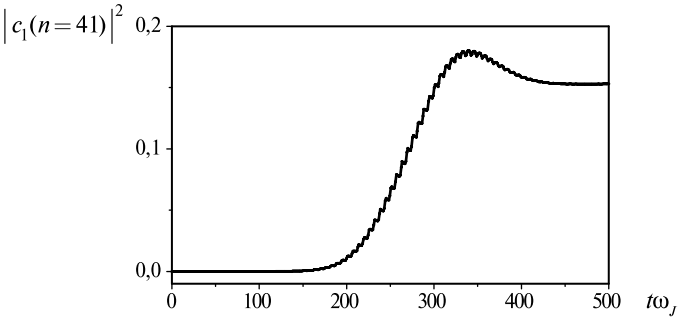


Рис. 4. Зависимость населенности возбужденного уровня кубита с номером $n = 41$ от времени t, измеряемом в обратных единицах ω_J

На рис. 4 приведена зависимость населенности возбужденного уровня кубита с номером $n = 41$ от времени t, измеряемом в обратных единицах ω_J.

Полученные результаты демонстрируют возможность создания периодической модуляции населенности в волноводной линии с джозефсоновскими переходами при помощи электромагнитных импульсов. При этом период модуляции можно эффективно контролировать, он определяется волновым числом k. Амплитудой модуляции можно также управлять, изменяя амплитуды импульсов или их ширину (см. (15), (16)).

ДИСПЕРСИОННОЕ УРАВНЕНИЕ ДЛЯ ВОЗБУЖДЕНИЙ В ФОТОННОМ КРИСТАЛЛЕ

Пусть в джозефсоновской линии путем воздействия импульсов сильного внешнего поля создано периодическое распределение населенностей, которое согласно полученным ранее выражениям (5), (8), (15) и (16), приводит к модуляции функции $\chi(z)$:

$$\chi(z) = \chi_0 + \chi_c \left(1 + \cos\left(\frac{2\pi z}{L_m}\right)\right), \tag{17}$$

где L_m — период модуляции; χ_0 — константа, определяемая квантовыми состоянием кубитов с минимальным $|c_{1,\min}|$:

$$\chi_0 = \chi_{00} |c_{0,\min}|^2 + \chi_{11} |c_{1,\min}|^2;$$

для случая, изображенного на рис. 3, $\chi_0 = \chi_{00}$ (поскольку при выполнении условия $\cos(2kz + \varphi_0) = -1$ кубиты находятся в невозбужденном состоянии, $|c_{1,\min}| = 0$). Величина χ_c показывает, насколько сильно возбуждены кубиты, для которых выполняется условие $\cos(2kz + \varphi_0) = +1$ (им соответствует значение коэффициента $c_1 = c_{1,\max}$):

$$\chi_c = \frac{(\chi_{11} - \chi_{00})}{2}\left(|c_{1,\max}|^2 - |c_{1,\min}|^2\right).$$

Для случая, изображенного на рис. 3, соответственно

$$\chi_c = \frac{(\chi_{11} - \chi_{00})}{2} \left| c_{1,\max} \right|^2.$$

Приближенное уравнение для волн с малой амплитудой в модулированной среде с $\chi(z)$, определенной (17), имеет вид:

$$\frac{\partial^2 a}{\partial t^2} - \tilde{\upsilon}^2 \frac{\partial^2 a}{\partial z^2} = -\chi(z) a. \tag{18}$$

Решение уравнения (18) будем искать по аналогии с приближением слабой связи в теории твердого тела [Ashcroft, Mermin, 1976] в виде:

$$a_k(z,t) = u_k(z) \exp(i(kz - \omega t)), \tag{19}$$

где $u_k(z)$ — периодическая функция с периодом L_m ($u_k(z) = u_k(z + L_m)$). Используя (17) и (18), получим дисперсионное соотношение:

$$\omega_k^2 - \tilde{\upsilon}^2 k^2 - W_0 = \pm \left| W_k \right|, \tag{20}$$

где

$$W_k = \frac{1}{L_m} \int\limits_{-L_m/2}^{L_m/2} \chi(z) \exp(i2kz)\, dz, \quad k = \frac{\pi n}{L_m}, \tag{21}$$

$$W_0 = \frac{1}{L_m} \int\limits_{-L_m/2}^{L_m/2} \chi(z)\, dz.$$

Из формулы (20) следует, что при каждом $k = \pi n / L_m$ в спектре частот будет наблюдаться щель, если соответствующая фурье-компонента $\chi(z)$ отлична от нуля (рис. 5, см. с. 20) и ее ширина определяется величиной

$$\delta\omega_n = \sqrt{\left(\frac{\pi\tilde{\upsilon} n}{L_m}\right)^2 + W_0 + \left| W_{k_n} \right|} - \sqrt{\left(\frac{\pi\tilde{\upsilon} n}{L_m}\right)^2 + W_0 - \left| W_{k_n} \right|} \approx \frac{W_{k_n}}{\sqrt{\left(\frac{\pi\tilde{\upsilon} n}{L_m}\right)^2 + W_0}}, \tag{22}$$

$$\left| W_{k_n} \right| \ll \left(\frac{\pi\tilde{\upsilon} n}{L_m}\right)^2 + W_0.$$

Щель наблюдается при $k_{n=1} = \pi / L_m$ (вблизи частоты $\omega_1 = \left((\tilde{\upsilon} k_1)^2 + W_0\right)^{1/2}$), для которого из формулы (21) следует $W_{k1} = \chi_c/2$, $W_0 = \chi_0 + \chi_c$, а ширина щели составляет

$$\delta\omega_1 \approx \frac{\chi_{\tilde{n}}}{\sqrt{\left(\frac{\pi\tilde{\upsilon} n}{L_m}\right)^2 + W_0}}. \tag{23}$$

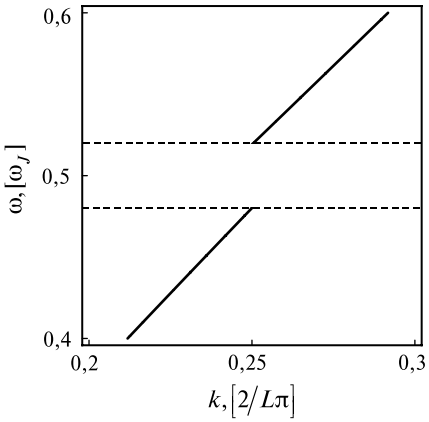


Рис. 5. Дисперсионная зависимость квантового фотонного кристалла. В силу пространственной периодичности χ на графике дисперсионной зависимости образуются энергетические щели (положение щели обозначено пунктиром). Их положение зависит от величины периода L_m, ширина — от периода L_m и величины χ_0, т. е. от квантового состояния, в котором находятся кубиты. Эта особенность позволяет управлять параметрами квантового фотонного кристалла путем изменения квантовых состояний, в которых находятся кубиты. Вычисления произведены для следующих параметров: $L_m = 25L_0/2$; $\chi_0 = \chi_{00}$, $\chi_c = 0{,}0195\omega_J^2$ $(\chi_{00} = 0{,}1\omega_J^2,\ \chi_{11} = 0{,}3\omega_J^2)$, $\tilde{\upsilon} = c$. Используется модуляция населенностей кубитов, рассчитанная на рис. 3

Отсюда следует, что для получения довольно широких щелей необходимы большие значения χ_c.

ПРОХОЖДЕНИЕ ЭЛЕКТРОМАГНИТНЫХ ИМПУЛЬСОВ ЧЕРЕЗ ВОЛНОВОДНУЮ ЛИНИЮ С ПЕРИОДИЧЕСКОЙ МОДУЛЯЦИЕЙ НАСЕЛЕННОСТИ КУБИТОВ

Рассмотрим теперь прохождение слабых электромагнитных импульсов через волноводную линию, когда функция $\chi(z)$ является периодической функцией координаты z с периодом L_m. Дисперсионное соотношение (20) показывает, что при определенных частотах электромагнитная волна будет испытывать отражение от области, содержащей последовательность кубитов с периодической модуляцией населенности. Это связано с существованием энергетических щелей в спектре этой среды (см. рис. 5).

Из рис. 5 видно, что вблизи значения $\omega = \pi\tilde{\upsilon}/L_m = \omega_J/2$ существует щель. Ее ширина согласно формуле (23) равна $\delta\omega_1 \approx 0{,}032\omega_J$. На рис. 6*б* (см. с. 21) представлен результат взаимодействия электромагнитного импульса, изображенного на рис. 6*а*, с последовательностью кубитов с периодической модуляцией населенности (см. рис. 3). Видно, что происходит частичное отражение импульса от структуры, а его частота $\omega = \omega_J/2$ лежит в области щели. Следует отметить, что частичное прохождение связано с тем, что ширина щели сравнима с неопреде-

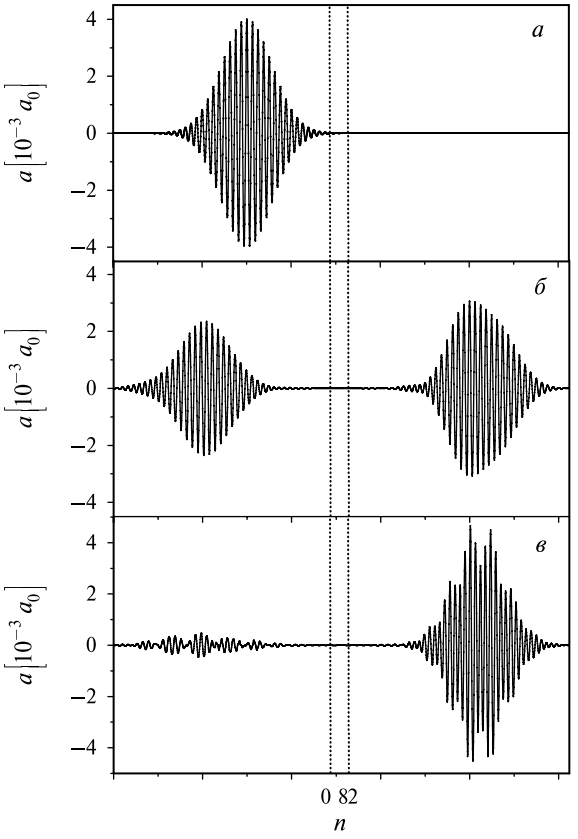


Рис. 6. Прохождение электромагнитных импульсов через волноводную линию с периодической модуляцией населенности кульбитов: *а* — пробный импульс в начальный момент времени. Амплитуда импульса $A = 4\cdot10^{-3}$; ширина импульса $l = 160L_0$; несущая частота $\omega = \omega_J/2$; скорость $\tilde{\upsilon} = c$; $\chi_{00} = 0{,}1\omega_J^2$, $\chi_{11} = 0{,}36\omega_J^2$. Параметры среды (области, содержащей джозефсоновские переходы) те же, что были использованы при расчетах рис. 2. Используется модуляция населенностей кубитов, рассчитанная на рис. 3; *б* — распределение поля в момент времени $t = 540\omega_J^{-1}$. Частичное прохождение импульса связано с его конечной шириной; *в* — распределение поля в момент времени $t = 540\omega_J^{-1}$ для импульса с несущей частотой, лежащей выше щели. Параметры импульса в момент времени $t = 0$: амплитуда $A = 4\cdot10^{-3}$; ширина $l = 160L_0$; несущая частота $\omega = 0{,}55\omega_J$; скорость $\tilde{\upsilon} = c$

ленностью частоты импульса $\Delta\omega = 2\pi\tilde{\upsilon}/l$, которая для указанных параметров составляет достаточно большую величину, равную $0{,}0785\omega_J$, превышающую ширину щели $\delta\omega_1 \approx 0{,}032\omega_J$. Для сравнения на рис. 6*в* приведено распределение поля для импульса с частотой, лежащей несколько выше щели, $\omega = 0{,}55\omega_J$. Лишь незначительная часть импульса отражается от области, содержащей кубиты, что свидетельствует о том, что несущая частота данного импульса не попадает в энергетическую щель квантового фотонного кристалла.

ЗАКЛЮЧЕНИЕ

В работе рассмотрено распространение электромагнитных импульсов в волноводной линии, содержащей джозефсоновские переходы, которые формируют двухуровневые системы (кубиты). Продемонстрирована возможность создания пространственной периодической модуляции населенности кубитов при помощи электромагнитных импульсов, распространяющихся навстречу друг другу вдоль волноводной линии. При этом период модуляции определяется длиной волны импульсов как половина длины волны импульсов. Полученная в результате структура проявляет свойства фотонных кристаллов, в ней могут наблюдаться энергетические щели в спектре частот, причем параметры такого кристалла определяются квантовым состоянием кубитов и могут варьироваться путем изменения параметров среды. Рассмотрено условие эффективного возбуждения кубитов. Показано также, что квантовое состояние последовательности кубитов можно анализировать при помощи слабых электромагнитных импульсов. Наличие периодической модуляции населенности кубитов можно обнаружить по сильно нелинейному отклику, проявляющемуся в существовании областей частот, в которых среда становится непрозрачной для слабых электромагнитных импульсов.

ЛИТЕРАТУРА

[Agranovich et al., 2004] *Agranovich V. M.*, *Shen Y. R.*, *Baughman R. H.*, *Zakhidov A. A.* Linear and nonlinear wave propagation in negative refraction metamaterials // Physical Rev. B. 2004. V. 69. P. 165112-1–165112-7.

[Allen, Eberly, 1975] *Allen L.*, *Eberly J. H.* Optical resonance and two-level atoms. N. Y.; L.; Sydney; Toronto: John Willey and Sons, 1975. 233 p.

[Ashcroft, Mermin, 1976] *Ashcroft N. W.*, *Mermin N. D.* Solid State Physics. Harcourt: Orlando, 1976. 848 p.

[Barone, Paterno, 1982] *Barone A.*, *Paterno G.* Physics and applications of the Josephson effect. N. Y.: Wiley, 1982. 529 p.

[Du et al., 2006] *Du C. G.*, *Chen H. Y.*, *Li S. Q.* Quantum left-handed metamaterial from superconducting quantum-interference devices // Physical Rev. B. 2006. V. 74. P. 113105-1–113105-4.

[Fain, Khanin, 2003] *Fain V. M.*, *Khanin Ya. I.* Quantum Electronics. Cambridge, Massachusetts, USA: The MIT Press, 2003. 350 p.

[Korobkin et al., 2006] *Korobkin D.*, *Urzhumov Ya.*, *Shvets G.* Enhanced near-field resolution in midinfrared using metamaterials // J. Optical Society of America B. 2006. V. 23. P. 468–478.

[Lazarides, Tsironis, 2007] *Lazarides N.*, *Tsironis G. P.* A rf superconducting quantum interference device metamaterials // Applied Physics Letters. 2007. V. 90. P. 163501-1–163501-3.

[Likharev, 1986] *Likharev K. K.* Dynamics of Josephson junctions and circuits. Gordon and Breach, 1986, 632 p.

[Nakamura et al., 2002] *Nakamura Y.*, *Pashkin Yu. A. Tsai J. S.* Rabi Oscillations in a Josephson-Junction Charge Two-Level System // Physical Rev. Letters. 2002. V. 87. P. 246601-1–246601-4.

[Rabi, 1937] *Rabi I. I.* Space Quantization in a Gyrating Magnetic Field // Physical Rev. 1937. V. 51. P. 652–654.

[Rakhmanov et al., 2008] *Rakhmanov A. L.*, *Zagoskin A. M.*, *Savel'ev S.*, *Nori F.* Quantum metamaterials: Electromagnetic waves in a Josephson qubit line // Physical Rev. B. 2008. V. 77. P. 144507-1–144507-7.

[Ricci et al., 2005] *Ricci M.*, *Orloff N.*, *Anlage S. M.* Superconducting metamaterials // Applied Physics Letters. 2005. V. 87. P. 034102-1–034102-3.

[Savel'ev et al., 2005] *Savel'ev S.*, *Yampol'skii V. A.*, *Nori F.* Surface Josephson Plasma Waves in Layered Superconductors // Physical Rev. Letters. 2005. V. 95. P. 187002-1–187002-4.

[Shvets, 2003] *Shvets G.* Photonic approach to making a material with a negative index of refraction // Physical Rev. B. 2003. V. 67. P. 035109-1–035109-8.

[Veselago, 2003] *Veselago V. G.* Electrodynamics of materials with negative index of refraction // Soviet Physics Uspekhi. 2003. V. 46 (7). P. 764–768.

[Yampol'skii et al., 2007] *Yampol'skii V. A.*, *Savel'ev S.*, *Usatenko O. V.*, *Mel'nik S. S.*, *Kusmartsev F. V.*, *Krokhin A. A.*, *Nori F.* Controlled terahertz frequency response and transparency of Josephson chains and superconducting multilayers // Physical Rev. B. 2007. V. 75. P. 014527-1–014527-7.

[Zhang et al., 2006] *Zhang S.*, *Fan W.*, *Malloy K. J.*, *Brueck S. R. J.*, *Panoiu N. C.*, *Osgood R. M.* Demonstration of metal-dielectric negative-index metamaterials with improved performance at optical frequencies // J. Optical Society of America B. 2006. V. 23. P. 434–438.

[Zharov et al., 2003] *Zharov A. A.*, *Shadrivov I. V.*, *Kivshar Yu. S.* Nonlinear Properties of Left-Handed Metamaterials // Physical Rev. Letters. 2003. V. 91. P. 037401-1–037401-4.

[Ziolkowski, 2006] *Ziolkowski R. W.* Ultrathin, metamaterial-based laser cavities // J. Optical Society of America B. 2006. V. 23. P. 451–460.

CONTROL OF QUBITS POPULATION BY USING OF ELECTROMAGNETIC PULSES

A. V. Shvetsov[1], A. M. Satanin[1], A. J. Gelman[2]

[1] *Nizhny Novgorod State University* (*NNSU*)

[2] *The Institute of Applied Physics of the Russian Academy of Sciencies* (*IAP RAS*), *Mocsow*

The possibility to create a periodical modulation of the population in the waveguide filled with identical Josephson junctions each of which at low excitation may be considered as two-level quantum system (qubit) is studied. It is shown that the dispersion characteristics of the medium depend on the states of qubits, which can be initialized by electromagnetic pulses propagating in the waveguide. Analytically and numerically demonstrated that the counter propagating pulses allow to carry out the pumping system and a periodic modulation of the populations of qubits with a period equal to half the wavelength pulses. The resulting quantum structures exhibit the properties of photonic crystals. The conditions of qubits efficient excitation in the structure are examined in detail.

Keywords: metamaterials, Josephson junctions, qubits.

Shvetsov Alexander Vladimirovich — scientist, candidate of physical and mathematical sciences, e-mail: alexshdze@mail.ru.

Satanin Arkady Mikhailovich — doctor of physical and mathematical sciences, professor, e-mail: sarkady@mail.ru.

Gelman Alexander Josephovich — junior scientist, candidate of physical and mathematical sciences, e-mail: gelman@appl.sci-nnov.ru.

УДК 530.145

ПЕРЕХОДЫ ЛАНДАУ – ЗИНЕРА МЕЖДУ СОСТОЯНИЯМИ ВЗАИМОДЕЙСТВУЮЩИХ КУБИТОВ

М. В. Денисенко, А. М. Сатанин

Нижегородский государственный университет им. Н. И. Лобачевского (*ННГУ им. Н. И. Лобачевского*)

Изучается ландау-зинеровское туннелирование в системе связанных кубитов, помещенных в сильное переменное поле. Показано, что если туннельное расщепление уровней кубитов мало, то резонансные условия для многофотонных переходов оказываются зависящими от константы связи. Результаты, полученные в рамках теории возмущений, подтверждаются численными расчетами с использованием квазиэнергетического представления. Изучено влияние константы связи на условия возникновения целых и дробных резонансов. Выполнены расчеты населенностей уровней кубитов и исследованы эффекты формирования интерференционной картины населенностей, обусловленной переходами Ландау – Зинера. Обсуждается возможность экспериментального наблюдения обнаруженных эффектов.

Работа выполнена при финансовой поддержке АВЦП «Развитие потенциала высшей школы», фонда «Династия» и РФФИ (проект № 11-02-97058-р_поволжье_а).

Ключевые слова: кубиты, переходы Ландау-Зинера, квазиэнергия, многофотонные переходы.

ВВЕДЕНИЕ

Воздействие переменного внешнего поля на макроскопические квантовые системы позволяет глубже понять проявление квантовых когерентных эффектов и открывает новые перспективы их применения для квантовых вычислений [Mooij, 2005]. Примером таких устройств могут служить сверхпроводящие джозефсоновские кубиты. Манипуляция состояниями кубитов осуществляется с использованием метода раби-спектроскопии [Nakamura et al., 1999; van der Wal et al., 2000; Izmalkov et al., 2004]. В данном методе частота внешнего поля настраивается на точный резонанс, определяемый расстоянием между уровнями, а амплитуда внешнего поля считается малой по сравнению со всеми энергетическими характеристиками системы.

Другой способ изучения когерентной динамики — возбуждение системы сильным полем на фиксированной частоте сигнала, что положено в основу экспериментального метода изучения многофотонных переходов (амплитудной спектроскопии). К настоящему времени данная методика успешно продемонстрирована на примере одиночных сверхпроводящих кубитов [Berns et al., 2006, 2008; Oliver et al., 2005]. При этом система эволюционирует адиабатически, кроме как в непосредственной близости от квазипересекающихся уровней, где наблюдаются квантово-когерентные переходы Ландау – Зинера и интерференция Щтюкельберга.

Денисенко Марина Валерьевна — студентка, e-mail: mar.denisenko@gmail.com.

Сатанин Аркадий Михайлович — доктор физико-математических наук, профессор, e-mail: sarkady@mail.ru.

В данной работе исследуются многофотонные переходы, обусловленные влиянием сильного внешнего поля на систему взаимодействующих кубитов. Два связанных кубита представляют собой пример квантовой системы, где нетривиально проявляются качественно новые эффекты, обусловленные «перепутыванием» состояний. Подробно изучены процессы возникновения многофотонных переходов, как теоретически в рамках теории возмущений (основываясь на малости туннельных расщеплений уровней кубитов, что соответствует экспериментам группы Е. Ильичёва [Izmalkov et al., 2004; Grajcar et al., 2005; Il'ichev et al., 2010]), так и с использованием численного моделирования в рамках квазиэнергетического подхода. Исследовано влияние константы связи кубитов на положения многофотонных переходов и характер формирования интерференционной картины населенностей, обусловленной ландау – зинеровскими переходами.

МОДЕЛЬ СИСТЕМЫ И ТЕОРЕТИЧЕСКИЙ АНАЛИЗ

Основные черты динамического поведения связанных кубитов (i = 1, 2), изучаемых в экспериментах [Grajcar et al., 2005; Il'ichev et al., 2010; Groot et al., 2010], могут быть описаны гамильтонианом

$$H = -\frac{1}{2}\begin{pmatrix} \varepsilon_1(t)+\varepsilon_2(t)+J & \Delta_2 & \Delta_1 & 0 \\ \Delta_2 & \varepsilon_1(t)-\varepsilon_2(t)-J & 0 & \Delta_1 \\ \Delta_1 & 0 & -\varepsilon_1(t)+\varepsilon_2(t)-J & \Delta_2 \\ 0 & \Delta_1 & \Delta_2 & -\varepsilon_1(t)-\varepsilon_2(t)+J \end{pmatrix}, \quad (1)$$

где Δ_i — туннельное расщепление уровней i-го кубита; J — константа взаимодействия. Параметры «смещения» $\varepsilon_i(t) \approx f_i^{dc} + f_i^{ac}$ описывают воздействие постоянного (f_i^{dc}) и переменного (f_i^{ac}) магнитных полей, которые реализуются с помощью управляющих катушек, схематически изображенных на рис. 1*а*, что позволяет манипулировать состояниями кубитов.

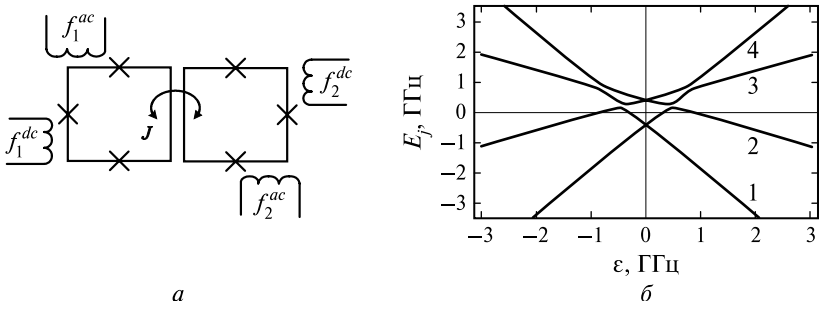


Рис. 1. Схема связанных джозефсоновских кубитов с управляемыми катушками (*а*); зависимость собственных значений системы E_j (j = 1...4) при изменении управляющего параметра $\varepsilon = \varepsilon_1 = 2\varepsilon_2$ (*б*). Параметры кубитов: $\Delta_1 = \Delta_2 = 0{,}2$ ГГц; $J = 0{,}8$ ГГц

Управляющие параметры можно задать в виде:

$$\varepsilon_1(t) = \varepsilon_1 + A_1 \cos(\omega t + \varphi), \quad \varepsilon_2(t) = \varepsilon_2 + A_2 \cos(\omega t + \varphi). \tag{2}$$

Энергетический спектр системы взаимодействующих кубитов, когда переменное поле отсутствует ($A_i = 0$ в формуле (2)), несложно найти, решая стационарное уравнение Шредингера $H\left|\psi_j\right\rangle = E_j\left|\psi_j\right\rangle$, где $j = 1...4$ (рис. 1*б*). Отметим, что все энергетические величины измеряются в гигагерцах, причем выбрана система единиц, в которой постоянная Планка $\hbar = 1$.

При воздействии на систему импульсов переменного поля, длительность которых намного больше периода поля $T = 2\pi/\omega$, возбуждаются переходы между стационарными состояниями системы E_j. Для соответствия теоретической модели проводимым экспериментам [Oliver et al., 2005; Berns et al., 2006, 2008] следует учесть возможные флуктуации длительностей импульсов на периоде T (фаза φ в формуле (2)), а также разброс времен приходов сигналов на систему [Денисенко и др., 2010; Shirley, 1965]. Фазы в выражении (2) выбраны одинаковыми, так как считается, что сигнал, возбуждающий систему, приходит одновременно на оба кубита, хотя величины амплитуд сигналов для отдельных кубитов могут различаться.

Анализируя экспериментальные значения энергетических характеристик систем сверхпроводящих кубитов, можно заметить, что в ряде случаев туннельные расщепления уровней кубитов Δ_i малы по сравнению с остальными параметрами системы [Grajcar et al., 2005; Il'ichev et al., 2010], что позволяет использовать теорию возмущений, разложив оператор эволюции и волновые функции системы в ряды по степеням Δ_i. Представим гамильтониан системы (1) в виде

$$H(t) = H_0(t) + V, \tag{3}$$

где $H_0(t)$ — диагональная часть гамильтониана (1); V — малая поправка ~Δ_i. Волновые функции $\left|\psi_j(0)\right\rangle$, где $j = 1...4$, рассчитывают в рамках стационарной теории возмущений. Динамика же подчиняется уравнению Шредингера:

$$i\frac{\partial}{\partial t}\left|\psi(t)\right\rangle = H(t)\left|\psi(t)\right\rangle, \tag{4}$$

где параметры гамильтониана (1) зависят от времени согласно (2). Введем оператор эволюции, определяемый соотношением

$$\left|\psi(t)\right\rangle = U(t)\left|\psi(0)\right\rangle, \tag{5}$$

который будем искать в виде ряда по степеням Δ_i: $U(t) = U^{(0)}(t) + U^{(1)}(t)$, где поправка нулевого порядка определяется диагональной частью гамильтониана $U_j^{(0)}(t) = \mathrm{e}^{i\varphi_j(t)}\mathrm{I}$ (I — единичная матрица фазы $\varphi_j(t) = \int_0^t H_0(\tau)_{jj}\mathrm{d}\tau$), а поправка первого порядка определяется выражением $U^{(1)}(t) = -i\mathrm{e}^{i\varphi_j(t)}\int_0^t U^{(0)}(\tau)^{-1} V U^{(0)}(\tau)\,\mathrm{d}\tau$. В рассматриваемом базисе оператор эволюции изображается матрицей:

$$U(t)=-\frac{1}{2}\begin{pmatrix} -2e^{i\varphi_1(t)} & -i\Delta_2 e^{i\varphi_1(t)}a_{12} & -i\Delta_1 e^{i\varphi_1(t)}a_{13} & 0 \\ -i\Delta_2 e^{i\varphi_2(t)}a_{12}^* & -2e^{i\varphi_2(t)} & 0 & -i\Delta_1 e^{i\varphi_2(t)}a_{24} \\ -i\Delta_1 e^{i\varphi_3(t)}a_{13}^* & 0 & -2e^{i\varphi_3(t)} & -i\Delta_2 e^{i\varphi_3(t)}a_{34} \\ 0 & -i\Delta_1 e^{i\varphi_4(t)}a_{24}^* & -i\Delta_2 e^{i\varphi_4(t)}a_{34}^* & -2e^{i\varphi_4(t)} \end{pmatrix}, \quad (6)$$

где введены обозначения $a_{ij}=\int_0^t e^{-i(\varphi_i(\tau)-\varphi_j(\tau))}d\tau$. Будем полагать, что в начальный момент времени система находилась в основном состоянии $|\psi_1(0)\rangle$ (определяемом в рамках стационарной теории возмущения), тогда волновая функция системы (5) с точностью до величин ~Δ_i запишется в виде:

$$\psi(t)=\begin{pmatrix} e^{i\varphi_1(t)} \\ \frac{i}{2}\Delta_2 e^{i\varphi_2(t)}a_{12}^* + d^{(2)}e^{i\varphi_2(t)} \\ \frac{i}{2}\Delta_1 e^{i\varphi_3(t)}a_{13}^* + d^{(1)}e^{i\varphi_3(t)} \\ 0 \end{pmatrix}, \quad (7)$$

где введено обозначение $d^{(i)}=\Delta_i/2(\varepsilon_i\pm J)$. При этом амплитуда перехода между основным и первым возбужденным состоянием равна:

$$f_{1\to 2}=\langle\psi_2(0)|\psi(t)\rangle=\frac{i\Delta_2 A_2}{2(\varepsilon_2+J)}e^{i\varphi_2(t)}\int_0^t d\tau\cos(\omega\tau+\varphi)\,e^{i(\varepsilon_2+J)\tau+i\frac{A_2}{\omega}(\sin(\omega\tau+\varphi)-\sin\varphi)}. \quad (8)$$

Используя разложение $e^{ix\sin\omega t}=\sum_{n=-\infty}^{\infty}J_n(x)\,e^{in\omega t}$ и соотношение для функций Бесселя $J_{n+1}(x)+J_{n-1}(x)=\frac{2n}{x}J_n(x)$, найдем вероятность перехода:

$$P(t)_{1\to 2}=\frac{\Delta_2^2\omega^2}{4(\varepsilon_2+J)^2}\int_0^t d\tau\cdot e^{i(\varepsilon_2+J)\tau+in\omega\tau}\times$$
$$\times\int_0^t d\tau'\cdot e^{-i(\varepsilon_2+J)\tau'-in'\omega\tau'}\sum_{n,n'}nn'J_n\left(\frac{A_2}{\omega}\right)J_{n'}\left(\frac{A_2}{\omega}\right)e^{i\varphi(n-n')}. \quad (9)$$

Исходя из предположения о том, что времена приходов импульсов и их длительности случайны, нетрудно усреднить (9) по случайным фазам φ и длительностям импульсов t $(t\gg T)$ (см. [Shirley, 1965]), что приводит к выражению

$$\bar{P}_{1\to 2}=\frac{\Delta_2^2\omega^2}{2(\varepsilon_2+J)^2}\sum_n n^2 J_n\left(\frac{A_2}{\omega}\right)\frac{1}{(\varepsilon_2+J+n\omega)^2}. \quad (10)$$

Как видно из выражения (10), при выполнении условия $\varepsilon_2+J+n\omega\approx 0$ вероятность перехода сильно возрастает, т. е. имеет резонансный характер как функция параметров ε_2 и J. Заметим, что возникающие при выполнении точного

резонанса расходимости в (10) устраняются, если учесть процессы релаксации, что фактически приводит к возникновению конечных ширин возбужденных уровней системы. В рамках теории возмущений нетрудно вычислить вероятности переходов между другими состояниями системы. При этом получаются следующие условия для многофотонных резонансов:

$$\begin{aligned} \bar{P}_{1\to 3}: &\quad \varepsilon_1 + J + m\omega \approx 0, \\ \bar{P}_{2\to 4}: &\quad \varepsilon_1 - J + n'\omega \approx 0, \\ \bar{P}_{3\to 4}: &\quad \varepsilon_2 - J + m'\omega \approx 0, \end{aligned} \tag{11}$$

определяемые линейными соотношениями между управляющими параметрами кубитов и константой связи. Аналогичные условия многофотонных резонансов можно получить в высокочастотном приделе $\Delta_i \ll \omega$ в приближении вращающейся волны [Денисенко, 2010].

Заметим, что в первом порядке по Δ_i амплитуда перехода $f_{1\to 4} = 0$ и для ее расчета необходимо учитывать поправки в следующем порядке теории возмущений. Учитывая поправки порядка Δ_i^2: $U = U^{(0)} + U^{(1)} + U^{(2)}$, получаем:

$$f_{1\to 4} = b\left(\Delta_2 d_-^{(1)} + \Delta_1 d_-^{(2)}\right) e^{i\varphi_1(t)} - d_-^{(1)}\left(\frac{i}{2}\Delta_2 a_{12} + d_+^{(2)}\right) e^{i\varphi_2(t)} - d_-^{(2)}\left(\frac{i}{2}\Delta_1 a_{13} + d_+^{(1)}\right) e^{i\varphi_3(t)} +$$

$$+ e^{i\varphi_4(t)}\left(-\frac{\Delta_1\Delta_2}{4}(a_{12}a_{24} + a_{13}a_{34}) + \frac{i}{2}(\Delta_1 d_+^{(2)} a_{24} + \Delta_2 d_+^{(1)} a_{34}) + b(\Delta_1 d_+^{(2)} + \Delta_2 d_+^{(1)})\right), \tag{12}$$

где введен параметр $b = 1/4(\varepsilon_1 + \varepsilon_2)$. Данное выражение будет использовано далее для объяснения дробных резонансов, полученных в численных экспериментах.

КВАЗИЭНЕРГЕТИЧЕСКИЙ ПОДХОД

Теория возмущений позволила получить простые выражения для резонансных переходов. Для более глубокого понимания поведения системы вне рамок теории возмущений и анализа полученных результатов в работе был выполнен численный анализ, основывающийся на квазиэнергетическом представлении. Это преставление дает точные промежуточные состояния системы в переменном поле произвольной амплитуды и позволяет выявить особенности резонансных переходов, обусловленных движением и пересечением квазиуровней при изменении поля.

В силу периодичности во времени гамильтониана (1) имеет место теорема Флоке [Grifoni, Hanngi, 1998], согласно которой существует полный набор решений $\left|\psi_j(t)\right\rangle$, который может быть записан в виде

$$\left|\psi_j(t)\right\rangle = \left|u_j(t)\right\rangle e^{-iQ_j t}, \tag{13}$$

где величины Q_j не зависят от времени и называются квазиэнергиями [Зельдович, 1973; Shirley, 1965; Grifoni, Hanngi, 1998], а векторы состояний зависят периодически от времени $u_j(t) = u_j(t + T)$. Состояния Флоке обладают полнотой

$\sum_j \left|u_j(t)\right\rangle\left\langle u_j(t)\right| = \mathrm{I}$ и свойством ортогональности, поэтому любое решение $\left|\psi(t)\right\rangle$ уравнения Шредингера может быть представлено в виде:

$$\left|\psi(t)\right\rangle = \sum_j C_j \left|u_j(t)\right\rangle \mathrm{e}^{-iQ_j t}, \tag{14}$$

где коэффициенты C_j не зависят от времени и устанавливается начальными условиями при $t = t_0$. Амплитуда перехода системы в момент времени t определяется действием на начальный вектор состояния оператора Флоке:

$$F(t, t_0) = \sum \left|u_k(t)\right\rangle \mathrm{e}^{-iQ_k(t-t_0)} \left\langle u_k(t_0)\right|. \tag{15}$$

Поскольку вероятность перехода $P_{\alpha\to\beta}(t, t_0)$ зависит от квадрата модуля матричного элемента оператора Флоке, то $P_{\alpha\to\beta}(t, t_0)$ определяется соотношением, где α, β — стационарные уровни системы:

$$P_{\alpha\to\beta}(t, t_0) = \sum_{k,j} \mathrm{e}^{-i(Q_k - Q_j)(t-t_0)} M_k(t) M_j^*(t), \tag{16}$$

где $M_j(t) = \left\langle \beta\left|u_j(\tau)\right\rangle\right\langle u_j(\tau + t)\left|\alpha\right\rangle$. Выражение (16) необходимо усреднить по начальным временам t_0 прихода импульса поля на систему и по длительности самого импульса при фиксированной фазе сигнала, тогда получим вероятность $\overline{P}_{\alpha\to\beta}$ [Shirley, 1965; Денисенко и др., 2010], которая будет соответствовать наблюдаемой на опыте населенности при воздействии на систему последовательности импульсов со случайным временем прихода и разбросом длительностей воздействия:

$$\overline{P}_{\alpha\to\beta} = \sum_j \sum_{n,m} \left|\left\langle \beta\left|u_j^{(n-m)}\right\rangle\right|^2 \left|\left\langle u_j^{(n)}\left|\alpha\right\rangle\right|^2, \tag{17}$$

где $\left|u_j^{(n)}\right\rangle = \frac{1}{T}\int_0^T e^{in\omega t}\left|u_j(t)\right\rangle dt$ — фурье-компонента квазинергетической функции.

АНАЛИЗ И ОБСУЖДЕНИЕ РЕЗУЛЬТАТОВ

Как видно из (11), резонансные условия для многофотонных переходов зависят от константы взаимодействия J кубитов, что обуславливает сдвиг пиков резонансов многофотонных переходов. В случае несимметричных кубитов положим $\varepsilon = \varepsilon_2(t) = k\varepsilon_1(t)$ (k — параметр «наклона» линии в плоскости ε_1 и ε_2). При изменении постоянного магнитного поля ε смещение резонансов для переходов 1→2 и 3→4 происходит на величину $\left|J\right|$, а для 1→3 и 2→4 — на величину $\left|J\right|/k$. В зависимости от знака константы связи J (ферромагнитная или антиферромагнитная связь кубитов), происходит смещение резонансных пиков вправо и влево, для переходов 1→3; 3→4 и 1→2; 2→4 соответственно.

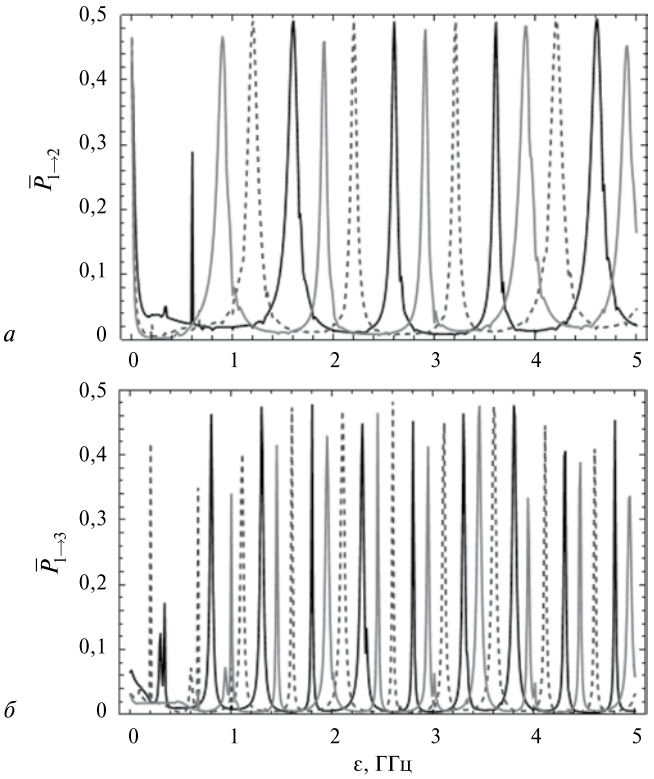


Рис. 2. Зависимость вероятностей переходов: *а* — 1→2; *б* — 1→3 при изменении постоянного магнитного поля $\varepsilon = \varepsilon_2 = 2\varepsilon_1$; $A = A_2 = 2A_1 = 7$ ГГц; $\Delta_1 = \Delta_2 = 0{,}2$ ГГц, где черная сплошная кривая соответствует $J = 0{,}4$ ГГц, черная пунктирная — $J = 0{,}8$ ГГц, серая — $J = 1{,}1$ ГГц

На рис. 2 приведены результаты численного моделирования вероятностей переходов с использованием (17). Как видно, при изменении ε происходят соответствующие смещения резонансных условий, что на языке квазиэнергий объясняется сближением квазиуровней [Денисенко и др., 2010]. Отметим, что для получения точности численного расчета населенностей уровней $\sim 10^{-5}$ при усреднении (17) была учтена 61-я гармоника внешнего поля.

Новые нелинейные эффекты были обнаружены в результате расчета перехода 1→4, когда могут возникать дробные резонансы. Их характерная особенность — возникновение пиков для переходов при нецелых значениях управляющего параметра кубитов ε, а также отсутствие смещения при изменении константы связи J. Данные эффекты были ранее замечены и для случая симметричных кубитов ($\Delta_1 = \Delta_2$, $\varepsilon_1(t) = \varepsilon_2(t)$) [Денисенко и др., 2010]. Более сложную структуру резонансных пиков имеют кубиты с отличающимися управляющими параметрами ($\varepsilon = \varepsilon_2(t) = k\varepsilon_1(t)$). Положение пиков на рис. 3*а* (см. с. 31) можно качественно объяснить одновременным выполнением парных условий резонансов:

$$\begin{cases} \varepsilon_2 + J + n\omega \approx 0, \\ \varepsilon_1 - J + n'\omega \approx 0; \end{cases} \quad \begin{cases} \varepsilon_1 + J + m\omega \approx 0, \\ \varepsilon_2 - J + m'\omega \approx 0. \end{cases} \tag{18}$$

Таким образом, возбуждение происходит с участием промежуточных уровней: второго (левая система в (18)) и третьего (правая система в (18)), что соответствует слагаемому порядка Δ_i^2 в (12). Положения резонансных пиков при фиксированном J определяется выражением $\varepsilon = s\omega/(k+1)$, где $s \equiv n + n' = m + m'$. Видно, что для данного перехода отсутствует сдвиг пиков в зависимости от константы связи. Например, для представленных на рис. 3*а* зависимостей от ε (при $k = 2$) можно рассчитать значения управляющего параметра, при которых произойдет резонансное возбуждение в системе: ε = 0; 0,33; ...; 1,66; 2; 2,33 ГГц. Можно также проследить за поведением резонансных пиков при изменении константы связи. Например, вблизи резонанса ε = 1,66 ГГц пики, соответствующие переходу с участием второго уровня, происходят при $J \approx k\varepsilon + n'\omega$ (J = ...–0,68; 0,32; ... ГГц), а с участием третьего уровня — при $J \approx \varepsilon + m'\omega$

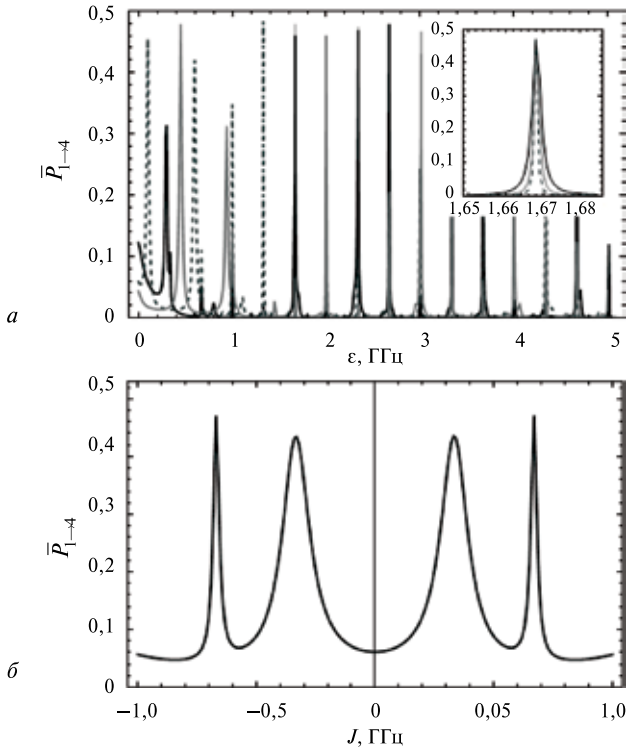


Рис. 3. Вероятность перехода 1→4 при изменении постоянного магнитного поля $\varepsilon = \varepsilon_2 = 2\varepsilon_1$ (*а*) и константы связи J (*б*) вблизи резонанса ε = 1,66 ГГц. Параметры кубитов аналогичны представленным значениям на рис. 2

(J = ...–0,34; 0,66; ... ГГц), что хорошо подтверждается численным расчетом, продемонстрированным на рис. 3*б*. Наличие нелинейных зависимостей и размытия резонансов, которые получены при численном расчете, объясняются довольно большими значениями параметров Δ_i относительно других энергетических характеристик системы. С ростом туннельных параметров Δ_i наблюдается размытие резонансных пиков, что объясняется увеличением вероятностей туннелирования.

Заметим, что при изменении амплитуд переменных сигналов также наблюдается резонансных характер переходов, хотя в данном случае константа взаимодействия не влияет на положение многофотонных переходов, что видно из (10). Это объясняется тем, что осцилляции населенностей определяются функциями

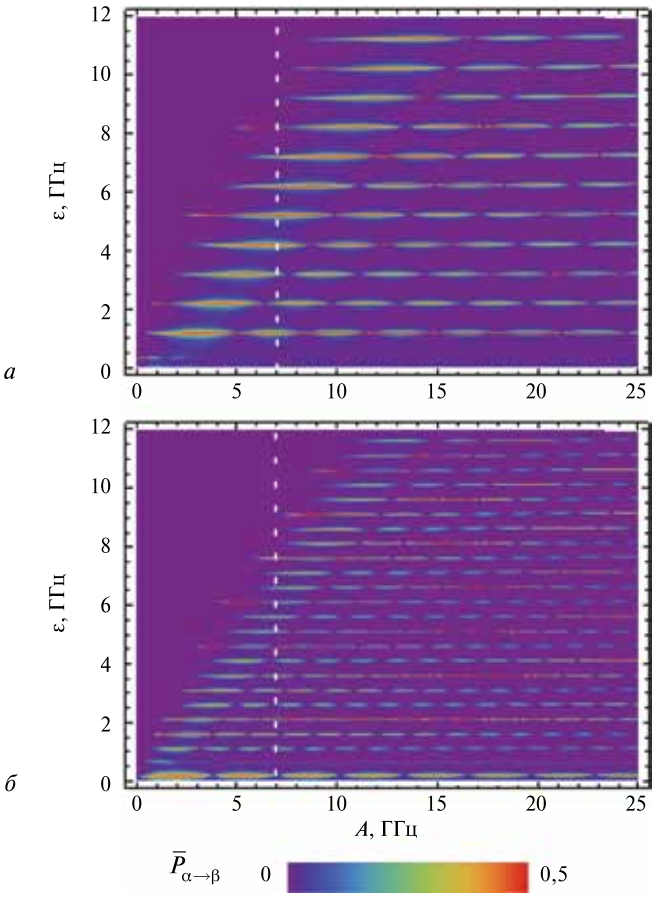


Рис. 4. Вероятности населенностей при переходе 1→2 (*а*) и 1→3 (*б*) при одновременном изменении амплитуды переменного A и постоянного ε магнитных полей. Соотношение между амплитудами полей $\varepsilon = \varepsilon_2 = 2\varepsilon_1$ и $A = A_2 \equiv 2A_1$. Параметры системы: $\Delta_1 = \Delta_2 = 0{,}2$ ГГц и $J = 0{,}8$ ГГц

Бесселя $\sim J_n(A_2/\omega)$ (для переходов 1→2; 3→4) и $\sim J_n(A_1/\omega)$ (для переходов 1→3; 2→4), где амплитуды полей соотносятся как $A \equiv A_2 = kA_1$.

На рис. 4 (см. с. 32)представлены интерференционные картины, обобщающие приведенные выше результаты для одновременного изменения амплитуд постоянного и переменного полей, действующих на систему кубитов. Белой пунктирной линией показано сечение, представленное на рис. 2 для фиксированной амплитуды $A = 7$ ГГц. Отметим, что квантово-когерентные переходы Ландау – Зинера и интерференция Щтюкельберга приводят к формированию областей резонансов, которые выражены яркими областями на интерференционных картинах (см. рис. 4). Сиреневые области свидетельствуют о малом туннелировании, соответственно, о малых вероятностях переходов. Например, видно, что за счет отличия амплитуд внешних полей в два раза ($k = 2$) число резонансных областей при переходе 1→3 также отличается в 2 раза по сравнению с переходом 1→2.

ЗАКЛЮЧЕНИЕ

Как следует из проведенного анализа, взаимодействие кубитов приводит к качественно новым результатам. Во-первых, показано, что константа связи существенно влияет на характер многофотонных переходов. Зависимости резонансных пиков многофотонных переходов от параметров системы исследованы как в рамках теории возмущений, так и путем численных расчетов с использованием квазиэнергетического представления. Квазиэнергетический подход является точным и наиболее адекватным для описания поведения систем в переменных полях, поскольку основывается на теореме Флоке, и справедлив при любой амплитуде переменного поля. Во-вторых, изучено влияние константы связи на условия возникновения дробных резонансов. Подобные резонансы являются типичными для классических нелинейных систем, однако в квантовых системах они изучены недостаточно подробно. Отметим необычный характер возбуждения системы на высший уровень, которое происходит с участием промежуточных уровней системы. Благодаря этому и наблюдаются дробные резонансы, положения которых не меняются при изменении константы связи. Наконец, выполнены расчеты населенностей уровней кубитов и исследованы эффекты формирования интерференционной картины населенностей, обусловленной переходами Ландау–Зинера. Численный расчет позволил наглядно продемонстрировать возникновение интерференционной картины для населенностей переходов и качественно подтвердить полученные аналитические выражения для положений резонансов.

Анализ положений многофотонных резонансов позволяет использовать ландау – зинеровскую спектроскопию для получения параметров системы. Например, по сдвигу положений резонансов в переходе 1→2 можно определять величину константы связи J кубитов, по расстоянию между минимумами и максимумами в осцилляциях Щтюкельберга или наклону интерференционных полос — калибровать мощность генератора. Аналогичные исследования уже проводились для случая одного кубита [Oliver et al., 2005; Berns et al., 2006, 2008].

ЛИТЕРАТУРА

[Денисенко и др., 2010] *Денисенко М. В.*, *Сатанин А. М.*, *Ashhab S.*, *Nori F.* Динамика взаимодействующих кубитов в сильном переменном электромагнитном поле // Физика твердого тела. 2010. Т. 52. № 11. С. 2138.

[Зельдович, 1973] *Зельдович Я. Б.* Рассеяние и излучение квантовой системой в сильной электромагнитной волне // Успехи физ. наук. 1973. Т. 110. С. 139.

[Berns et al., 2006] *Berns D. M.*, *Oliver W. D.*, *Valenzuela S. O.*, *Shytov A. V.*, *Berggren K. K.*, *Levitov L. S.*, *Orlando T. P.* Coherent Quasiclassical Dynamics of a Persistent Current Qubit // Physical Rev. Letters. 2006. V. 97. Art. No. 150502.

[Berns et al., 2008] *Berns D. M.*, *Rudner M. S.*, *Valenzuela S. O.*, *Berggren K. K.*, *Oliver W. D.*, *Levitov L. S.*, *Orlando T. P.* Amplitude spectroscopy of a solid-state artificial atom // Nature. 2008. V. 455. P. 51.

[Grajcar et al., 2005] *Grajcar M.*, *Izmalkov A.*, *van der Ploeg S. H. W.*, *Linzen S.*, *Il'ichev E.*, *Wagner Th.*, *Hübner U.*, *Meyer H.-G.*, *van den Brink A. M.*, *Uchaikin S.*, *Zagoskin A. M.* Direct Josephson coupling between superconducting flux qubits // Physical Rev. B. 2005. V. 72. Art. No. 020503.

[Grifoni, Hanngi, 1998] *Grifoni M.*, *Hanngi P.* Driven quantum tunneling // Physical Rep. 1998. V. 304. P. 229.

[Groot de et al., 2010] *de Groot P. C.*, *Lisenfeld J.*, *Schouten R. N.*, *Ashhab S.*, *Lupascu A.*, *Harmans C. J. P. M.*, *Mooij J. E.* Selective darkening of degenerate transitions demonstrated with two superconducting quantum bits // Nature. 2010. V. 6. P. 783.

[Il'ichev et al., 2010] *Il'ichev E.*, *Shevchenko S. N.*, *van der Ploeg S. H. W.*, *Grajcar M.*, *Temchenko E. A.*, *Omelyanchouk A. N.*, *Meyer H.-G.* Multiphoton excitations and inverse population in a system of two flux qubits // Physical Rev. B. 2010. V. 81. Art. No. 012506.

[Izmalkov et al., 2004] *Izmalkov A.*, *Grajcar M.*, *Il'ichev E.*, *Wagner Th.*, *Meyer H.-G.*, *Smirnov A. Yu.*, *Amin M. H. S.*, *van den Brink A. M.*, *Zagoskin A. M.* Evidence for Entangled States of Two Coupled Flux Qubits // Physical Rev. Letters. 2004. V. 93. Art. No. 037003.

[Mooij et al., 2005] *Mooij J. E.* The road to quantum computing // Science. 2005. V. 307. P. 1210–1211.

[Nakamura et al., 1999] *Nakamura Y.*, *Pashkin Y. A.*, *Tsai J. S.* Coherent control of macroscopic quantum states in a single-Cooper-pair box // Nature. 1999. V. 398. P. 786.

[Oliver et al., 2005] *Oliver W. D.*, *Yu Yang*, *Lee J. C.*, *Berggren K. K.*, *Levitov L. S.*, *Orlando T. P.* Mach-Zehnder Interferometry in a Strongly Driven Superconducting Qubit // Science. 2005. V. 310. P. 1653.

[Shirley, 1965] *Shirley J. H.* Solution of the Schrödinger Equation with a Hamiltonian Periodic in Time // Physical Rev. 1965. V. 138. P. B979.

[van der Wal et al., 2000] *van der Wal C. H.*, *ter Haar A. C. J.*, *Wilhelm F. K.*, *Schouten R. N.*, *Harmans C. J. P. M.*, *Orlando T. P.*, *Lloyd S.*, *Mooij J. E.* (2000) Quantum Superposition of Macroscopic Persistent-Current States // Science. 2000. V. 290. P. 773.

LANDAU-ZENER TRANSITIONS BETWEEN STATES OF THE INTERACTING QUBIT

Nizhny Novgorod State University (*NNSU*)

Landau – Zener transitions in a system of two coupled qubits placed in a strong alternating field are studied. It is shown that if the tunnel splitting of qubits is small

enough, the resonance conditions for multiphoton transitions are dependent on the coupling constant. The results obtained in the framework of perturbation theory are confirmed by numerical calculations by using of the quasi-energy representation. The influence of coupling parameter on the condition for integer and fractional resonances is studied. The level populations of qubits and the effects of the formation of the interference pattern of the populations, due to the Landau – Zener transitions are investigated. We discuss the possibility of experimental observation of the predicted effects.

Keywords: qubits, Landau-Zener transitions, quasienergy, multiphoton transitions.

Denisenko Marina Valeryevna — student, e-mail: mar.denisenko@gmail.com.
Satanin Arkady Mikhailovich — doctor of physical and mathematical sciences, professor, e-mail: sarkady@mail.ru.

УДК 510.52, 519.71

КВАНТОВЫЕ ВЕТВЯЩИЕСЯ ПРОГРАММЫ — НОВАЯ ПАРАДИГМА МОДЕЛИ КВАНТОВЫХ АЛГОРИТМОВ

Ф. М. Аблаев

Казанский (Приволжский) федеральный университет (КФУ)
Государственное бюджетное учреждение «Институт информатики Академии наук Республики Татарстан» (ИИ АН РТ)

Классическая ветвящаяся программа — один из вариантов схемных моделей вычислений (circuit models of computation) [8, 23]. Определена и изучается математическая модель квантовой ветвящейся программы.

В работе рассматриваются классические (детерминированные, вероятностные) и квантовые ветвящиеся программы, определяются их алгебраические и схемные представления. Приводятся сравнительные сложностные характеристики классических и квантовых ветвящихся программ.

Ключевые слова: сложность вычислений, оценки сложности вычислений, классы сложности, вероятностные алгоритмы, квантовые алгоритмы, детерминированные ветвящиеся программы, вероятностные ветвящиеся программы, квантовые ветвящиеся программы.

ВВЕДЕНИЕ

Начало работ в области квантовых вычислений связано с публикацией Р. Феймана в 1982 г. [14], в которой исследовались проблемы использования квантовомеханических эффектов при решении задач, требующих больших вычислительных ресурсов. Ранее, в 1980 г., Ю. Манин в своей книге [3], указал на важность исследований возможности применения квантовомеханических систем для построения вычислительных моделей.

Фундаментальные свойства квантовых алгоритмов и теоретическая возможность построения универсального квантового компьютера были установлены на первых этапах исследований в 80–90-х гг. XX в. В эти годы было введено понятие квантовой модели машины Тьюринга и доказано, что 1) квантовая машина Тьюринга моделируется классической детерминированной машиной Тьюринга [13], 2) существует универсальная квантовая машина Тьюринга [12].

Исследования возможностей квантовых алгоритмов — интенсивно развивающаяся область математической кибернетики. В теоретической квантовой информатике выделяют ряд направлений: квантовая теория информации, квантовая теория кодирования и криптографии, теория квантовых вычислений. По квантовой информатике за последние годы опубликован ряд книг, в частности, переводные [2, 1]. Электронный архив препринтов [15] содержит большое число работ по квантовым вычислениям.

Естественно сравнивать вычислительные возможности квантовых моделей вычислений с классическими вероятностными моделями вычислений. В теории сложности наиболее известны классы сложности *BPP* (Bounded-error Probabilistic

Аблаев Фарид Мансурович — профессор, заведующий кафедрой теоретической кибернетики Казанского федерального университета, доктор физико-математических наук, e-mail: fablayev@gmail.com.

Polynomial-time) и *PP* (Probabilistic Polynomial-time). Класс сложности *BPP* содержит задачи, решаемые полиномиальными по времени классическими вероятностными машинами Тьюринга с большой надежностью, он важен с практической точки зрения. Класс сложности *PP* имеет теоретическое значение и определяется как класс, содержащий задачи, решаемые полиномиальными по времени классическими вероятностными машинами Тьюринга с произвольной (большой и малой) надежностью.

В 1990-е гг. по аналогии с известными классическими вероятностными классами сложности *BBP*, *PP* введены в рассмотрение квантовые классы сложности и установлены общие соотношения между классическими и квантовыми классами [12]. Наибольшее внимание привлекает класс сложности *BQP* (Bounded-error Quantum Polynomial-time), содержащий задачи, решаемые полиномиальными по времени квантовыми машинами Тьюринга с большой надежностью. Класс *BQP* определен по аналогии с классом *BPP*. Установлены включения (несобственные) $BPP \subseteq BQP \subseteq PP$ [12]. Интерес к классу *BQP* вызван тем, что в 1990-х гг. удалось построить эффективные (надежные, полиномиальные по времени) квантовые алгоритмы для ряда задач, для которых эффективные (полиномиальные по времени и надежные) классические алгоритмы не известны. Особой известностью пользуется полиномиальный квантовый алгоритм Шора факторизации чисел [20]. Задача факторизации и близкая к ней задача дискретного логарифма имеют важную, но специфическую область применения — создание систем шифрования с открытым ключом. Отметим, что на сегодняшний день пока не найдены широкие области теории и практики вычислений, где квантовые алгоритмы имеют значительное преимущество по сравнению с классическими. Открытым остается вопрос построения эффективного квантового алгоритма для *NP* полных задач или доказательство того, что это не возможно.

В связи с тем, что технологии построения квантовых вычислителей еще находятся в самом начале своего развития, важными являются исследования моделей вычислений с ограниченными ресурсами («простых» вычислительных моделей). В 1990-х гг. были определены различные квантовые вычислительные модели конечных автоматов, схем из функциональных элементов, ветвящихся программ, коммуникационных протоколов. Для ряда таких моделей со специальными ограничениями обнаружены задачи, которые решаются значительно эффективнее на квантовых моделях по сравнению с классическими.

1. ДЕТЕРМИНИРОВАННАЯ ВЕТВЯЩАЯСЯ ПРОГРАММА

Детерминированная ветвящаяся программа *DBP* (Deterministic Branching Program), с одной стороны, — частный случай контактных схем, с другой стороны, ее можно рассматривать как машинную модель вычислений. Мы приводим определение ветвящейся программы, следуя книге [23].

Определение 1. Программа *DBP* над множеством n переменных $X = \{x_1, \ldots, x_n\}$ — это ориентированный ациклический граф, вершины которого делятся на множество внутренних и множество финальных вершин. Финальные вершины не имеют исходящих ребер и помечены нулем или

единицей соответственно. Каждая внутренняя вершина имеет два исходящих ребра, помеченных 0 ($x = 0$) и 1 ($x = 1$), ей соответствует переменная $x \in X$.

Такое задание *DBP* будем называть «графовым» заданием *DBP*.

Представление булевой функции. Программа *DBP* представляет (вычисляет) булеву функцию $f(X)$ $(f : \{0,1\}^n \to \{0,1\})$ следующим образом. Вычисление значения $f(\sigma)$ для входного набора $\sigma \in \{0,1\}^n$ начинается из выделенной начальной вершины. Для каждой внутренней вершины, помеченной переменной x_j, осуществляется переход из этой вершины либо по 0-ребру, либо по 1-ребру, в соответствии со значением σ_j, которое принимает переменная x_j во входном наборе, до тех пор пока не будет достигнута конечная вершина. Значение функции $f(\sigma)$ для входа σ — это значение достигнутой конечной вершины.

- *Сложность* $S(P)$ программы P — это число ее внутренних вершин.
- *Сложность* $S(f)$ реализации функции f в ветвящихся программах определяется как минимум сложности по всем ветвящимся программам P вычисляющих f:

$$S(f) = \min S(P)$$

Обозначим **P-BP** множество булевых функций, вычисляемых ветвящимися программами полиномиальной (от числа переменных функции) сложности. Известно [23], что

$$\textbf{P-BP} = \textbf{L}/\textbf{poly},$$

где **L/poly** — неоднородный класс сложности, содержащий задачи, решаемые детерминированными машинами Тьюринга с логарифмической памятью, имеющими «полиномиальную подсказку» (вспомогательную ленту с заранее записанной информацией полиномиального объема).

Нижние оценки сложности. Известно, что почти все булевы функции экспоненциально сложны для ветвящихся программ общего вида, т.е. их нельзя представить программами сложности меньше чем $O\left(2^{n-1}/n\right)$.

Задача построения индивидуальной функции, сложно реализуемой в функциональных схемах общего вида (в том числе и ветвящихся программах), является открытой проблемой теории сложности. Наиболее высокая нижняя оценка сложности для явно заданной булевой функции была получена методом Нечипорука. Для ветвящихся программ нижнюю оценку сложности $\Omega\left(n^2/\log^2 n\right)$ реализации индивидуальной функции в *DBP* доказал П. Пудлак [23] в 1984 г.

Идея метода Нечипорука состоит в том, что функция, имеющая много разных подфункций, не может быть представлена ветвящейся программой «маленького размера».

Определение 2. Пусть $S \subseteq X$ — подмножество множества входных переменных X. Все подфункции функции $f(X)$, полученные заменой всех пере-

менных из $X\backslash S$ на некоторые константы, называются подфункциями f на множестве S.

Теорема 1 (*Нечипорука – Пудлака*). Пусть функция $f(X)$ зависит от всех аргументов $x \in X$ существенным образом. Пусть $S_1, \ldots, S_k \subseteq X$ — попарно непересекающиеся подмножества множества X. Пусть s_i — число подфункций f на S_i. Тогда

$$S(f) = \Omega\left(\sum_{i=1}^{k} \frac{\log s_i}{\log\log s_i}\right). \qquad (1)$$

Максимально высокая нижняя оценка сложности реализации индивидуальной функции в ветвящихся программах (без специальных ограничений) получена применением теоремы Нечипорука – Пудлака. Одной из таких индивидуальных функций является функция *ISA* (см. [23]) от n переменных. С применением теоремы 1 доказана нижняя оценка

$$S(ISA) = \Omega\left(\frac{n^2}{\log^2 n}\right). \qquad (2)$$

Забывающие ветвящиеся программы. Ветвящаяся программа называется *уровневой*, если ее вершины могут быть разбиты на уровни 0, 1, ... таким образом, что для каждого i ребра из вершин уровня i ведут только в вершины уровня $(i + 1)$.

Ширина $w(P)$ уровневой ветвящейся программы P — это максимум от числа вершин на уровне, взятый по всем уровням программы P.

Глубина (*длина*) $l(P)$ уровневой ветвящейся программы P — это число уровней программы P.

Уровневая ветвящаяся программа P называется *забывающей*, если на любом уровне P тестируется только одна переменная. Каждая ветвящаяся программа P может быть преобразована (с полиномиальным усложнением) в забывающую ветвящуюся программу P', вычисляющую ту же самую функцию. Далее будем рассматривать забывающие ветвящиеся программы.

2. ВЕТВЯЩИЕСЯ ПРОГРАММЫ С ОГРАНИЧЕНИЯМИ

Ветвящиеся программы — удобная модель для определения различных ограниченных вариантов. Целый ряд таких ограниченных моделей широко используется на практике.

Читающие один раз ветвящиеся программы. Ветвящаяся программа P называется читающей один раз (read-once), если на каждом пути каждая переменная $x \in X$ тестируется не более одного раза.

Для читающих один раз программ в 1980-х гг. были различными авторами доказаны экспоненциальные нижние оценки сложности реализации индивидуальных функций [23].

Упорядоченные ветвящиеся диаграммы решений (*OBDD*). Упорядоченная ветвящаяся диаграмма решений (Ordered Binary Decision Diagram — *OBDD*) — это забывающая, читающая один раз ветвящаяся программа.

Таким образом, *OBDD* — это частный случай ветвящейся программы, все пути которой имеют одинаковую длину. Число вершин *OBDD P* на уровне *i* будем обозначать $width_i(P)$ и называть шириной уровня *i*:

$$width(P) = \max_i width_i(P).$$

Ширина $width(P)$ *OBDD P* также является характеристикой сложности OBDD. Понятно, что сложность *OBDD P* в смысле общего числа вершин *OBDD P* равна $width(P) \cdot n$, где *n* — это число переменных *OBDD*.

Модель *OBDD* и модель конечных автоматов тесно связаны. В ряде случаев в англоязычной литературе *OBDD* называют неоднородными автоматами.

Практическая значимость *OBDD*. Модель *OBDD* была определена в 1959 г. [22]. На практике *OBDD* широко используются для тестирования правильности проектирования СБИС [11, 21], программного и аппаратного оборудования, тестирования различных моделей и в других приложениях (см. [23]). Ключевым моментом такого применения *OBDD* является возможность представления функций в *OBDD* небольшой (полиномиальной) сложности. По двум *OBDD* P_1 и P_2 можно построить алгоритм проверки $P_1 \equiv P_2$?, работающий полиномиальное (от сложности программ P_1 и P_2!) число шагов. Например, схема проверки того, что сконструированная интегральная схема *C* действительно реализует требуемую функцию *f*, следующая. По функции *f* строят *OBDD* P_1, ее представляющую, по интегральной схеме — *OBDD* P_2; проверяют $P_1 \equiv P_2$?

Однако за удобство (применения *OBDD*) «платится высокая цена» (но похоже это происходит не только в Computer Science): целый ряд важных для практики функций сложно реализуемы в *OBDD*. Например, функция умножения $MULT(X,Y) = X \cdot Y$, где целые *X*, *Y* представлены в двоичном виде длины *n* каждое, экспоненциально (от $2n$) сложно реализуема в *OBDD*. Поэтому важным для теории и практики является расширение модели *OBDD* и выяснение вопроса: можно ли в чуть более общей модели сохранить удобство *OBDD* и при этом расширить класс функций, представимых с полиномиальной сложностью? Исследуемыми обобщениями классических *OBDD* являются определения вероятностных, а затем и квантовых *OBDD*.

3. ВЕРОЯТНОСТНАЯ ВЕТВЯЩАЯСЯ ПРОГРАММА

Вероятностная ветвящаяся программа PBP (Probabilistic Branching Program) — это программа, в которой каждая внутренняя вершина имеет выходную степень $\geqslant 2$. При этом из каждой внутренней вершины выходит два типа ребер — помеченные 0 и 1. Каждому ребру *e* приписана вероятность $p(e)$ $(0 \leqslant p(e) \leqslant 1)$. Для каждой вершины сумма вероятностей исходящих из нее всех ребер, помеченных 0 (1), равна 1. Вычисление на входном наборе $\sigma \in \{0,1\}^n$ осуществляется

следующим образом. На каждом шаге, начиная с выделенной начальной вершины, *PBP* считывает значение переменной, приписанной вершине, и в зависимости от него переходит в следующие вершины либо по 0-ребрам, либо по 1-ребрам с вероятностями, приписанными соответствующим ребрам. Вероятность $p_{acc}(\sigma)$ принятия *PBP* входа σ — это вероятность того, что вычисление на входе σ приведет в конечную принимающую вершину (вершину, помеченную 1).

Определение 3. Пусть $\varepsilon > 0$. *PBP P* вычисляет функцию *f* с надежностью $(1/2+\varepsilon)$, если для $\sigma \in f^{-1}(1)$ $p_{acc}(\sigma) \geqslant 1/2+\varepsilon$ и для $\sigma' \in f^{-1}(0)$ $p_{acc}(\sigma') \leqslant 1/2-\varepsilon$. При этом предполагается, что *P* вычисляет функцию *f* с ограниченной ошибкой.

Известно, что для ветвящихся программ общего вида (без ограничения на число считываний переменных) вероятностные модели не имеют существенных преимуществ в сложности реализации функций. А именно, по произвольной *PBP*, вычисляющей функцию с ограниченной ошибкой, можно построить детерминированную ветвящуюся программу (с полиномиальным увеличением сложности), вычисляющую ту же функцию. Отметим, что в основе построения *DBP* по исходной *PBP* существенно используется прием полиномиального «повторения» *PBP*, что приводит к новой *PBP*, вычисляющей ту же булеву функцию с «очень небольшой» ошибкой.

Обозначим **BPP-BP** множество булевых функций, вычисляемых *PBP* с ограниченной ошибкой, полиномиальной сложности. Таким образом справедливо равенство

$$\textbf{BPP-BP} = \textbf{P-BP}.$$

Это равенство нарушается в моделях ветвящихся программ с ограничениями на число считываний переменных. В частности, для модели *OBDD* были построены функции [4], которые представимы в вероятностных *OBDD* с полиномиальной сложностью, а для их представления в детерминированных *OBDD* требуется экспоненциальная сложность.

4. ЛИНЕЙНАЯ ВЕТВЯЩАЯСЯ ПРОГРАММА

Определяемая в данном разделе линейная ветвящаяся программа [6] является обобщением понятия забывающей детерминированной программы и определяемой далее квантовой ветвящейся программы.

Линейная ветвящаяся программа *LBP* (Linear Branching Program). *P* над множеством переменных $X = \{x_1, \ldots, x_n\}$ и над *d*-мерным векторным пространством $\mathbf{V}^d$ есть тройка

$$P = \langle T, |\mu_0\rangle, F\rangle.$$

• Множество $B^d = \{|1\rangle = (1,0,\ldots,0), \ldots, |d\rangle = (0,\ldots,0,1)\}$ базисных векторов будем называть базисными состояниями *LBP*.

• Векторы $|\mu\rangle = \sum_{i=1}^{d} z_i |i\rangle$ пространства $\mathbf{V}^d$ — состояния.

• Преобразования состояний P определяются последовательностью $T = (T_1,\ldots,T_\ell)$ (длины ℓ) инструкций. Каждая инструкция T_i — это тройка $T_i = \{j_i, M_{j_i}(0), M_{j_i}(1)\}$, где j_i определяет переменную x_{j_i}, считываемую на шаге i, $M_{j_i}(0)$ и $M_{j_i}(1)$ — это $d \times d$ -матрицы — линейные преобразования векторного пространства $\mathbf{V}^d$;

• $|\mu_0\rangle$ — начальное состояние программы $\mathcal{P}$;

• $F \subseteq B^d$ — подмножество базисных состояний, элементы которого будем называть принимающими состояниями, а элементы множества $\overline{F} = B^d \setminus F$ — отвергающими состояниями. Через Accept и Reject будем обозначать множества индексов принимающих и отвергающих состояний соответственно: $\text{Accept} = \{i : |i\rangle \in F\}$ и $\text{Reject} = \{i : |i\rangle \in \overline{F}\}$.

Вычисление программы P на входе $\sigma = \sigma_1 \ldots \sigma_n \in \{0,1\}^n$ определяется следующим образом:

1) вычисление P начинается из начального состояния $|\mu_0\rangle$;

2) на i-м шаге вычисления P применяется инструкция T_i: если $x_{j_i} = \sigma_{j_i}$, то к текущему состоянию μ применяется преобразование $M_{j_i}(\sigma_{j_i})$, и программа P переходит в состояние $|\mu'\rangle = M_{j_i}(\sigma_{j_i})|\mu\rangle$ (состояния программы представляются в виде векторов-столбцов);

3) финальным (состоянием после последнего шага ℓ) будет состояние

$$|\mu(\sigma)\rangle = \prod_{i=\ell}^{1} M_{j_i}(\sigma_{j_i})|\mu_0\rangle .$$

Определенная линейная программа является забывающей и имеет $\ell + 1$ уровень. Нумерация уровней начинается с нуля, последний (финальный) уровень имеет номер ℓ.

Шириной $w(P)$ *LBP* P будем называть размерность d пространства $\mathbf{V}^d$ состояний P, а число ℓ — длиной $l(P)$ программы P.

Забывающие детерминированные и вероятностные ветвящиеся программы (*DBP* и *PBP*) представляются в виде линейных ветвящихся программ следующим образом.

Линейное представление забывающей *DBP*. Забывающая *DBP* — это *LBP* над векторным пространством $\mathbb{F}^d$, где $\mathbb{F}$ — подходящее конечное поле. Множеством состояний такой LBP является множество B^d базисных состояний. На каждом шаге i работы программы матрицы $M_{j_i}(0)$, $M_{j_i}(1)$ задают преобразования $M_{j_i}(0): B^d \to B^d$ и $M_{j_i}(0): B^d \to B^d$ множества B^d при считывании значений 0 и 1 соответствующей переменной x_{j_i}. Входной набор σ принимается, если

$\left|\mu(\sigma)\right\rangle \in F$ или (другая эквивалентная запись), если для $\left|\mu(\sigma)\right\rangle = (\mu_1, \ldots, \mu_d)$ выполняется $\sum_{i \in \text{Accept}} \mu_i = 1$.

Линейное представление забывающей *PBP*. Линейное представление забывающей *DBP* естественным образом обобщается до линейного представления забывающей *PBP*. Забывающая *PBP* — это *LBP* над векторным пространством $\mathbb{R}^d$, где $\mathbb{R}$ — поле вещественных чисел. Множеством состояний такой *LBP* является множество стохастических векторов пространства $\mathbb{R}^d$. На каждом шаге *i* работы программы стохастические матрицы $M_{j_i}(0)$, $M_{j_i}(1)$ задают преобразования $M_{j_i}(0): \mathbb{R}^d \to \mathbb{R}^d$ и $M_{j_i}(0): \mathbb{R}^d \to \mathbb{R}^d$ множества $\mathbb{R}^d$ при считывании значений 0 и 1 соответствующей переменной x_{j_i}.

Вероятность Pr(σ) принятия входного набора σ определяется условием: для состояния $\left|\mu(\sigma)\right\rangle = (\mu_1, \ldots, \mu_d)$ полагают $\Pr(\sigma) = \sum_{i \in \text{Accept}} \mu_i$.

5. КВАНТОВАЯ ВЕТВЯЩАЯСЯ ПРОГРАММА

Квантовая ветвящаяся программа *QBP* (Quantum Branching Program) — это *LBP* над комплекснозначным гильбертовым *d*-мерным пространством H^d. Состояниями *QBP Q* являются векторы $\left|\psi\right\rangle = \sum_{i=1}^{d} z_i \left|i\right\rangle$ с единичной нормой $\left\|\left|\psi\right\rangle\right\| = \sqrt{\sum_{i=1}^{d} \left|z_i\right|^2} = 1$. Они называются чистыми состояниями или просто состояниями QBP. Преобразования QBP задаются комплекснозначными унитарными $d \times d$ матрицами.

Если $\left|\psi(\sigma)\right\rangle = \sum_{i=1}^{d} z_i \left|i\right\rangle$ — финальное состояние *QBP* после считывания входного набора σ, то вероятность $p_{acc}(\sigma)$ принятия этого набора σ программой *P* определяется как

$$p_{acc}(\sigma) = \sum_{i \in \text{Accept}} \left|z_i\right|^2.$$

Шириной квантовой ветвящейся программы *Q* называется размерность *d* пространства H^d, а длиной — число *l* инструкций в последовательности *T*.

Определение 4. Пусть $\varepsilon > 0$. При этом *QBP Q* вычисляет функцию *f* с надежностью $1/2 + \varepsilon$, если для $\sigma \in f^{-1}(1)$ выполняется неравенство $p_{acc}(\sigma) \geqslant 1/2 + \varepsilon$, а для $\sigma' \in f^{-1}(0)$ — неравенство $p_{acc}(\sigma') \leqslant 1/2 - \varepsilon$.

В частности, считается, что квантовая ветвящаяся программа *Q вычисляет булеву функцию f с односторонней ошибкой*, если существует $\varepsilon \in (0,1)$ такое, что

для любого $\sigma \in f^{-1}(1)$ вероятность принятия этого набора программой Q равна 1, а для любого $\sigma \in f^{-1}(0)$ вероятность принятия не превышает ε.

Определенная модель квантовой ветвящейся программы является квантовой вычислительной моделью с одним измерением ("measure-once"). Такие квантовые модели вычислений рассматриваются, в частности, для конечных квантовых автоматов [16, 10], в которых состояние системы изменяется унитарным образом в процессе считывания входного слова, а извлечение информации (финальное измерение) о состоянии автомата производится один раз после прочтения всего слова.

Квантовые один раз читающие ветвящиеся программы

Определение 5. Квантовая ветвящаяся программа Q называется *QOBDD* или один раз читающей квантовой ветвящейся программой, если каждая переменная $x \in \{x_1, \ldots, x_n\}$ появляется в последовательности инструкций T программы Q не более одного раза.

Обозначим через **P-QOBDD** класс функций, представимых квантовыми *OBDD* полиномиальной ширины.

Ветвящиеся программы ограниченной ширины. Определим классы сложности для ветвящихся программ ограниченной ширины.

• $\mathbf{P\text{-}BP}_k$ класс функций, представимых ветвящимися программами ширины k и полиномиальной длины.

Положим $\mathbf{BWP\text{-}BP} = \cup_k \mathbf{P\text{-}BP}_k$.

• $\mathbf{BPP\text{-}BP}_k$ — подкласс класса **BPP-BP**, который состоит из функций, представимых вероятностными ветвящимися программами полиномиальной сложности и ширины k с ограниченной ошибкой.

Обозначим $\mathbf{BWBPP\text{-}BP} = \cup_k \mathbf{BPP\text{-}BP}_k$.

• **BQP-BP** — класс функций, которые ε-принимаются квантовыми ветвящимися программами полиномиальной сложности для некоторой константы $\varepsilon \in (0,1/2)$.

• $\mathbf{BQP\text{-}BP}_k$ — подкласс класса **BQP-BP**, состоящий из всех функций, которые ε-принимаются квантовыми программами полиномиальной сложности и ширины k для некоторой константы $\varepsilon \in (0,1/2)$.

Обозначим $\mathbf{BWBQP\text{-}BP} = \cup_k \mathbf{BQP\text{-}BP}_k$.

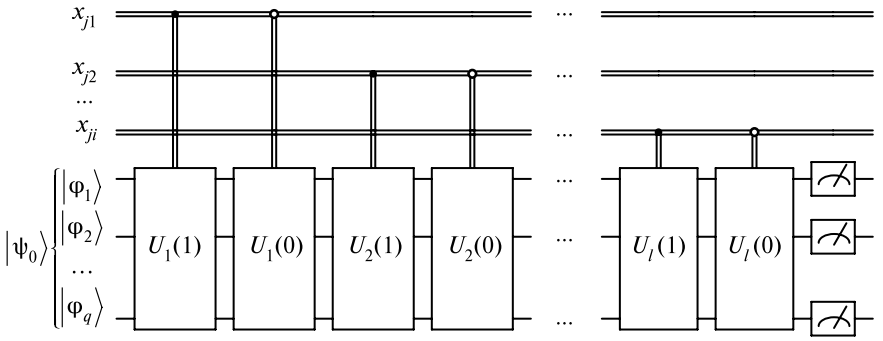
• **EQP-BP** — класс всех функций, которые вычислимы некоторыми квантовыми ветвящимися программами полиномиальной сложности без ошибки, т.е. с нулевой ошибкой.

• $\mathbf{EQP\text{-}BP}_k$ — подкласс класса **EQ3-BP**, который состоит из всех функций, вычисляющихся квантовыми ветвящимися программами полиномиальной сложности и ширины k без ошибки.

Обозначим $\mathbf{BWEQP\text{-}BP} = \cup_k \mathbf{EQP\text{-}BP}_k$.

5.1. Схемное представление

Предлагаемые нами методы построения эффективных квантовых алгоритмов в модели квантовых ветвящихся программ опираются на их схемное представление [7]. А именно, квантовая ветвящаяся программа рассматривается как квантовая схема, дополненная возможностью считывать классические биты в качестве контролирующих для унитарных операторов. Таким образом, любая квантовая схема есть квантовая ветвящаяся программа, которая существенно не зависит от своих классических входов.



Здесь $x_{j_1},\ldots,x_{j_l}$ — последовательность переменных (необязательно различных), обозначающих классические управляющие сигналы. Согласно принятой в литературе по квантовым схемам нотации, классическое управление обозначается на схеме двойными проводами, а квантовое — одиночными.

Заметим, что для квантовой ветвящейся программы в схемном представлении явным образом проявляется еще одна мера сложности — число кубит q, необходимое для физической реализации соответствующей квантовой системы с классическим управлением. Согласно постулатам квантовой механики для реализации квантовой ветвящейся программы ширины d (системы с d состояниями) потребуется как минимум $\log d$ кубит.

Определение 6. Назовем квантовую ветвящуюся программу q-кубитной, если она может быть реализована в виде классически управляемой квантовой системы, построенной на q кубитах.

5.2. Эффективные квантовые ветвящиеся программы

По определению квантовых ветвящихся программ унитарные операторы, используемые в инструкциях, могут быть произвольными, в том числе решающими NP-трудные задачи. Поэтому разумным ограничением является полиномиальная по времени конструируемость этих преобразований, т. е. они должны быть представимы в виде произведения не более чем полиномиального числа «элементарных» унитарных операторов из некоторого конечного базиса (например, из «стандартного» базиса $\{H, S, \mathrm{CNOT}, \pi/8\}$, см. [17]).

В силу вышесказанного эффективными будем называть те квантовые ветвящиеся программы, ширина и длина которых не превосходит $n^{O(1)}$. Так как при ограничении ширины $n^{O(1)}$ число кубит, задействованных в вычислениях, составляет $q = O(\log n)$, каждое унитарное преобразование такой программы можно представить в виде произведения $O(q^2 4^q) = n^{O(1)}$ (т. е. полиномиального числа) «элементарных» операторов. Поэтому полиномиальное ограничение на ширину и длину программы влечет и полиномиальную сложность используемых унитарных преобразований.

Аналогично классическому случаю, известно, что для почти всех булевых функций сложность квантовых ветвящихся программ экспоненциальна. В последнее десятилетие некоторыми авторами получены результаты, показывающие, что для целого ряда функций квантовые ветвящиеся программы могут быть более эффективными, чем классические детерминированные и вероятностные ветвящиеся.

В 1988 г. Д. М. Баррингтон [9] показал, что $\mathbf{P\text{-}BP}_{\text{const}} = \mathbf{P\text{-}BP}_5 = \mathbf{NC}^1$. Класс сложности $\mathbf{NC}^1$ состоит из всех булевых функций, вычисляемых классическими схемами из функциональных элементов полимиальной сложности и логарифмической глубины.

Ф. М. Аблаев, К. Мур и К. Полетт [6] показали, что

$$\mathbf{EQP\text{-}BP}_2 = \mathbf{NC}^1. \tag{3}$$

Содержательно равенство (3) показывает, что полиномиальные по времени вычисления, построенные на одном кубите, по своим возможностям весьма мощные (совпадают с классом функций, вычислимых схемами полиномиальной сложности и логарифмической глубины). Этот факт остается одним из результатов в области квантовых вычислений, в котором устанавливаются совпадения классов сложности, определяемых классическими и квантовыми моделями. Как видно из приведенной далее теоремы, все классы сложности, определенные ранее, равны между собой.

Теорема 2 ([6])

$$\mathbf{EQP\text{-}BP}_2 = \mathbf{BQP\text{-}BP}_2 = \mathbf{EQP\text{-}BP} = \mathbf{BQP\text{-}BP} = \mathbf{BPP\text{-}BP} = \mathbf{BWP\text{-}BP} = \mathbf{NC}^1.$$

Ранее мы уже привели формулу (2), дающую максимально высокую (известную на сегодняшний день) нижнюю оценку сложности реализации индивидуальной функции для ветвящихся программ без ограничений. Ситуация на сегодняшний день для общей модели ветвящихся программ такая же, как и для других общих моделей (схемных и машинных) вычислений: нет методов, позволяющих описывать в явном виде функции, сложно реализуемые (не полиномиально) в данной модели.

Для модели *OBDD*, которая является значительно более ограниченной моделью ветвящихся программ, ситуация иная. Определим классы сложности, с которыми будем иметь дело:

• **BQP-OBDD** — класс всех функций, вычисляемых с ограниченной ошибкой квантовыми OBDD полиномиальной сложности;

• **EQP-OBDD** — класс всех функций, которые вычисляются некоторыми квантовыми *OBDD* полиномиальной сложности без ошибки, т. е. с нулевой ошибкой;

• **RQP-OBDD** — класс всех функций, которые вычисляются с односторонней ошибкой некоторыми квантовыми *OBDD* полиномиальной сложности;

• класс квантовых *OBDD* с нулевой ошибкой, определяемый как

$$\mathbf{ZQP\text{-}OBDD} = \mathbf{RQP\text{-}OBDD} \cap \mathbf{coRQP\text{-}OBDD};$$

• класс всех обратимых программ из **P-OBDD**, называемый **Rev-OBDD**.

Последний детерминированный класс сложности, определенный ранее, является естественным детерминированным подклассом **BQP-OBDD**, точно так же, как **P-OBDD** — это естественный подкласс **BPP-OBDD**.

Следующий результат был получен М. Зауэрхоффом и Д. Зиилингом [18].

Теорема 3 ([18])

$$\mathbf{Rev\text{-}OBDD} = \mathbf{EQP\text{-}OBDD} = \mathbf{ZQP\text{-}OBDD}.$$

Однако оставшиеся классы сложности для OBDD не схлопываются аналогично с соответствующими классами для ветвящихся программ ограниченной ширины.

В работе [5] была предложена функция, для которой между *с*табильными вероятностными и стабильными квантовыми *OBDD* обнаружилась экспоненциальная разница в сложности. *Стабильность* — довольно сильное ограничение для *OBDD*.

Определение 7. Рассмотрим *OBDD P*, где $T = \left(j_i, M_i(0), M_i(1)\right)_{i=1}^{n}$ — подходящая последовательность преобразований и n — это длина входа. Тогда P называется стабильной, если $M_i(0) = M_j(0)$ и $M_i(1) = M_j(1)$ для всех $i, j \in \{1, \ldots, n\}$. Другими словами, преобразования не зависят от уровня P.

Функция MOD_p определяется следующим образом.

Определение 8 ([5]). $\mathrm{MOD}_p(\sigma) = 1$, тогда и только тогда, когда число единиц во входном наборе σ кратно p.

Теорема 4 ([5]). Функция MOD_p представима стабильной, один раз читающей, ширины $O(\log p)$ квантовой ветвящейся программой с односторонней ошибкой $\varepsilon > 0$.

Теорема 5 ([5]). Любая стабильная вероятностная *OBDD*, вычисляющая MOD_p, имеет ширину по крайней мере p.

С этим результатом родилась надежда, что удастся строго доказать превосходство вычислительной мощи квантовых вычислителей над вероятностными

для *OBDD*-модели. Однако в 2004 г. М. Зауэрхофф и Д. Зиилинг показали, что квантовые и классические *OBDD* несравнимы [18]. Для доказательства этого результата они использовали следующие функции.

Функция «*Проверка перестановочности матрицы*» — функция PERM_n определена на булевых матрицах размера $n \times n$.

Определение 9. На входном наборе $\sigma \in \{0,1\}^{n^2}$ $\mathrm{PERM}_n(\sigma) = 1$, тогда и только тогда, когда σ соответствует перестановочной матрице, т. е. матрице, в которой каждая строка и каждый столбец содержат ровно одну единицу.

Известно, что для представления функции PERM_n в один раз читающих ветвящихся программах требуется экспоненциальная сложность. Однако известен следующий результат [23].

Теорема 6. Функция PERM_n может быть вычислена с односторонней ошибкой $\varepsilon(n \epsilon(n)$ вероятностной один раз читающей упорядоченной ветвящейся программой размера

$$O\left(\varepsilon(n)^{-2} n^5 \log^3 n\right).$$

Кроме того, М. Зауэрхофф и Д. Зиилинг определили элегантную функцию *соседствующих единиц*.

Определение 10 ([18]). Функция соседствующих единиц NO_n определена на булевских переменных $x_1, \ldots, x_n$. Она принимает значение 1, тогда и только тогда, когда имеются две соседствующие переменные, значения которых на входе равны 1, т. е., существует индекс $i \in \{1, \ldots, n-1\}$ такой, что $x_i = x_{i+1} = 1$.

Функция NO_n вычислима детерминированной *OBDD* размера $O(n)$. Несравнимость детерминированных и квантовых *OBDD* показывают следующие результаты.

Теорема 7 ([18]). Существует квантовая *OBDD*, вычисляющая отрицание \neg PERM функции PERM с односторонней ошибкой $1/n$. Причем, размер такой программы будет $O(n^6 \log n)$. Таким образом, верно следующее:

$$\textbf{BQP-OBDD} \not\subseteq \textbf{P-OBDD}.$$

Теорема 8 ([18]). Размер любой квантовой OBDD G, вычисляющей NO_n с ограниченной ошибкой не меньше $2^{\Omega(n)}$. Таким образом, верно следующее:

$$\textbf{P-OBDD} \not\subseteq \textbf{BQP-OBDD}$$

ЛИТЕРАТУРА

[1] *Валиев К.А.*, *Кокин А.А.* Квантовые компьютеры: надежды и реальность. Ижевск: НИЦ «Регулярная и хаотическая динамика», 2001. 352 с.

[2] *Китаев А.*, *Шень А.*, *Вялый М.* Классические и квантовые вычисления. М.: МЦНМО, ЧеРО, 1999. 192 с.

[3] *Манин Ю.И.* Вычислимое и невычислимое. М.: Сов. радио, 1980.

[4] *Ablayev F.*, *Karpinski M.* On the power of randomized branching programs // Proc. Intern. Colloquim on Automata, Languages and Programming (ICALP'1996): Lecture Notes in Computer Science. Berlin, Heidelberg, N. Y.: Springer-Verlag, 1996. No. 1099. P. 438–356.

[5] *Ablayev F.*, *Gainutdinova A.*, *Karpinski M.* On computational power of quantum branching programs // Proc. 13th Intern. Symp. Fundamentals of Computation Theory (FCT'2001): Lecture Notes in Computer Science. Berlin, Heidelberg, N. Y.: Springer-Verlag, 2001. No. 2138. P. 59–70.

[6] *Ablayev F.*, *Moore C.*, *Pollett C.* Quantum and Stochastic Branching Programs of Bounded Width // Proc. of the Intern. Colloquium on Automata, Languages and Programming (ICALP'2002): Lecture Notes in Computer Science. Berlin: Springer-Verlag, 2002. P. 343–354.

[7] *Ablayev F.*, *Vasiliev A.* On computational power of quantum read-once branching programs // Electronic Proc. Theoretical Computer Science. 2011. V. 52. P. 1–12. [Электрон. текст]. Режим доступа: http://arxiv.org/abs/1103.2809v1.

[8] *Arora S.*, *Barak B.* Computational Complexity: A Modern Approach. Cambridge University Press, 2009.

[9] *Barrington D.M.* Finite monoids and the fine structure of NC^1 / D.M. Barrington, D. Thérien // J. ACM. 1988. V. 35. P. 941–952 (см. также русский перевод: *Баррингтон Д.* Ветвящиеся программы ограниченной ширины, имеющие полиномальную сложность, распознают в точности языки из NC^1 // Кибернетический сб. 1991. Вып. 28. С. 94–113.

[10] *Brodsky A.*, *Pippenger N.* Characterizations of 1-way quantum finite automata, quant-ph/9903014, 1999.

[11] *Bryant R.* Symbolic boolean manipulation with ordered binary decision diagrams // ACM Computing Surveys. 1992. V. 24. No. 3. P. 293–318.

[12] *Bernstein B.*, *Vazirany U.* Quantum complexity theory // Soc. for Industrial and Applied Mathematics (SIAM) J. on Computing. 1997. V. 26. No. 5. P. 1411–1473.

[13] *Deutsch D.* Quantum theory, the Church-Turing principle and the universal quantum computer // Proc. Royal Soc. of London. A 400. 1985. P. 97–117.

[14] *Feynman R.* Simulation physics with computers // Intern. J. Theoretical Physics. 1982. V. 21. No. 467.

[15] http://xxx.lanl.gov/archive/quant-ph (см. также российское зеркало: http://xxx.itep.ru).

[16] *Moore C.*, *Crutchfield J.* Quantum automata and quantum grammars // Theoretical Computer Science. 2000. 237. P. 275–306.

[17] *Nielsen M.A.*, *Chuang I.L.* Quantum Computation and Quantum Information. Cambridge University Press, 2000.

[18] *Sauerhoff M.*, *Sieling D.* Quantum branching programs and space-bounded nonuniform quantum complexity. 2004. [Электрон. текст]. Режим доступа: http://xxx.lanl.gov/archive/quant-ph.ph/0403164.

[19] *Sauerhoff M.* Quantum vs. Classical Read-Once Branching Programs // arXiv:quant-ph/0504198 v1. 2005.

[20] *Shor P.* Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer // Soc. for Industrial and Applied Mathematics (SIAM) J. on Computing. 1997. V. 26. No. 5. P. 1484–1509.

[21] *Wegener I.* Efficient data structures for boolean functions // Discrete Mathematics. 1994. V. 136. P. 347–372.
[22] *Lee C. Y.* Representation of switching circuits by binary-decision programs // Bell Systems Technical J. 1959. V. 39. P. 985–999.
[23] *Wegener I.* Branching Programs and Binary Decision Diagrams. SIAM Monographs on Discrete Mathematics and Applications, 2000.

QUANTUM BRANCHING PROGRAMS — NEW PARADIGM OF MODELS OF QUANTUM ALGORITHMS

F. M. Ablayev

Kazan Federal University and Institute for Informatics of Tatartan Academy of Science

Classical branching programs is a known model of circuit computations. In the paper we present a mathematical model of quantum branching programs. We consider classical (deterministic, probabilistic) and quantum branching programs. We detemine their algebraic and circuit presentations. We present comparative complexity characterictics of classical and quantum branching programs.

Keywords: quantum computations, complexity, complexity classes, branching programs, quantum circuits.

Ablayev Farid — professor, doctor of physical and mathematical sciences, e-mail: fablayev@gmail.com.

УДК 539.101

ОСНОВНЫЕ ПОДХОДЫ К ПОСТРОЕНИЮ ГРИД-ИНФРАСТРУКТУРЫ НАЦИОНАЛЬНОЙ НАНОТЕХНОЛОГИЧЕСКОЙ СЕТИ

А. П. Крюков, А. П. Демичев, В. А. Ильин, Л. В. Шамардин

Научно-исследовательский институт им. Д. В. Скобельцына Московского государственного университета имени М. В. Ломоносова (*НИИЯФ МГУ*)

Основная цель грид-инфраструктуры Национальной нанотехнологической сети (ГридННС) — это предоставление ученым, инженерам, аспирантам и студентам унифицированного безопасного удаленного доступа к суперкомпьютерным ресурсам. Впервые в мире в ГридННС используется архитектурный стиль REST для реализации грид-сервисов. В инфраструктуре ГридННС разнесены потоки данных и управляющие потоки. Управление данными в ГридННС осуществляется по принципу передачи данных непосредственно с сервера хранения данных на сайт, где будет выполняться задача, а результатов — на сервер хранения данных. Проведено сравнение разработанного сервиса распределения нагрузки Pilot и аналогичного сервиса программного обеспечения (ПО) gLite. Показана его эффективность, устойчивость и надежность.

Работа выполнена при частичной финансовой поддержке РФФИ (проект № 10-07-00332) и Министерства образования и науки РФ (контракты № 01.647.11.2004, 02.740.11.0388, 07.514.11.4022 и НШ № 4142.2010.2).

Ключевые слова: высокопроизводительные вычисления, распределенные вычисления, грид, ГридННС, REST, RESTful-веб-сервисы.

ВВЕДЕНИЕ

Концепция грид была предложена в конце 90-х гг. прошлого века К. Кессельманом и Я. Фостером [Foster, Kesselman, 1998]. Ими же был разработан первый инструментарий для построения грид-систем — Globus Toolkit ver. 2.0 (2001). С тех пор в мире было выполнено много исследовательских проектов в области грид, но только немногие достигли производственного уровня. За это время грид прошли значительный путь развития. Так, первый успешный инструментарий Globus Toolkit ver. 2.0 (GT2) использовал собственные протоколы взаимодействия грид-сервисов. На основе опыта использования GT2 была сформулирована открытая архитектура грид-сервисов OGSA (2006), первая версия которой была принята в 2005 г. В качестве реализации принципов OGSA была предложена спецификация OGSI (2003), которая в дальнейшем была заменена на WSRF (2006). В подходе WSRF объединяются веб-сервис и ресурс, с которым взаимодействует этот веб-сервис. Канонической реализацией WSRF является Globus Toolkit ver. 4.0 (GT4).

Крюков Александр Павлович — ведущий научный сотрудник, кандидат физико-математических наук, e-mail: kryukov@theory.sinp.msu.ru.

Демичев Андрей Павлович — старший научный сотрудник, кандидат физико-математических наук, e-mail: demichev@theory.sinp.msu.ru.

Ильин Вячеслав Анатольевич — заведующий лабораторией, доктор физико-математических наук, e-mail: ilyin@sinp.msu.ru.

Шамардин Лев Витальевич — младший научный сотрудник, e-mail: shamardin@theory.sinp.msu.ru.

Основная цель стандартов WSRF — создать универсальные механизмы и протоколы, которые могут быть использованы практически для любых грид. Предполагалось, что, используя WSRF, можно будет обеспечить совместимость грид-сервисов не только независимо от их реализации, но и не зная деталей их реализации. Однако стандарт WSRF оказался исключительно сложным и, как следствие, трудным в реализации. Это привело к тому, что даже каноническая его реализация в виде GT4 не является полной и содержит ряд ошибок. Все это поставило под угрозу достижение основной цели, а именно создание универсального, динамически изменяющегося грид.

В качестве альтернативного подхода в последнее время рассматривается вариант построения грид-сервисов с использованием архитектурного стиля REST [Richardson, Ruby, 2007].

В данной работе описываются основные принципы построения грид-инфраструктуры ГридННС (Грид для Национальной нанотехнологической сети. 2008. [Электрон. ресурс]. Режим доступа: http://ngrid.ru/ngrid), в которой был применен ряд новых решений, в том числе использование архитектурного стиля REST для разработки грид-сервисов.

1. СТРУКТУРА ГридННС

Структура ГридННС достаточно традиционна для современных грид. Можно выделить три слоя (рис. 1): слой пользовательских интерфейсов, слой грид-шлюзов к ресурсам и слой общих инфраструктурных сервисов.

Слой *пользовательских интерфейсов* (ПИ) включает следующие основные виды интерфейсов:

- интерфейс командной строки (ИКС);
- веб-интерфейс ГридННС (ВИГ);
- прикладные ПИ.

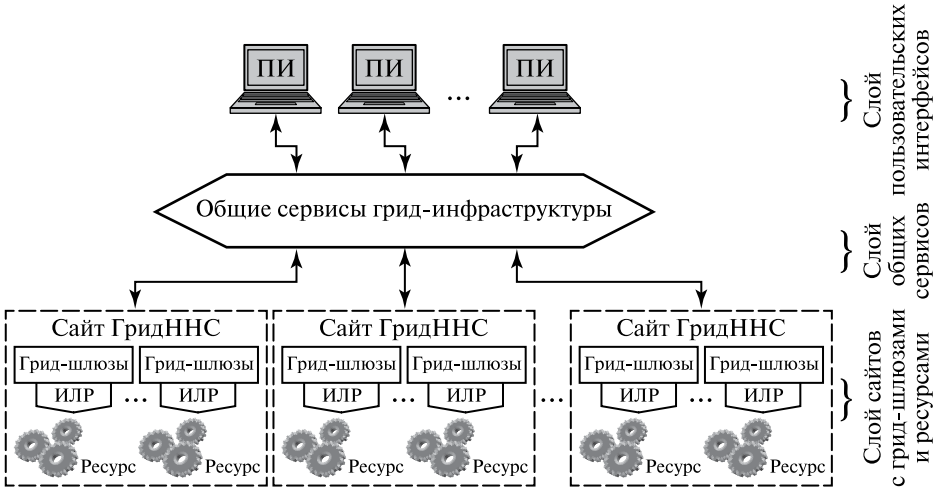


Рис. 1. Общая структура ГридННС

Интерфейс командной строки предназначен для пользователей с достаточно высоким уровнем компьютерной подготовки. С помощью консольных команд такие пользователи создают набор своих скриптов для автоматизации запуска задач. Основные консольные команды:

pilot-job-submit [options] job_definition.js	Запуск задания на выполнение в ГридННС
pilot-job-status [options] URI	Проверка текущего состояния задания
pilot-job-info [options] URI	Проверка текущего состояния задания с информацией о состоянии отдельных задач
pilot-task-status [options] URI	Проверка текущего состояния отдельной задачи
pilot-job-pause [options] URI	Приостановка выполнения задания
pilot-job-cancel [options] URI	Снятие задания с выполнения
pilot-query-jobs [options]	Запрос списка всех заданий, запущенных данным пользователем
pilot-job-resume [options] URI	Продолжение выполнения задания после приостановки
pilot-job-matchmake [options] job_file	Запрос списка ресурсов, удовлетворяющих требованиям задания

Интерфейс командной строки, дополненный GridFTP-сервером для упрощенного доступа к входным и выходным данным заданий, получил название «пользовательский интерфейс командной строки» (ПИКС).

Веб-интерфейс [Gulin et al., 2010] предназначен для создания и запуска заданий в ГридННС с использованием браузера. Он может быть применен для работы пользователей в грид-среде как с компьютера с установленной операционной системой (ОС) Linux, так и с установленной ОС Windows. Интерфейс поддерживает следующие основные функции:

- работа с сертификатами пользователя;
- создание заданий с использованием графических инструментов;
- создание задач с помощью веб-форм;
- работа с файлами пользователя и результатами вычислений;
- контроль прохождения задания и его отдельных задач (понятия «задание» и «задача» определены далее в п. 2.2).

Веб-интерфейс ВИГ реализован как надстройка веб-интерфейса над ПИКС, поэтому его функциональность совпадает по возможностям с ПИКС.

Прикладные ПИ для решения определенного класса вычислительных заданий в той или иной предметной области могут создаваться как на основе ПИКС, так и путем наращивания функциональных возможностей ВИГ.

Слой *грид-шлюзов* обеспечивает доступ к компьютерным ресурсам из ГридННС. В настоящее время ими являются вычислительные ресурсы и ресурсы хранения данных, представляющие собой GridFTP-серверы. В качестве грид-шлюза к вычислительным ресурсам используются WS-GRAM из GT4 с некоторыми расширениями. В частности расширен набор поддерживаемых менеджеров локальных ресурсов (МЛР). Разработаны адаптеры для МЛР Cleo [Prikhodko et al., 2010] и Slurm.

Слой *общесистемных сервисов и систем* отвечает за функционирование и управление ГридННС. Перечислим основные из них:

- информационная система;
- система мониторинга и учета ресурсов;
- сервис управления и распределения задач;
- система регистрации ресурсов и грид-сервисов;
- служба выдачи и управления цифровыми сертификатами;
- сервис проверки работоспособности ресурсов.

Информационная система (ИС) состоит из локальной ИС (ЛИС) и центральной ИС (ЦИС) [Степанова и др., 2010]. Обе части системы построены с использованием WS-MDS из GT4. Основная задача ИС — обеспечение других служб и сервисов актуальной информацией о текущем состоянии компонентов ГридННС и, в первую очередь, локальных ресурсов. В качестве схемы описания состояния ГридННС используется Glue Scheme вер. 1.3 (2007) с некоторыми изменениями. В настоящее время завершается работа по переходу ИС на InfoSys2, которая базируется на RESTful-сервисах и представляет собой полностью оригинальную разработку.

Система мониторинга и учета ресурсов (СМУР) [Belov et al., 2010] предоставляет веб-интерфейс, позволяющий пользователям ГридННС получать информацию о состоянии сервисов и ресурсов, их загрузке, а также данные о потребленных ресурсах. Объем предоставляемой информации зависит от прав конкретного пользователя или категории, к которой он принадлежит. Так, обычный пользователь может получить информацию о потребленных только им ресурсах, в то время как менеджер виртуальной организации (ВО) — о потребленных ресурсах всей ВО и каждым конкретным пользователем.

Система регистрации ресурсов и грид-сервисов (СРРГС) предназначена для ведения реестра всех компонент ГридННС за исключением ПИ. Она решает две основные задачи. Во-первых, это обнаружение сервисов (service discovery): здесь другие службы и сервисы могут узнать, что включено в ГридННС. Во-вторых, эта служба позволяет менеджерам ГридННС и сайтов управлять состоянием сайтов и отдельных сервисов, например, объявлять о планирующихся профилактических работах. Для выполнения задач используются только те сайты, которые зарегистрированы в СРРГС.

Служба выдачи и управления цифровыми сертификатами предназначена для выдачи пользователям и грид-сервисам ГридННС цифровых сертификатов стандарта X.509 (2005). Вся система безопасности ГридННС построена на Public Key Infrastructure (2001), с использованием цифровых сертификатов X.509.

Служба проверки работоспособности ресурсов ГридННС регулярно запускает тестовые задания на все ресурсы, которые находятся в рабочем режиме, и отслеживает результаты их выполнения.

Сервис управления и распределения задач «Пилот» (Pilot) предназначен для контроля за выполнением заданий пользователей и распределения задач из заданий по ресурсам в соответствии с требованиями пользователя и состоянием ресурсов. Данная система представляет собой RESTful-грид-сервис [Демичев и др., 2009, 2010].

Более подробно структура ГридННС рассмотрена в работе [Добрецов и др., 2010].

2. ПРИНЦИПЫ РАЗРАБОТКИ ПРОМЕЖУТОЧНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ГРИДННС

При построении ГридННС широко учитывался опыт работы с другими грид-инфраструктурами, полученный авторами, и, в первую очередь, опыт проектирования, развертывания и эксплуатации ресурсных центров и региональных базовых сервисов WLCG/EGEE (2001). Надо отметить, что указанная грид-инфраструктура является крупнейшей в мире, вышедшей на производственный уровень. Тем не менее, в процессе ее проектирования принят ряд, по мнению авторов, не очень удачных решений, которые постарались улучшить.

2.1. Использование архитектурного подхода REST

В своем развитии грид-инфраструктуры прошли ряд этапов. В настоящее время общепринятой является открытая архитектура грид-сервисов. На первом этапе для построения грид-инфраструктур использовались специальные протоколы и форматы данных. Сначала это был инструментарий GT2, в дальнейшем замененный на OGSI, которая являлась реализацией архитектуры OGSA. С развитием веб-сервисных технологий стало ясно, что необходимо слияние подходов, созданных в рамках развития грид первого поколения и веб-сервисных технологий, получивших широкое признание в мире. В результате появился набор спецификаций и стандартов, получивший название «инфраструктура ресурсов веб-сервисов» (Web Service Resource Framework, WSRF). Канонической реализацией WSRF стал инструментарий GT4.

Основной концепцией в WSRF является WS-Resource. В понимании WSRF, ресурсом является логическая сущность, которую можно отделить от других сущностей, т. е. идентифицировать. Ресурс может иметь не пустое множество свойств, представимых в виде информационного набора XML. Кроме того, он может иметь цикл существования. Его можно создать по запросу, и он может иметь ограниченное время жизни.

С точки зрения реализации, WS-Resource — это композиция из ресурса и веб-сервиса, через который можно получить доступ к этому ресурсу. Для каждого WS-Resource указывается End-Point Reference, указывающая на адрес ресурса. Спецификация WSRF предусматривает стандартные механизмы для управления циклом существования ресурсов (создание WS-Resource, присвоение ему идентификатора-EPR, уничтожение WS-Resource), получения множества свойств и взаимодействия со свойствами ресурса.

Опыт существующих грид-проектов показывает, что основные концепции, заложенные в основу открытой архитектуры грид-сервисов OGSA действительно являются ключевыми при создании грид. Однако WSRF имеет ряд существенных недостатков. В первую очередь это исключительная сложность протоколов и стандартов. В результате даже эталонная реализация OGSA в виде WSRF,

которой является инструментарий GT4, частично не соответствует стандартам и содержит многочисленные ошибки. В результате теряется главное, ради чего был предложен набор протоколов и стандартов WSRF, — построение универсальных, динамически расширяющихся грид. А если это так, то можно отойти от принципа «универсализма» в пользу более простых и, как следствие, проще реализуемых подходов.

В 2000 г. Р. Филдингом был предложен новый и гибкий подход к созданию сервисов — архитектурный стиль REST [Fielding, 2000]. Надо заметить, что WSRF и RESTful-сервисы имеют много общего (см., например, [Pautasso et al., 2008] и рис. 2).

Использование архитектурного стиля REST для построения веб-сервисов дало замечательный результат. Были реализованы, например, такие проекты как S3 сервис Amazon, Facebook предоставляет интерфейс для разработчиков, Atom Publishing Protocol.

Основываясь на этом положительном опыте, авторы предложили использовать стиль REST для реализации грид-сервисов в рамках концепции OGSA. Для проверки возможности реализации грид-сервисов на основе архитектурного стиля REST был выбран ключевой сервис в ГридННС — сервис управления потоком заданий «Пилот» [Демичев и др., 2010].

Сервис «Пилот» обеспечивает запуск и координацию выполнения задач на вычислительных ресурсах. Использование архитектурного стиля REST позволило создать грид-сервис с гораздо более простым интерфейсом прикладного программирования, чем в WSRF. Однако полноценная реализация грид-сервиса требовала формализации методов, имеющихся в REST, и расширения веб-сервиса, построенного в рамках REST, до грид-сервиса, имеющего стандартные методы управления циклом существования ресурсов, что было реализовано впервые в мире авторами настоящей работы.

Важным достоинством RESTful-грид-сервисов по сравнению с подходом, основанным на WSRF, является простая и ясная семантика запросов. В табл. 1 в качестве примера приведены интерпретации запросов к коллекции ресурсов.

Обратим внимание на то, что RESTful-сервисы использует протокол HTTP в своем прямом предназначении как протокол запросов, а не просто транспортный протокол, как это происходит в случае WSRF, где протокол HTTP

Рис. 2. Сравнение WSRF- и RESTful-подходов к проектированию веб-сервисов

используется для передачи SOAP (Simple Object Access Protocol) конвертов, сериализация и десериализация которых представляет дополнительные проблемы, в том числе и с совместимостью реализаций.

Таблица 1

Запросы к коллекции ресурсов

Запрос	Действие при обращении к коллекции	Действие при обращении к элементу коллекции
GET	Получение списка членов коллекции	Получение представления ресурса
PUT	Замещение существующей коллекции новой, т.е. коллекция рассматривается как самостоятельный ресурс	Обновление ресурса либо создание нового ресурса в коллекции с заранее определенным идентификатором ресурса
POST	Создание нового ресурса в коллекции с автоматическим присвоением ему идентификатора	Адресуемый ресурс рассматривается как самостоятельная коллекция, операция соответствует операции POST над коллекцией
DELETE	Удаление коллекции	Удаление ресурса

2.1.1. Управление циклом существования ресурсов

Управление циклом существования ресурсов является одной из ключевых особенностей грид-сервисов, не свойственной обычным веб-сервисам. Поскольку в грид, построенном согласно принципам OGSA, постоянно происходит создание и уничтожение ресурсов, и, как следствие, связанных с ними логических сервисов, необходим единый механизм для информирования о времени жизни сервисов/ресурсов, а также предсказуемого изменения этого времени жизни.

Для контроля времени жизни ресурсов и управления им авторами было предложено использовать заголовки протокола HTTP. Так, с параметром CurrentTime спецификации WSRF сопоставлен стандартный заголовок Date, для обозначения времени жизни ресурса — нестандартный заголовок Termination-Time.

Наличие заголовка Termination-Time в запросе используется для изменения времени жизни ресурса. В случае возможности его изменения грид-сервис должен выполнить действия, соответствующие запросу, и изменить время жизни ресурса. В ответе на запрос должен присутствовать заголовок Termination-Time, содержащий новое время жизни ресурса. В случае, если указанное изменение невозможно или не поддерживается для данного ресурса, грид-сервис отвергает запрос и должен вернуть ответ с кодом состояния 409, содержащий заголовок Location: urn:X-RESTful-Grid:invalid-termination-time.

Наличие заголовка Termination-Time в ответе грид-сервиса указывает, что данный ресурс может быть автоматически уничтожен сервисом в указанный момент времени, однако не гарантирует уничтожения ресурса. Отсутствие заголовка Termination-Time в ответе грид-сервиса означает, что время жизни ресурса не определено явно, и клиент не должен делать предположений о времени жизни ресурса.

2.1.2. Свойство идемпотентности REST-запросов

Для традиционных WSRF-веб-сервисов протокол HTTP выполняет исключительно транспортные функции, поэтому такие сервисы не могут использовать преимущества, связанные со свойством идемпотентности ряда методов протокола HTTP. RESTful-грид-сервисы могут использовать эти свойства для упрощения обработки ошибок на уровне доставки запросов и ответов при условии, что реализация грид-сервиса полностью выполняет требования HTTP на безопасность и идемпотентность, а именно:

• действия, реализуемые методами GET и HEAD, не должны выполнять никаких функций, помимо собственно возвращения представления ресурсов, и не должны вызывать изменения внутреннего состояния сервиса или ресурса, т.е. иметь побочные эффекты;

• действия, выполняемые методами PUT и DELETE, должны быть идемпотентными, т.е. побочные эффекты, вызываемые запросом с таким методом, должны быть одинаковы для любого числа повторений одного и того же запроса.

Выполнение этих требований позволяет повысить надежность работы клиентских приложений через плохие каналы связи, так как в случае отсутствия ответа сервера клиент может просто повторить идемпотентный запрос еще раз.

Однако до сих пор говорилось об использовании метода POST для создания новых ресурсов, а он не является ни безопасным, ни идемпотентным.

Один из способов обеспечения идемпотентности метода POST приводится в проекте спецификации Post Once Exactly (POE) [Nottingham, 2005]. Идея спецификации заключается в создании специального протокола в рамках HTTP, позволяющего ресурсам, поддерживающим этот протокол, гарантировать обработку операции POST не более одного раза независимо от числа запросов. Таким образом, операция POST становится идемпотентной.

Альтернативным решением проблемы создания ресурсов идемпотентными методами является генерация идентификаторов ресурсов на стороне клиентов следуя правилам, описанным в интерфейсе прикладного программирования сервиса, и использование метода PUT для создания ресурсов. При использовании этого подхода возникает проблема гарантии уникальности идентификатора создаваемого ресурса для исключения возможности случайного изменения уже существующего ресурса.

Используя такой метод создания ресурсов, можно сократить повторную пересылку представлений при помощи метода условного продолжения/прерывания запросов Expect/Continue протокола HTTP 1.1 и условного выполнения запросов If-None-Match.

2.1.3. Обработка ошибок

Спецификация WSRF предусматривает универсальный подход к передаче и обработке ошибочных состояний. Для этого вводится класс ошибок BaseFault, и накладывается требование использовать этот класс как базовый для любых ошибок, связанных с ресурсом [Lui, Madler, 2006]. Кроме того, определяются две стандартные ошибки:

- ResourceUnknownFault: данный ресурс не известен данному сервису;
- ResourceUnavailableFault: ресурс, указанный в сообщении, не доступен.

В RESTful-грид-сервисе используются для индикации ошибок стандартные коды статусов HTTP с документами, поясняющими детали произошедшей ошибки (табл. 2). Они должны использовать вместо соответствующих классов ошибок коды состояния HTTP со значениями 4xx и 5xx.

В качестве базового класса используется код состояния 400 Bad Request, которое указывается для любых ошибок, не попадающих в один из стандартных классов, перечисленных далее. При этом суть произошедшей ошибки должна описываться в представлении ресурса, содержащемся в ответе сервиса.

Для индикации обращения к неизвестному ресурсу следует использовать код состояния 404 Not Found (аналог ResourceUnknownFault), а для указания на недоступность ресурса применяется код состояния 403 Forbidden или 401 Unauthorized в зависимости от причины недоступности ресурса (аналог ResourceUnavailableFault). Кроме того, RESTful-грид-сервисы могут использовать прочие стандартные коды состояния HTTP для индикации распространенных ошибочных ситуаций.

Таблица 2

Коды ошибок и их интерпретация для RESTful-грид-сервисов

Код ошибки	Интерпретация
405 Method Not Allowed	Используется в том случае, если запрашиваемая операция над ресурсом не предусматривается данным сервисом. Например, сервис должен возвращать данный код при попытке изменения методом PUT ресурса, доступного только для чтения. В качестве дополнительной информации сервис в данном случае может также вернуть заголовок Allow, содержащий указание на приемлемые для данного ресурса HTTP-методы
406 Not Acceptable	Используется для согласования представления ресурса с клиентом в случае, если клиент запросил представление ресурса в формате, который не может быть предоставлен данным сервисом
409 Conflict	Используется в случае, если результат выполнения запрошенной операции может привести сервис или ресурс в невозможное или несовместимое состояние. Например, этот код ошибки может использоваться при попытке создания ресурса с идентификатором, который уже используется другим ресурсом
410 Gone	Данный ресурс более не доступен через данный сервис. Новый способ доступа к ресурсу не известен
415 Unsupported Media Type	Запрос к серверу содержал представление ресурса в формате, который не может быть обработан данным сервисом
503 Service Unavailable	Данный запрос не может быть обслужен сервисом в настоящее время. Ответ с таким кодом состояния может содержать заголовок Retry-After, при наличии этого заголовка пользователь может повторить попытку такого же запроса в указанное время

Для детализации причины, вызвавшей ошибку, ответ может содержать заголовок Location с URI, указывающим на ресурс, являющийся причиной ошибки. При этом в качестве ресурса может выступать в том числе и URN, указывая в таком случае категорию ошибки.

Таким образом, RESTful-грид-сервисы предоставляют разработчику даже более гибкий набор сообщений об ошибках, чем WSRF.

2.2. Обработка потока заданий (workflow)

Задания в ГридННС являются композитными объектами, состоящими из совокупности задач, порядок выполнения которых описывается с помощью направленного ациклического графа. Задачи являются терминалами в том смысле, что каждая должна быть распределена на конкретный ресурс для своего исполнения. Задания и задачи представляются в виде JSON-объектов [Crockford, 2006], которые валидируются при помощи соответствующих схем [Zyp, 2010]. На рис. 3 приведен пример задания, состоящего их двух задач, которые должны быть выполнены последовательно друг за другом.

Заметим, что определения задач могут находиться как непосредственно внутри задания (задача “b”), так и в отдельном файле (задача “a”). При передаче задания «Пилоту» содержание задачи всегда вставляется в тело задания. Таким образом, «Пилот» всегда имеет дело с развернутой формой задания, когда описания задач представлены явно.

Задания описывают логику выполнения. В приведенном примере задача “b” будет запущена на выполнение только по завершении выполнения задачи “a”.

Так как задание представляет собой ациклический направленный граф, то из одного узла может исходить несколько ребер. Это означает, что от завершения

```
{ «version»:2,
 «description»:»test job»,
 «default_storage_base»:»gsiftp://tb01.ngrid.ru/tmp/staging/»,
 «tasks»:[
 { «id»:»a»,
 «description»:»task #1»,
 «filename»:»task_a.js»,
 «children»:[«b»]
 },
 { «id»:»b»,
 «definition»:
 { «version»:2,
 «executable»:»/bin/cat»,
 «arguments»:[«-n»],
 «stdin»:»test.log»,
 «stdout»:»gsiftp://tb05.ngrid.ru/home/john/examples/result
s.txt»
 }
 }
 ]
}
```

Рис. 3. Пример композитного задания ГридННС

родительской задачи зависят несколько задач-детей. Аналогично, в узел может приходить несколько ребер. В этой ситуации выполнение задачи может начаться только по завершении выполнения всех задач, от которых она зависит.

Если пользователь по ошибке создал циклический граф, то «Пилот» диагностирует такую ситуацию и задание будет отвергнуто с сообщением об ошибке.

Отметим, что граф определяет логические связи между задачами и никак не связан с данными, порождаемыми в результате выполнения задач. Вопрос управления данными рассмотрен далее.

Важно понимать, что такое «успешное завершение» задачи. В ГридННС принято следующее определение успешности. Для каждой задачи может быть установлено максимально допустимое значение кода возврата задачи (exit code), которое считается успешным завершением. По умолчанию, это значение равно нулю. Задача, завершившаяся с кодом возврата, превышающим указанное значение, признается не успешной, и все задачи, зависящие от нее, не будут запущены. Остальные задачи обрабатываются как обычно.

Важную роль в процессе выполнения композитных заданий играют дополнительные требования, которым должны удовлетворять ресурсы со стороны запускаемых задач. Требования могут иметь различный характер. Это может быть, например, заказ определенного количества ЦПУ, оперативной памяти, времени выполнения. Могут быть и более специфические требования к наличию предустановленного прикладного ПО, библиотек.

Описание заданий и задач в ГридННС поддерживает богатый набор требований. Если какое-либо требование указано только в описании задания, то оно используется при запуске задачи. Если требование указано в теле задачи, то всегда используется именно оно. Приведем пример требования: “software”: “mvapich, abinit > 6, orca==2.6.35”. Оно указывает, что ресурс, на котором будет исполняться задача, должен иметь предустановленный пакет ABINIT версии больше, чем 6, ORCA — версии 2.6.35, должна быть установлена поддержка MVAPICH. Данную информацию «Пилот» получает из ИС и учитывает при выборе ресурса, на который будет запущена задача.

2.3. Модель управления данными

ГридННС является вычислительным гридом, ориентированным на суперкомпьютерные вычисления. Поэтому управление данными является важной, но не определяющей стороной инфраструктуры. В связи с этим в ГридННС нет средств создания множественных реплик и других средств, характерных для грид, ориентированных на обработку большого объема данных. В частности, предполагается, что GridFTP-серверы небольшого объема, которые будут устанавливаться пользователями самостоятельно или в рамках ВО, обеспечат достаточный уровень надежности сервиса. В частности, можно использовать GridFTP-серверы, которые входят в состав пользовательских интерфейсов ВИГ и ПИКС.

Важной особенностью подхода к работе с данными в ГридННС является полное разделение потока управления заданиями и данными. Это существенно отличает ГридННС от WLCG/EGEE, где данные передаются совместно с описаниями

задач и обрабатываются одним сервером управления нагрузкой (WMS). Подход ГридННС является разновидностью P2P сетей, которые имеют высокую нагрузочную способность вследствие распределенного характера.

Управление данными включает в себя передачу данных на вычислительный ресурс и получение результатов вычислений. При этом под данными понимается вся совокупность данных, которая необходима для выполнения задачи, включая программу. Таким образом, в момент запуска задачи на ресурсе должны быть все необходимые компоненты. Для этого в описании задачи указывается местоположение всех входных файлов. Им может быть локальная файловая система вычислительного ресурса и/или GridFTP-сервер. Каждый файл описывается парой ключ – значение. Пример описания файлов в задаче приведен на рис. 4.

Если несколько файлов находятся на одном GridFTP-сервере, то можно указать параметр default_storage_base, а в именах файлов — путь к файлам относительно пути, прописанного в этом параметре. Аналогичный параметр можно использовать в теле задания. В этом случае он будет использован для всех задач, в которых данный параметр не определен. Это позволяет значительно сократить запись имен файлов.

Передача файлов между GridFTP-серверами происходит с использованием сервиса RFT [Kettimuthu et al., 2007], обеспечивающего контроль передачи данных. При работе с файлами следует обращать внимание на наличие бита “executable” на выполняемых файлах, поскольку попытка выполнения исполняемого файла без этого бита приводит к ошибке. Поэтому для начальной выгрузки файлов с компьютера пользователя на сервер GridFTP рекомендуется использовать команду uberftp. У пользователей ВИГ такая проблема отсутствует, так как наличие этого бита контролируется ВИГ.

2.4. Модель аутентификации ГридННС

Модель аутентификации ГридННС построена на основе PKI с использованием цифровых сертификатов стандарта X.509. В рамках инфраструктуры был развернут центр выдачи сертификатов. В настоящее время для работы в ГридННС признаются как сертификаты, выданные центром выдачи сертификатов ГридННС, так и РДИГ.

```
«default_storage_base»:»gsiftp://example.org/other/files/»,
«input_files»:{
 «hello.txt»:»hello.txt»,
 «foo.txt»:»/bar.txt»,
 «qux»:»gsiftp://example.org/my/directory/qux/»
},
«ouput_files»: {
 «qux/test.txt»: «gsiftp://example.org/my/output/117 test.txt»
}
```

Рис. 4. Пример описания файлов в задаче

Выполнение заданий пользователя происходит с использованием короткоживущих прокси-сертификатов. Это существенно повышает защищенность системы от возможного компрометирования отдельных грид-сервисов. Для выполнения задач длительностью больше, чем время жизни прокси-сертификата, используется механизм обновления прокси-сертификатов с помощью специального сервиса — MyProxy [Novotny et al., 2001].

Передача прав от одного сервиса к другому производится методом делегации, что исключает передачу закрытых ключей по каналам связи.

2.5. Виртуальные организации

Модель организации работы пользователей в ГридННС основана на понятии виртуальных организаций [Foster et al., 2001]. Это классический подход организации работ в грид.

Все пользователи ГридННС должны быть зарегистрированы хотя бы в одной ВО. Такая организация работ решает две основные проблемы: авторизацию доступа пользователей к ресурсам и разделение их прав доступа.

Каждый ресурс имеет своего владельца, которому принадлежат исключительные права на его использование. Стандартная процедура получения доступа к ресурсу подразумевает, что пользователь договаривается с владельцем ресурса, получает учетную запись с возможностью удаленного доступа по паролю.

Для грид такой подход неприемлем, так как требует огромного количества договоренностей каждого с каждым. Кроме того, это создает большую нагрузку на системных администраторов по поддержке большого числа учетных записей. В грид используется подход, основанный на том, что менеджер ВО, представляя всех пользователей своей ВО, договаривается от их имени об обслуживании с владельцем ресурса. Вопрос о поддержке учетных записей решается за счет использования динамического пула учетных записей для пользователей грид. Это означает, что поступивший запрос на обслуживание от имени какого-либо пользователя получает первую свободную запись, которая будет освобождена после завершения работы.

Для решения проблемы прав доступа используется VOMS-сервер [Ceccanti, 2007], который обеспечивает регистрацию пользователей в ВО и регистрацию их прав и ролей. Механизм передачи информации о правах и роли пользователя в данной ВО — это добавление некритических VOMS-расширений в прокси-сертификат пользователя. Таким образом, задача приходит на ресурс вместе с прокси-сертификатом, который содержит информацию о правах данного пользователя.

Сертификаты решают проблему аутентификации, а VOMS-расширения — проблему авторизации пользователей.

2.6. Эксплуатация ГридННС

Опыт эксплуатации крупнейшей в мире грид-инфраструктуры WLCG/EGEE показал, что успех зависит не только от технических решений, но, в первую очередь, от людей, которые будут обеспечивать работу инфраструктуры, что

подразумевает наличие квалифицированного персонала, полноты регламентов, эффективной поддержки пользователей и многое другое.

ГридННС с самого начала строился с учетом этих факторов. Большое значение имеет организованная служба поддержки системных администраторов ресурсов и пользователей на основе системы билетов Trac. Дополнительно, пользователи ГридННС могут обмениваться своим опытом через списки рассылки. Был создан сайт http://www.ngrid.ru, использующий технологию Wiki, на котором публикуются новости ГридННС, размещаются многочисленные инструкции и регламенты. В дальнейшем на сайт предполагается поместить учебные материалы.

Для обеспечения эксплуатации ГридННС была организована система проверки работоспособности как отдельных сервисов, так и инфраструктуры в целом. Эта система проводит запуск тестовых заданий на регулярной основе. По результатам тестов системные администраторы получают уведомления о сбоях в работе.

3. ЭКСПЕРИМЕНТАЛЬНОЕ ИССЛЕДОВАНИЕ ПРОИЗВОДИТЕЛЬНОСТИ СЕРВИСА PILOT

Для оценки производительности сервиса Pilot были проведены замеры различных времен, требующихся для запуска заданий в ГридННС. В качестве референтной точки использовался сервис WMS ППО gLite.

Для этого был собран стенд на виртуальных машинах, каждой их которых было доступно 512 МБ оперативной памяти и 6 ГБ жесткого диска. Сам стенд был развернут на сервере с общим объемом оперативной памяти 16 ГБ, который имел 3 ЦПУ Intel Xeon E5430 (2,66 ГГц). Запуск заданий велся на макет сайта, состоявшего из шлюза, к которому был подключен вычислительный узел. В качестве локального менеджера управления заданиями на сайте использовалась система Torque. Для случая запуска задач gLite использовался грид-шлюз CREAM.

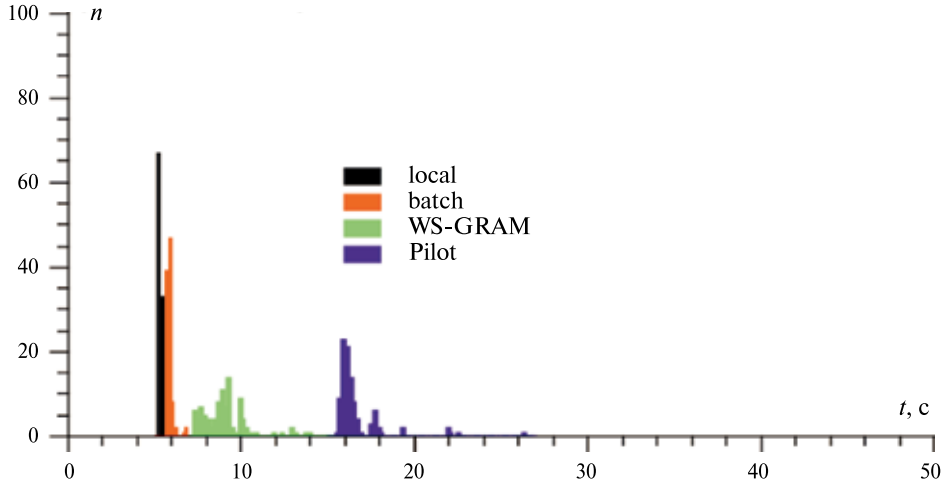


Рис. 5. Расход времени на запуск задачи: локальный, через Torque, через грид-шлюз ГридННС, через Pilot

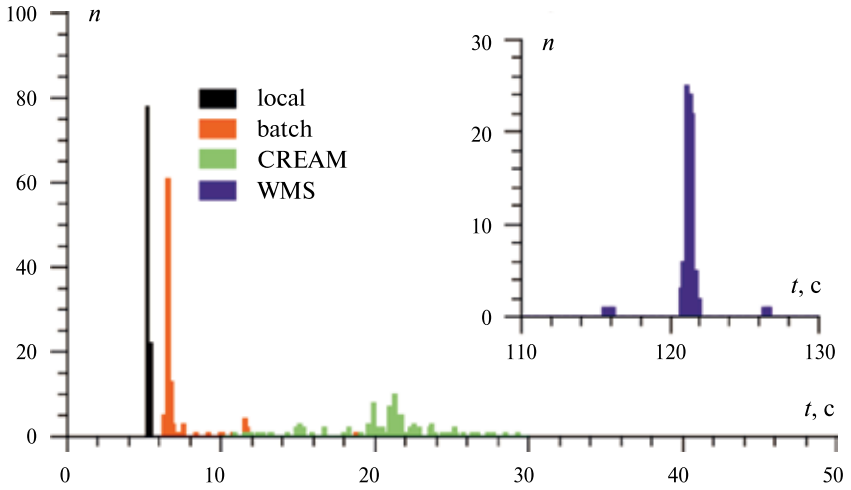


Рис. 6. Расход времени на запуск задачи: локальный, через Torque, через грид-шлюз gLite CREAM, через gLite WMS. Гистограмма запуска через WMS выполнена в виде врезки, так как ее пик находится в районе 120 с

Результаты экспериментов представлены на рис. 5 (см. с. 64) и 6, из которых видно, что расходы времени на локальный запуск задач в ГридННС и gLite составляют 0,5 и 1,8 с соответственно. Расходы времени с учетом обработки задачи на грид-шлюзе составляют 3,3 с для ГридННС шлюза и 13,4 с для шлюза gLite CREAM.

С учетом указанных данных время обработки задачи для сервиса Pilot составляет 7,5 с, в то время как для gLire WMS — 99 с.

В рамках экспериментального исследования была проведена проверка сервиса Pilot на предмет устойчивости и надежности. Для этого в течение нескольких суток проводились запуски заданий блоками по 7–13 через каждые 10 мин от 8 разных пользователей. Примерная нагрузка составила 480 заданий в час.

Проведенные эксперименты показали эффективность сервиса, его устойчивую и надежную работу.

ЗАКЛЮЧЕНИЕ

В процессе разработки ГридННС был решен ряд принципиальных вопросов создания грид-инфраструктур. Главным результатом, достигнутым в ходе реализации проекта, является обобщение RESTful-веб-сервисов до полноценных грид-сервисов. В частности, были предложены механизм реализации контроля за временем существования грид-сервисов и принципы построения RESTful-грид-сервисов с использованием идемпотентных операций. Последнее позволяет значительно повысить устойчивость сервисов к возможным сбоям и обеспечить их корректное восстановление после сбоев.

Предложенные подходы позволили отказаться от сложных в реализации, развертывании, настройке WSRF-сервисов для ряда ключевых компонент. С использованием архитектурного стиля REST был реализован сервис управления композитными заданиями (workflow) и распределения задач по ресурсам. В настоящее время завершается работа по переводу информационной системы ГридННС на RESTful-грид-сервисы. В дальнейшем предполагается полный отказ от WSRF-сервисов.

Как следствие, переход на RESTful-грид-сервисы позволяет уйти от использования SOAP в качестве протокола обмена запросами. Был использован гораздо более простой и эффективный с точки зрения сериализации и десериализации формат представления композитных заданий на основе JSON. Важно заметить, что в указанном подходе протокол HTTP используется в своем прямом назначении как протокол обмена запросами, а не просто как транспортный протокол в случае SOAP.

Реализованная в ГридННС модель управления данными отличается простотой. Она позволила разделить потоки управления и данных. Их смешение в WLCG/EGEE/EGI грид значительно усложняет работу системы и является ее узким местом.

ГридННС — это полнофункциональная грид-инфраструктура, содержащая полный набор сервисов и систем, обеспечивающих ее работу. Предусмотрены службы поддержки пользователей, развернут сайт, на котором публикуются новости ГридННС, помещены многочисленные инструкции и регламенты.

В настоящее время ГридННС развернута на полигоне, который состоит из 11 сайтов. Суммарно полигон охватывает более 5000 ЦПУ. Контроль за работоспособностью сайтов осуществляется путем регулярного запуска тестовых задач.

ЛИТЕРАТУРА

[Демичев и др., 2010] *Демичев А.*, *Ильин В.*, *Крюков А.*, *Шамардин Л.* Реализация программного интерфейса грид-сервиса Pilot на основе архитектурного стиля REST // Вычисл. методы и программирование. 2010. Т. 11. С. 62–65.

Демичев А., *Крюков А.*, *Шамардин Л.* Принципы построения грид с использованием Restful-веб-сервисов // Программные продукты и системы. 2009. № 4.

[Добрецов и др., 2010] *Добрецов В. Ю.*, *Ильин В. А.*, *Кореньков В. В.*, *Крюков А. П.*, *Рябов Ю. Ф.* Проектирование и разработка грид-инфраструктуры для национальной нанотехнологической сети // Тр. Международ. конф. «Грид-2010». ОИЯИ, Дубна, 2010. С. 357–363.

Степанова М. М., *Стесик О. Л.*, *Кастерин Д. С.*, *Яковлев С. Л.* Информационная система ГридННС // Тр. Международ. конф. «Грид-2010». ОИЯИ, Дубна, 2010. С. 423–428.

[Belov et al., 2010] *Belov S.*, *Korenkov V.*, *Lensky I.*, *Matveev M.*, *Matveeva E.*, *Mitsyn S.*, *Oleynik D.*, *Petrosyan A.*, *Semenov R.* Monitoring and Accounting for GridNNN Project // Proc. Intern. Conf. Grid-2010. JINR, Dubna, 2010. P. 43–46.

Belov S. D., *Mitsyn S. V.* Development of Real-Time Visualization Service Based on Google Earth for GridNNN Project // Proc. Intern. Conf. Grid-2010. JINR, Dubna, 2010. P. 47–50.

[Ceccanti, 2007] *Ceccanti A.* A VOMS Overview // NRENS and Grids Workshop, Malaga, 29–30 Nov., 2007. [Электрон. текст]. Режим доступа: http://www.terena.org/activities/nrens-n-grids/workshop-06/slides/ceccanti-voms-overview.pdf.

[Crockford, 2006] *Crockford D.* The application/json Media Type for JavaScript Object Notation (JSON): Technical rep. IETF Network Working Group. July 2006. RFC4180.

[Fielding, 2000] *Fielding R. T.* Architectural styles and the design of network-based software architectures: Doctoral dissertation. Univ. California, Irvine, 2000.

[Foster, Kesselman, 1998] *Foster I., Kesselman C.* The Globus Project: A Status Report // Proc. Heterogeneous Computing Workshop. IEEE Press, 1998. P. 4–18.

[Foster et al., 2001] *Foster I., Kesselman C., Steven T.* Anatomy of the Grid [Электрон. текст]. Режим доступа: http://www.globus.org/alliance/publications/papers/anatomy.pdf.

Globus Toolkit ver. 2.0. [Электрон. ресурс]. Режим доступа: http://www.anl.gov/techtransfer/docs/GlobusToolkt_2new_logo.doc.

GLUE Scheme. [Электрон. ресурс]. Режим доступа: http://forge.ogf.org/sf/projects/glue-wg.

[Gulin et al., 2010] *Gulin A. P., Kiryanov A. K., Klopov N. V., Oleshko S. B., Ryabov Yu. F.* Web-Based User Interface for GridNNN // Proc. Intern. Conf. Grid-2010. JINR, Dubna, 2010. P. 121–124.

[Kettimuthu et al., 2007] *Kettimuthu R., Allcock W., Liming L., Navarro J.-P., Foster I.* GridCopy: Moving Data Fast on the Grid // Proc. 4th High Performance Grid Computing Workshop to be held in conjunction with Intern. Parallel and Distributed Processing Symp. (IPDPS 2007). March, 2007.

[Lui, Madler, 2006] *Lui L., Madler S.* Web Service Base Fault 1.2: Technical rep. OASIS Standerd, April 2006.

[Nottingham, 2005] *Nottingham M.* POST One Exactly (POE): Technical rep. IETF Network Working Group Internet – Draft. Mar. 2005. [Электрон. текст]. Режим доступа: http://tools.ietf.org/html/draft-nottingham-http-poe-00.

[Novotny et al., 2001] *Novotny J., Tuecke S., Welch V.* An Online Credential Repository for the Grid: MyProxy // Proc. 10th Intern. Symp. High Performance Distributed Computing (HPDC-10). IEEE Press, Aug. 2001.

OASIS. Web Services Resource Framework (WSRF) — Primer v1.2. Committee Draft. 02–23 May 2006. [Электрон. ресурс]. Режим доступа: http://docs.oasis-open.org/wsrf/wsrf-primer-1.2-primer-cd-02.pdf.

Open Grid Services Infrastructure (OGSI). Version 1.0. GDF-R-P.15. 27 June 2003.

[Pautasso et al., 2008] *Pautasso C., Zimmermann O., Leymann F.* RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision // Proc. 17th Intern. World Wide Web Conf. (WWW2008). Bejing, China, Apr. 2008.

Prikhodko N. V., Abramovsky V. A., Kryukov A. P. Integration of Local Resource Managers in GridNNN: Cleo Example // Proc. of Intern. Conf. Grid-2010. JINR, Dubna, 2010. P. 191–195.

Public Key Infrastructure [Электрон. ресурс]. Режим доступа: http://www.sun.com/blueprints/0801/publickey.pdf.

[Richardson, Ruby, 2007] *Richardson L., Ruby S.* RESTful Web Services Web services for the real world. O'Reilly Media, 2007.

Shamardin L., Demichev A., Kryukov A., Ilyin V. GridNNN Job Execution Service: A Restful Grid Service // Proc. Intern. Conf. Grid-2010. JINR, Dubna, 2010. P. 215–219.

The Open Grid Services Architecture. Version 1.5. GFD-I.080. 24 July 2006.

WLCG. [Электрон. ресурс]. Режим доступа: http://lcg.web.cern.ch/lcg/, 2001, gLite. URL: http://www.glite.org.

X.509: Public-key and attribute certificate frameworks [Электрон. ресурс]. Режим доступа: http://www.itu.int/rec/T-REC-X.509-200508-I, 2005.

[Zyp, 2010] *Zyp K.* A JSON Media Type for Describing the Structure and Meaning of JSON Documents: Technical rep. IETF Network Workong Group, draft-zyp-json-schema-02, Mar. 2010.

MAIN APPROACHES TO BUILDING OF GRID INFRASTRUCTURE OF NATIONAL NANOTECHNOLOGY NETWORK

A. P. Kryukov, A. P. Demichev, V. A. Ilyin, L. V. Shamardin

Skobeltsyn Institute of Nuclear Physics,
Lomonosov Moscow State University (*SINP MSU*), *Moscow*, *Russia*

The main objective of the Grid infrastructure of the National Nanotechnology Network (GridNNN) — is to provide scientists, engineers and graduate students a unified secure remote access to supercomputer resources. For the first time in the world GridNNN used the REST architectural style for implementing Grid services. The infrastructure GridNNS separated data streams and control streams. Data Management GridNNS is done on the data directly from the server data storage to the site where the task will run, and the results — on the storage server. A comparison of the developed workload service Pilot and a similar service on gLite. Shown its effectiveness, sustainability and reliability.

This work was partially supported by RFBR (N. 10-07-00332), Ministry of Education and Science of RF (Contracts N. 01.647.11.2004, 02.740.11.0388 and SSh N. 4142.2010.2).

Keywords: high performance computing, distributed computing, Grid, GridNNN, REST, RESTful-Web-service.

Kryukov Alexander Pavlovich — leading scientist, candidate of physical and mathematical sciences, e-mail: kryukov@theory.sinp.msu.ru.

Demichev Andrey Pavlovich — senior scientist, candidate of physical and mathematical sciences, e-mail: demichev@theory.sinp.msu.ru.

Ilyin Vyacheslav Anatolevich — head of the laboratory, doctor of physical and mathematical sciences, e-mail: ilyin@sinp.msu.ru.

Shamardin Lev Vitalevich — junior scientist, e-mail: shamardin@theory.sinp.msu.ru.

УДК 004.75+004.41

АРХИТЕКТУРНО-ТЕХНОЛОГИЧЕСКИЕ АСПЕКТЫ ЭВОЛЮЦИИ ГРИД

В. А. Васенин, А. С. Шундеев

Научно-исследовательский институт механики Московского государственного университета им. М.В. Ломоносова (НИИ механики МГУ)

Архитектура и концептуальные положения грид были разработаны в период 2001–2006 гг. Этот период связан с бурным развитием технологии больших веб-сервисов, положенной в основу архитектуры грид. После 2006 г. возник ряд альтернативных подходов как к построению распределенных программных систем (RESTful-веб-сервисы), так и к решению задач, которые традиционно относятся к области грид (облачные вычисления). В этой связи актуальными являются исследования, направленные на анализ истории и перспектив развития технологии грид. Результаты такой работы представлены в настоящей статье.

Ключевые слова: грид-технология, облачные вычисления, промежуточное программное обеспечение.

ВВЕДЕНИЕ

В 2011 г. исполнилось десять лет с момента выхода в свет фундаментальной работы [Foster et al., 2001], определившей направление развития технологии грид на последующее десятилетие. В этой связи актуальным представляется исследование, направленное на анализ тех задач, которые в явной или неявной форме были поставлены в этой работе. Особый интерес в настоящее время представляет анализ выполнения этих задач в рамках выбранных и развиваемых впоследствии подходов (моделей, методов и средств) их решения.

Работа [Foster et al., 2001] содержит концептуальное описание введенного авторами класса распределенных программных систем, получивших название *грид-системы*, функционирующих в рамках так называемых *виртуальных организаций*. Понятие виртуальной организации является одним из ключевых и изначально имело следующую трактовку. В рамках проведения распределенных, как правило, междисциплинарных научных исследований научные коллективы (реально существующие организации) на договорных началах объединяют имеющиеся у них информационные и вычислительные ресурсы с целью решения поставленных перед ними задач. При этом вырабатываются политики совместного, разделяемого и безопасного использования объединенных ресурсов. Автоматизация процессов подобного использования ресурсов возлагается на грид-систему. Впоследствии понятие виртуальной организации было обобщено и вышло за рамки научных коллективов и выполняемых ими исследований.

Васенин Валерий Александрович — заведующий лабораторией, доктор физико-математических наук, e-mail: vasenin@msu.ru.

Шундеев Александр Сергеевич — ведущий научный сотрудник, кандидат физико-математических наук, e-mail: shundeev@msu.ru.

Свое описание грид-систем авторы работы [Foster et al., 2001] назвали «анатомическим» и построили его вокруг введенного стека протоколов грид-системы. С позиций общей архитектуры эти протоколы расположены на пяти уровнях и обеспечивают механизмы взаимодействия грид-системы: с предоставляемыми в рамках виртуальной организации ресурсами нижнего уровня (уровень *адаптации*); с внешними потребителями (уровень *приложений*); взаимодействия отдельных компонентов грид-системы друг с другом (уровни *связи*, *ресурсов*, *кооперации*). Необходимо отметить, что выделенные уровни в работе были только обозначены и качественно описаны. Их детальное проектирование и реализация должны были быть предметом последующих работ. В работе [Foster et al., 2001] при описании архитектуры используется метафора песочных часов, которые имеют широкие основание и верх, а также узкое горло. Основанием в случае протоколов грид является уровень адаптации, а верхом — уровень приложений. В рамках разработки архитектуры грид выдвигается минимальный набор требований к этим двум уровням. Цель такого подхода заключалась в том, чтобы избежать ограничений как на потенциальные прикладные области и задачи, в которых может применяться данная технология, так и на типы вычислительных и информационных ресурсов, которые могут быть задействованы в рамках виртуальных организаций. В то же время узкое горло, в которое попадали уровни связи, ресурсов и кооперации, планировалось проработать более детально.

Следующей по важности является вышедшая через год работа тех же авторов [Foster et al., 2002], в которой дается «физиологический» взгляд на грид-системы. Если анатомический взгляд касался протоколов взаимодействия распределенных компонентов грид-системы, то физиологический направлен на внутреннюю архитектуру подобных компонентов. Вводится понятие динамически создаваемого грид-сервиса, а также принципов работы с ним. Другое важное решение касалось уровня связи. Было принято принципиальное решение реализовывать его на основе так называемых больших веб-сервисов, о которых речь пойдет далее. Следует отметить, что в то время выбор такого подхода был далеко не очевиден. Например, в предшествующей работе [Foster et al., 2001] в качестве промежуточного программного обеспечения рассматривалась технология CORBA (Common Object Request Broker Architecture) [Henning, 2006]. Более того, в дальнейшем, при проектировании открытой инфраструктуры грид-сервисов [Banks et al., 2004], были заложены механизмы совместимости дескрипторов грид-сервисов со ссылками протокола IIOP, на котором базируется взаимодействие распределенных объектов CORBA.

Работа [Foster et al., 2002] дала толчок к созданию открытой инфраструктуры грид-сервисов [Tuecke et al., 2003; Banks et al., 2004], в рамках которой были разработаны механизмы, относящиеся к узкому горлу протоколов грид. Разработка открытой инфраструктуры и сопутствующих инструментальных средств позволила выполнить ряд значимых проектов по созданию прикладных грид-систем. Опыт выполнения этих проектов был обобщен и проанализирован [Foster et al., 2004], что позволило оформить результаты этой деятельности в виде набора типовых сценариев — стандартных вариантов использования грид-технологии. Проделанная за эти годы работа позволила сформулировать основные положения открытой архитектуры грид-сервисов [Foster et al., 2005, 2006]. В рамках этой

архитектуры были выделены группы грид-сервисов с позиций их функциональных возможностей и решаемых задач, а также концептуально описаны общие сценарии взаимодействия между грид-сервисами при решении тех или иных типовых прикладных задач.

Необходимо отметить, что все рассмотренные работы относятся к 2001–2006 гг. Этот период можно охарактеризовать как эпоху больших веб-сервисов*, которая ознаменовалась рождением, бурным ростом, а затем и определенной стагнацией этой технологии. Учитывая, что большие веб-сервисы являются технологическим фундаментом для грид-систем, особую актуальность обретают следующие вопросы, попытке ответа на которые и посвящена данная работа: насколько технология больших веб-сервисов подходит для создания и развития грид-систем, имеются ли альтернативы для ее замены, насколько перспективны предлагаемые решения?

1. ПРОМЕЖУТОЧНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Как было отмечено ранее, грид-системы представляют собой некоторый класс распределенных программных систем и для их создания, естественно, необходимо использовать некоторое промежуточное программное обеспечение [Emmerich et al., 2007]. Проанализируем, какие возможности в выборе промежуточного программного обеспечения имелись на период начала развития грид-технологий и какие варианты имеются сейчас.

1.1. Удаленные вызовы процедур

Распределенная программная система состоит из набора компонентов, которые выполняются в виде отдельных процессов операционной системы. Процессы при этом могут выполняться как на одном, так и на разных компьютерах. Сетевые операционные системы содержат базовый набор инструментов, позволяющих реализовать межпроцессное взаимодействие. Подобные инструментальные средства носят низкоуровневый характер. В результате, если ограничиваться только ими, то основные усилия разработчиков будут направлены на реализацию именно межпроцессного взаимодействия, а не бизнес-логики распределенной системы. Это вызвало потребность в появлении особо класса программного обеспечения, получившего название *промежуточного* (*middleware*). Наиболее полно данное понятие раскрыто в работе [Emmerich et al., 2007]. Под промежуточным программным обеспечением понимаются высокоуровневые инструментальные средства, облегчающие разработку распределенных систем, принимая весь «груз» взаимодействия с ядром операционной системы на себя.

Исторически первым примером промежуточного программного обеспечения можно считать *программные гнезда* (*sockets*), реализованные в 1982 г. в операционной системе UNIX BSD 4.1. Данный механизм предоставляет разработчику

* Под большими веб-сервисами будем понимать их реализацию на основе набора спецификаций XML, SOAP, WSDL, BPEL, более подробные сведения о которых будут представлены далее.

интерфейс, не зависящий от сетевого протокола и среды передачи данных. Несмотря на то, что программные гнезда устраняют ряд существенных неоднородностей, связанных со средой передачи данных, использование их напрямую остается непростой задачей (одновременно приходится манипулировать девятью библиотечными функциями — *socket*, *bind*, *connect*, *listen*, *accept*, *send*, *recv*, *shutdown*, *close*) и требует высокой квалификации от разработчика. Данная сложность кажется излишней, если учесть, что на практике наиболее распространенным был следующий сценарий межпроцессного взаимодействия.

Выделяются два процесса, один из которых называется *клиентом*, а другой — *сервером*. Сервер может принимать от клиента данные, обрабатывать их и возвращать результат обработки назад клиенту. На время обработки и до получения результата клиент блокируется. Создается иллюзия того, что клиент удаленно на сервере вызывает выполнение некоторой процедуры. Естественно, что в рамках распределенной системы один и тот же компонент может выступать как в роли клиента, так и в роли сервера. Описанный сценарий межпроцессного взаимодействия был положен в основу промежуточного программного обеспечения, получившего название *вызовы удаленных процедур* (Remote Procedure Calls, RPC). Впервые механизм RPC был реализован в 1984 г. компанией Sun Microsystems.

Кратко рассмотрим общую архитектуру механизма RPC, схематично представленную на рис. 1. Для описания интерфейса сервера, как правило, используется специализированный *язык описания интерфейсов* (Interface Definition Language, IDL). Он должен предоставлять возможность описания типов данных, используемых в качестве аргументов вызова, и результатов выполнения удаленных процедур, а также сигнатур самих удаленных процедур.

В табл. 1 приведен пример подобного описания, в котором в качестве IDL выбран язык Slice (Specification Language for ICE), используемый в программном средстве ICE (Internet Communications Engine) [Henning, Spruiell, 2010]. В данном описании интерфейс сервера состоит всего из одной процедуры ping, получающей на вход два строковых аргумента. Результатом выполнения процедуры является структура данных pingResult, имеющая два поля.

В состав инструментария промежуточного программного обеспечения класса RPC может входить специализированный компилятор, который обеспечивает трансляцию IDL описания в тексты программ на некотором языке программирования общего назначения (например, C, C#, Python и другие). Данные тексты

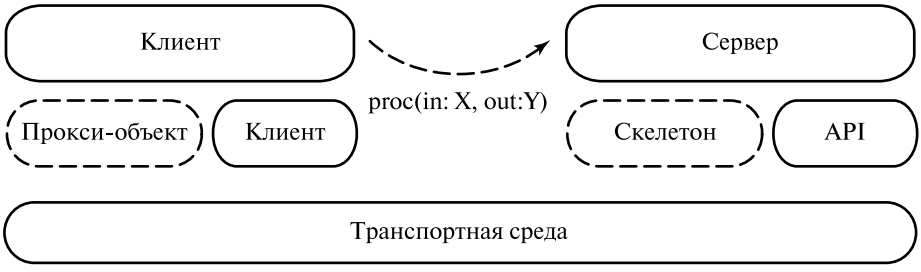


Рис. 1. Архитектура удаленного вызова процедур

программ используются для реализации клиента и сервера. Часть сгенерированного программного кода, используемого для реализации клиента, принято называть *прокси-объектом* (proxy). Вызов соответствующего метода у прокси-объекта инициирует выполнение удаленной процедуры на сервере. Часть сгенерированного программного кода, используемого для реализации сервера, принято называть *скелетоном* (skeleton). Если прокси-объект представляет собой готовый программный код, не требующий дополнительных модификаций, то скелетон представляет собой лишь заготовку. В случае объектного языка программирования в качестве скелетона может выступать класс, содержащий виртуальные методы, которые соответствуют удаленным процедурам. Для реализации сервера подобные виртуальные методы должны быть переопределены. Представленную схему использования RPC можно назвать «канонической». Далее в работе будут рассмотрены и другие схемы.

Таблица 1

Пример описания интерфейса на языке Slice

Описание данных	Описание процедур

`struct pingResult`
`{`
`bool flerror;`
`string message;`
`}`
	`interface Server`
	`{`
	`pingResult ping(`
	`string weblogname,`
	`string weblogurl);`
	`}`

Существенным недостатком ранних механизмов RPC было то, что они позволяли создавать только однородные распределенные системы, компоненты которых реализовывались на одном языке программирования (как правило — C) и могли выполняться только под одной операционной системой (как правило — семейства Unix). В результате к середине 90-х гг. прошлого столетия назрела острая необходимость в создании промежуточного программного обеспечения типа RPC, позволявшего создавать гетерогенные распределенные системы, охватывающие наиболее распространенные языки программирования и операционные системы. Результатом стало создание технологии CORBA, работы над которой велись в рамках консорциума OMG (Object Management Group). В 1997 г. была опубликована спецификация CORBA 2.0, в которой поддерживалось отображение IDL в языки программирования C и C++, а в 1998 г. было добавлено отображение в язык Java.

Несмотря на большие ожидания, которые возлагались в свое время на CORBA, данный проект можно назвать неудачным. Во всяком случае, надежды, которые возлагались на данную технологию, не были оправданы [Henning, 2006]. Одна из причин провала данной технологии была чисто организационной. Различными комитетами и рабочими группами консорциума OMG вносились и принимались в некоторых случаях противоречащие друг другу архитектурные и технологические решения. При этом для принятия вносимого предложения не требовалось предъявления работающего эталонного прототипа или макета, подтверждающего эффективность предлагаемого решения. Отметим, что впоследствии ряд участников рабочих групп консорциума OMG основали компанию

ZeroC. В рамках этой компании уже на новом технологическом уровне была успешно реализована технология ICE, которую можно считать преемницей CORBA. Технология ICE заняла свою нишу. Ее распространению, в частности, способствовало то обстоятельство, что она базируется на программном обеспечении с открытым исходным кодом. Вместе с тем, широкого признания, которое предсказывали технологии CORBA, ее преемница так и не получила.

Другую причину неудачи CORBA можно найти в том, что крупные производители программного обеспечения настороженно и недоверчиво отнеслись к этой технологии. Отметим, что в то же самое время компания Microsoft продвигала свою альтернативу — технологию DCOM (Distributed Component Object Model). Эта технология, так же как и CORBA, являлась промежуточным программным обеспечением класса RPC и реализовывала схожую идею построения распределенных объектов. Если изначально DCOM ограничивался средой Windows, то впоследствии осуществлялись попытки ее портирования под операционные системы семейства Unix. Однако эта платформа так и не стала успешной альтернативой CORBA, и работы над ней были свернуты. В 2000 г. наступила эпоха больших веб-сервисов.

1.2. Большие веб-сервисы

В развитии веб-технологий принято выделять три периода. Первый период — с 1992 по 2000 г. — относится к возникновению глобальной веб-инфраструктуры и формированию ее современного вида. Завершение данного периода ознаменовалось разработкой базовых средств (фундамента) технологии, а также ее теоретического обоснования. Технологическим фундаментом стали протокол HTTP (HyperText Transfer Protocol) и спецификация URI (Uniform Resource Identifier). Теоретическим обоснованием стал архитектурный стиль REST (REpresentational State Transfer), предложенный Ройем Филдингом в своей диссертационной работе [Fielding, 2000]. Архитектурный стиль REST описывает введенный Филдингом особый класс распределенных программных систем — *сетевые системы гипермедиа* (network-based hypermedia systems), составляющими которых являются известные пользователям Интернет веб-серверы, веб-сайты, веб-браузеры.

С момента своего возникновения веб-технологии стали оказывать заметное влияние на индустрию разработки и использования промежуточного программного обеспечения. В 1998 г. (почти одновременно с CORBA) появилась технология XML-RPC. От CORBA она отличается удивительной простотой, что обеспечило ее использование и до настоящего времени.

Основные характеристики XML-RPC следующие. В качестве транспортного механизма используется протокол HTTP. Методом POST клиент передает входное сообщение на сервер и в синхронном режиме получает ответное. Для разметки сообщений используется язык XML (eXtensible Markup Language). Входные сообщения в интуитивно понятном виде задают имя удаленной процедуры и набор входных параметров, а выходные — результат выполнения удаленной процедуры. Кроме примитивных типов данных поддерживаются неименованные структуры и списки. В табл. 2 приведен пример входных и выходных сообщений XML-RPC.

Таблица 2

Пример сообщений XML-RPC

Входное сообщение	Выходное сообщение
POST /RPC2 HTTP/1.0 Host: rpc.weblogs.com Content-Type: text/xml Content-length: 250 <?xml version=»1.0»?> <methodCall> <methodName>ping</methodName> <params> <param> <value>Scripting News</value> </param> <param> <value> http://www.scripting.com/ </value> </param> </params> </methodCall>	HTTP/1.1 200 OK Content-Length: 333 Content-Type: text/xml Date: Sun, 30 Sep 2001 20:02:30 GMT <?xml version=»1.0»?> <methodResponse> <params> <param> <value> <struct> <member> <name>flerror</name> <value> <boolean>0</boolean> </value> </member> <member> <name>message</name> <value>Thanks for the ping.</value> </member> </struct> </value> </param> </params> </methodResponse>

Технология XML-RPC предлагает новую схему использования механизма RPC, отличную от ранее рассмотренной канонической схемы. Язык XML, выступающий в качестве внешнего (аппаратно и программно независимого) формата данных, благодаря своей интуитивной понятности делает излишними многие традиционные атрибуты RPC. Возникает возможность отказаться от встроенных модулей, реализующих преобразование данных из внутренних форматов во внешний и обратно, а также модулей, осуществляющих автоматическую генерацию прокси-объектов. Излишним становится использование языка описания интерфейсов, который может быть с успехом заменен набором схем XML-данных, описывающих входные и выходные сообщения.

Технологии CORBA и XML-RPC являются предшественниками больших веб-сервисов — технологии, ставшей на долги годы лидером в области промежуточного программного обеспечения. Период с 2000 по 2006 г., связанный с бурным развитием больших веб-сервисов, принято считать вторым значимым периодом развития веб-технологий.

Под большими веб-сервисами понимается реализация взаимосвязанного набора спецификаций, базовыми из которых являются следующие:

- SOAP (Simple Object Access Protocol);
- WSDL (Web Services Description Language);
- BPEL (Business Process Execution Language);
- UDDI (Universal Description Discovery and Integration).

Спецификация SOAP определяет стандартный XML-формат для представления сообщений, которыми могут обмениваться компоненты распределенной программной системы. При этом могут реализовываться два архитектурных шаблона построения распределенной системы, а именно — удаленные вызовы процедур, подробно рассмотренные в данной работе, а также *сервисная шина предприятия* (Enterprise Service Bus). В качестве транспортного механизма для передачи сообщений обычно используются протоколы HTTP или SMTP.

В табл. 3 представлены примеры SOAP-сообщений, реализующих вызов удаленной процедуры. В качестве транспорта используется протокол HTTP.

Таблица 3

Пример сообщений SOAP

Входное сообщение	Выходное сообщение
POST /weblogUpdates HTTP/1.0 Host: rpc.weblogs.com Content-Type: text/xml; SOAPAction: «/weblogUpdates» <?xml version=»1.0»?> <SOAP-ENV:Envelope ...> <SOAP-ENV:Body> <ping> <weblogname xsi:type=»xsd:string»> Scripting News </weblogname> <weblogurl xsi:type=»xsd:string»> http://www.scripting.com/ </weblogurl> </ping> </SOAP-ENV:Body> </SOAP-ENV:Envelope>	HTTP/1.1 200 OK Content-Type: text/xml; Date: Mon, 01 Oct 2001 14:09:02 GMT <?xml version=»1.0»?> <SOAP-ENV:Envelope ....> <SOAP-ENV:Body> <pingResponse> <result> <flerror xsi:type=»xsd:boolean»> 0 </flerror> <message xsi:type=»xsd:string»> Thanks for the ping. </message> </result> </pingResponse> </SOAP-ENV:Body> </SOAP-ENV:Envelope>

Изначально под веб-сервисом понимался серверный компонент распределенной системы, реализующий возможность взаимодействия с ним по протоколу SOAP. Имеет смысл сравнить представленные сообщения с сообщениями из табл. 2. Сообщения протокола SOAP более сложные, чем аналогичные сообщения протокола XML-RPC. Следует отметить, что выразительные возможности SOAP с точки зрения представления различных структур данных превосходят XML-RPC. В принципе, любую структуру данных, которая может быть определена на современном языке программирования общего назначения, можно закодировать в SOAP-сообщении. Этот факт свидетельствует о том, что изначально разработчиками предполагался автоматический режим формирования и разбора SOAP-сообщений. Для этой цели был разработан язык описания веб-сервисов WSDL.

Описание на языке WSDL можно разделить на две логические части. Первая часть представляет собой описание интерфейса, а вторая — описание самого сервиса. Изначально в спецификации WSDL для обозначения интерфейса применяли термин *тип порта* (port type). Однако впоследствии, в версии WSDL 2.0,

стал использоваться более привычный термин — интерфейс, который и будет использоваться в этой работе. В табл. 4 приведен пример описания интерфейса, функционально эквивалентный описанию из табл. 1, выполненному на языке Slice.

Таблица 4

Пример описания интерфейса на языке WSDL

Описание данных	Описание процедур
<types> <s:complexType name=»pingResult»> <s:sequence> <s:element minOccurs=»1» maxOccurs=»1» name=»flerror» type=»s:boolean»/> <s:element minOccurs=»1» maxOccurs=»1» name=»message» type=»s:string» /> </s:sequence> </s:complexType> </types>	<message name=»pingRequest»> <part name=»weblogname» type=»s:string»/> <part name=»weblogurl» type=»s:string»/> </message> <message name=»pingResponse»> <part name=»result» type=»pingResult»/> </message> <portType name=»pingPort»> <operation name=»ping»> <input message=» pingRequest»/> <output message=» pingResponse»/> </operation> </portType>

Описание интерфейса включает в себя набор определений операций, поддерживаемых интерфейсом, в том числе — имя каждой операции, определение входных и выходных сообщений, а также структур данных, на базе которых и формируются подобные сообщения. Каждый сервис может одновременно поддерживать несколько интерфейсов. При определении сервиса каждый из его интерфейсов привязывается к конкретному транспортному протоколу и сетевому адресу, по которому клиент может взаимодействовать с соответствующим интерфейсом сервиса.

Даже на основе представленного краткого фрагмента WSDL-описания можно сделать вывод, что данных язык является сложным для непосредственного использования (ручного проектирования) без привлечения сторонних средств автоматизации разработки. Одна из распространенных практик использования WSDL сводится к следующему. Инструментальные программные средства позволяют разработчику реализовать методы интерфейса веб-сервиса на каком-то языке программирования общего назначения. Эти же средства позволяют в автоматическом режиме сформировать WSDL-описание, которое может быть опубликовано или передано внешним заинтересованным лицам.

Одна из важных задач, для решения которой разрабатывалась технология больших веб-сервисов, заключается в интеграции унаследованных программных систем, которые используются в организации и были созданы в разное время, на основе разных архитектурных и технологических принципов. Фактически, веб-сервис представляет собой обертку (wrapper), представляющую

стандартизованный интерфейс (с точки зрения транспортного протокола и форматов представления данных) для взаимодействия с унаследованной системой. Таким образом, задача описания и автоматизации схем взаимодействия между интегрируемыми системами может быть сведена к аналогичной задаче для веб-сервисов. С этой целью был разработан язык BPEL, позволяющий описывать схемы взаимодействия (бизнес-процессы) веб-сервисов и оформлять подобные описания в виде композитных* веб-сервисов.

Кратко рассмотрим спецификацию UDDI. Она определяет механизм публикации описаний веб-сервисов с целью их последующего распространения и поиска всеми заинтересованными сторонами (потенциальными потребителями). Данная спецификация является еще более сложной, чем язык WSDL. В 2006 г. компании Microsoft и IBM закрыли свои публичные UDDI-сервисы. Это событие можно рассматривать как знаковое. Назрела потребность пересмотра и, возможно, частичного ухода от чрезмерно сложных, а в некоторых случаях и просто надуманных решений, которые были привнесены в индустрию промежуточного программного обеспечения в связи с появлением больших веб-сервисов.

1.3. RESTful-веб-сервисы

С момента возникновения глобальной веб-инфраструктуры ее важной составляющей стала электронная коммерция. Например, уже в 1995 г. компания Amazon запустила в эксплуатацию свой электронный магазин. Первоначально бизнес-модель электронной коммерции была следующей. Имелся один *поставщик* (продавец) товаров или услуг и по совместительству владелец электронного магазина. В качестве *потребителей* выступали пользователи, заходившие через веб-браузер на веб-сайт электронного магазина. Таким образом, *сеть распространения* товаров и услуг строилась из числа посетителей одного веб-сайта и ими же ограничивалась.

Потребность в расширении сети распространения вызвала необходимость добавления нового субъекта в существовавшую бизнес-модель электронной коммерции. Подобным субъектом стал *посредник*. Посредник владеет собственным сайтом, через который осуществляет перепродажу исходных услуг и товаров. Возникла необходимость в интеграции веб-сайтов исходного поставщика и посредника. Фактически потребовалось осуществить модернизацию веб-сайта в веб-сервис с минимальными изменениями. Создавать параллельно с веб-сайтом новую информационную систему для взаимодействия с посредником представлялось крайне нецелесообразным. В качестве идеальной рассматривалась ситуация, когда изменялся только формат сообщений, которые передаются по протоколу HTTP (например, замена HTML на XML). Теоретически, итоговые веб-сервисы должны были удовлетворять архитектурному стилю REST. Веб-сервисы, для которых это утверждение справедливо, получили название RESTful-веб-сервисов [Richardson, Ruby, 2007].

---

* Композитным веб-сервисом здесь и далее называется сервис, сценарий реализации которого предполагает автоматизированное исполнение других сервисов.

В настоящее время идея использования целевого приложения одновременно и как веб-сайта и как веб-сервиса оформилась в стандартное требование к современным программным платформам построения веб-приложений. В этой связи следует упомянуть работу [Kaufmann, Kossmann, 2009], в которой авторы реализуют новаторскую идею построения подобной платформы на базе языка манипулирования XML-данными XQuery [Кац и др., 2005]. В этой работе обозначенное требование формулируется в явном виде и архитектурно реализуется.

С 2006 г. начался новый период в развитии веб-технологий, связанный с построением распределенных программных систем на базе RESTful-веб-сервисов. Опишем основные идеи данного подхода, который можно охарактеризовать как ресурсно-ориентированный.

Программная система представляется в виде набора *ресурсов* — поименованных данных. Взаимодействие с программной системой сводится к манипулированию подобными ресурсами через *унифицированный интерфейс*, содержащий ограниченный набор типов операций. К числу основных операций относятся — создать/изменить/удалить ресурс, получить состояние ресурса, а также получить список всех ресурсов заданного типа.

В рамках RESTful-веб-сервиса для наименования ресурсов используется спецификация URI, а для реализации унифицированного интерфейса — протокол HTTP. В качестве примера рассмотрим гипотетический веб-сервис, позволяющий управлять заданиями, выполняющимися на некоторой высокопроизводительной вычислительной установке. Предположим, что данный сервис располагается по адресу http://computing.ru.

Адрес http://computing.ru/job может использоваться для наименования ресурса, содержащего список всех выполняющихся (и выполненных) вычислительных заданий. Метод GET применяется для получения текущего состояния ресурса (в данном случае — списка заданий), метод POST — для создания новых ресурсов. Например, для создания нового вычислительного задания необходимо воспользоваться методом POST по адресу http://computing.ru/job. В результате будут созданы новое задание и одновременно ресурс, позволяющий с ним работать. Для наименования этого ресурса естественно использовать адрес вида http://computing.ru/job/{job_id}, где job_id — уникальный идентификатор, присваиваемый заданию при его создании. С помощью метода GET по этому адресу будет получена информация о текущем состоянии выполнения задания. Метод PUT позволит изменить это состояние (например, приостановить его выполнение). Для уничтожения ресурса (аннулирования задания) необходимо воспользоваться методом DELETE.

Кроме базовых методов — GET, POST, PUT, DELETE — допускается использование HTTP методов HEAD и OPTIONS. Метод HEAD используется для получения метаинформации о ресурсе, OPTIONS — возвращает список допустимых операций. Например, для ресурса http://computing.ru/job — допустимыми являются методы GET, POST, в то время как для ресурса http://computing.ru/job/{job_id} — допустимы GET, PUT и DELETE.

Операции GET, HEAD должны быть *безопасными*. Их выполнение не должно повлечь за собой изменение состояния ресурса. Операции PUT, DELETE должны быть *идемпотентными*. Их повторное выполнение игнорируется.

Когда утверждается, что метод GET возвращает состояние ресурса, то это не совсем верно. Он возвращает *представление* (representation) состояния ресурса. У одного и того же ресурса может одновременно поддерживаться несколько форматов представлений (например, XHTML, XML, JSON). Установка предпочтительного формата, а также используемого по умолчанию, осуществляется с помощью параметра Content-Type в заголовке HTTP запроса.

Рекомендуется при создании RESTful-веб-сервисов реализовывать связность ресурсов, сохраняя в их представлениях ссылки на другие ресурсы. Например, естественным выглядит наличие ссылки на список заданий в представлении индивидуального задания.

1.4. Гибридные веб-сервисы

Учитывая широкую распространенность и признание протокола HTTP, не удивляет тот факт, что до настоящего времени было создано огромное число веб-приложений. Большинство из них не удовлетворяют архитектурному стилю REST. В некоторых случаях даже веб-сайты архитектурно имеют больше общих черт с технологией XML-RPC, чем со стилем REST. Подобные веб-приложения естественно назвать гибридными веб-сервисами, одновременно содержащими черты RESTful-веб-сервисов и технологии XML-RPC. В этом обобщенном разделении большие веб-сервисы естественно рассматривать как разновидность XML-RPC-веб-сервисов.

Проанализируем причины возникновения гибридов. Приведенная ранее «каноническая» схема построения RESTful-веб-сервиса не всегда может быть реализована в силу объективных технических причин. Методы протокола HTTP, отличные от GET, POST, могут просто не поддерживаться. Это происходит при использовании XHTML-представлений ресурсов. У ряда распространенных веб-серверов имеются ограничения на размер строки URI при выполнении GET-запросов. По этой причине могут возникнуть ситуации, в которых придется обходиться только методом POST.

Изначальный выбор между построением приложения в виде большого веб-сервиса или RESTful-веб-сервиса является нетривиальной задачей, о чем свидетельствуют результаты исследований, представленных в работе [Pautasso et al., 2008]. В данной работе было показано, что эти два подхода отличаются исходными задачами, для решения которых они были разработаны. Каждый из этих подходов имеет право на существование. Нередко разработчики намеренно реализуют для своей программной системы два интерфейса. Этим путем, в частности, пошла компания Amazon. Сервисы данных этой компании поддерживают интерфейс RESTful-веб-сервиса, использующий методы CET, PUT и DELETE протокола HTTP, и одновременно интерфейс большого веб-сервиса, использующий только метод POST. Таким образом в данном случае получился гибрид.

По всей видимости, в области веб-технологий будущее как раз за гибридами. Косвенным подтверждением этого факта может служить версия WSDL 2.0. Данная спецификация в явном виде позволяет привязывать интерфейс веб-сервиса к протоколу HTTP, а не только к SOAP.

2. ГРИД-СИСТЕМЫ

Как было показано, реализация основных архитектурных концепций грид приходится на период 2001–2006 гг. Это период бурного развития больших веб-сервисов, которые были положены в основу инструментальных средств построения грид-систем. В свое время это была единственная «зрелая» платформа для построения распределенных программных систем. Однако в настоящее время ситуация изменилась. Появились и завоевали популярность новые, альтернативные решения. Кроме того, произошло переосмысление самой методики применения больших веб-сервисов. В связи с изложенным ранее закономерным является вопрос: возможно ли выбрать иную, отличную от больших веб-сервисов, платформу построения грид-систем?

2.1. Открытая инфраструктура грид

В программной инженерии принято разделять этапы концептуального и логического моделирования. Проясним эти понятия на примере. Блок-схема алгоритма является его *концептуальной* моделью, в то время как реализация алгоритма в виде программы на конкретном языке программирования — его *логическая* модель. Каждой концептуальной модели может одновременно соответствовать несколько логических моделей, привязанных к конкретным средам выполнения.

В работе [Banks et al., 2004] описывается открытая инфраструктура грид-сервисов, определяющая архитектурные решения, которым должны удовлетворять все грид-сервисы. Эти решения относятся к узкому горлу песочных часов протоколов грид. Фактически они представляют собой некоторую логическую модель, описанную на языке больших веб-сервисов. Попытаемся выделить соответствующую концептуальную модель.

Все рассмотренные модели построения распределенных систем сводились к организации обмена сообщениями между ее компонентами. Подобный обмен сообщениями можно интерпретировать как выполнение запросов на сервере, инициированное клиентом. Выделяют синхронную и асинхронную модели обмена сообщениями, схематично представленные на рис. 2 (см. с. 82).

Синхронная модель подразумевает блокировку клиента на время выполнения запроса на сервере. В некоторых случаях это является сильным ограничением. Запрос может выполняться продолжительное время. Ответ может формироваться частями, и у клиента должна быть возможность получать эти части по мере их формирования. Кроме того, может потребоваться временное хранение ответа на сервере и неоднократный доступ к нему. В перечисленных случаях необходимо реализовывать асинхронную модель выполнения запросов.

В простейшем случае асинхронная модель выглядит следующим образом. После начала выполнения запроса сервер возвращает клиенту *дескриптор* (handle) временной сущности, созданной на сервере для хранения состояния выполнения запроса. Впоследствии используя этот дескриптор, клиент может выполнить проверочный запрос на сервере. Когда запрос будет выполнен, клиент с помощью дескриптора получит ответ.

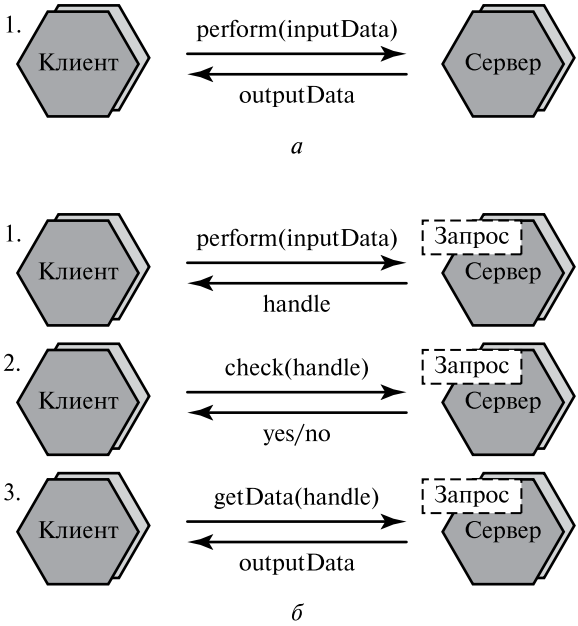


Рис. 2. Модель выполнения запросов: *а* — синхронная; *б* — асинхронная

Простейшая модель может быть дополнена механизмом уведомлений (callback). В случае завершения выполнения запроса клиент получает уведомительное сообщение. Представленные рассуждения можно считать отправной точкой для выработки концепции грид-сервисов, схематично представленной на рис. 3 (см. с. 83).

Изначально выделяются два типа грид-сервисов — *фабрики* и динамически создаваемые временные *экземпляры*. В дальнейшем под грид-сервисами будем подразумевать подобные экземпляры. На начальном этапе клиент, выбрав соответствующую фабрику, создает временный грид-сервис. После создания грид-сервису присваивается временная метка — период его существования. После завершения этого периода грид-сервис уничтожается. У клиента, при необходимости, имеется возможность увеличить этот период.

После начала выполнения запроса клиент может управлять ходом процесса и, в частности, осуществлять мониторинг. Через дескриптор грид-сервиса можно запросить и/или изменить его состояние. Он может также реагировать на входящие информационные сообщения. В этом случае грид-сервис выступает в качестве *источника* (source) событий об изменении его состояния, а клиент — *подписчика* (sink).

Другая интересная особенность грид-сервиса связана с разделением понятий дескриптор и *ссылка* (reference) на грид-сервис. Дескриптор является уникальным идентификатором грид-сервиса на протяжении всего его жизненного цикла. В то же время ссылка может изменяться с течением времени, и, более того, одновременно может поддерживаться несколько действующих ссылок. На основе дескриптора, используя соответствующий информационный сервис,

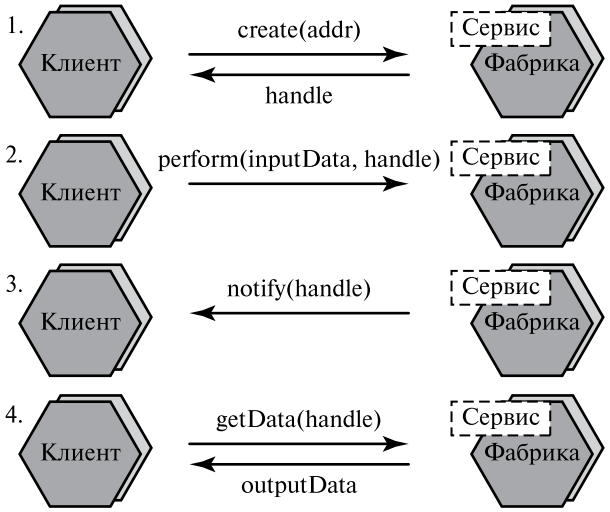


Рис. 3. Концептуальная модель грид-сервиса

клиент может получить действующую ссылку на грид-сервис. В свою очередь, используя ссылку, клиент может осуществлять непосредственное взаимодействие с грид-сервисом. Подобное, на первый взгляд усложненное решение позволяет осуществлять миграцию грид-сервиса с одной фабрики на другую, а также поддерживать их репликацию, что обеспечивает устойчивость и масштабируемость в работе грид-систем.

Подводя итог изложенному, можно сделать вывод, что концепция грид-сервиса базируется на совместной реализации следующих архитектурных шаблонов:

- фабрика и временные экземпляры объектов;
- управление жизненным циклом экземпляра объекта;
- миграция объектов;
- подписка на источник событий и уведомления.

2.2. Открытая архитектура грид

После выработки основополагающих решений в рамках создания открытой инфраструктуры грид-сервисов был осуществлен ряд практически значимых проектов по построению грид-систем. Полученный практический опыт был осмыслен и систематизирован в виде набора сценариев (вариантов использования) грид-сервисов и систем [Foster et al., 2004]. Данная работа послужила отправной точкой для создания открытой архитектуры грид-сервисов [Foster et al., 2004, 2005]. В рамках открытой архитектуры были выделены наборы грид-сервисов в соответствии с функциональными возможностями и назначением. Схематично подобное разделение представлено на рис. 4.

Первая группа, получившая название *инфраструктурных сервисов* (infrastructure services), на самом деле не является отдельными грид-сервисами. Это интерфейсы и механизмы, которые должны поддерживаться другими грид-сервисами.

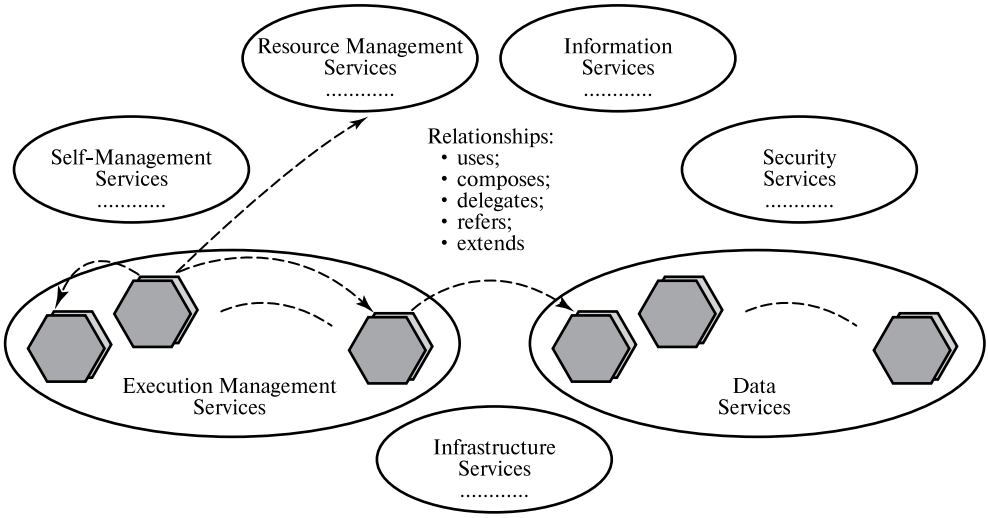


Рис. 4. Открытая архитектура грид-сервисов

Основные функциональные возможности целевых грид-систем реализуются *сервисами управления выполнением* (execution management services, EMS) и *сервисами данных* (data services). Эти два набора сервисов будут далее рассмотрены отдельно. Имеются также четыре вспомогательных набора сервисов: *безопасности* (*security services*), *информационные* (information services), *управления ресурсами* (*resource management services*), *самоуправления* (self-management services).

Отметим, что сервисы внутри выделенных наборов и между наборами функционируют не изолированно друг от друга. В рамках открытой архитектуры были выделены типовые отношения, которые могут возникать между грид-сервисами в процессе их работы. Например, было введено отношение «*передачи на рассмотрение*» (refers). Так, сервис управления выполнением перед началом своей работы может отправить запрос, полученный от клиента, сервису безопасности для валидации.

Перечислим оставшиеся отношения: «*использование*» (uses) — грид-сервис может использовать в своей работе другой грид-сервис, «*расширение*» (extends) — грид-сервис может расширять функциональные возможности другого грид-сервиса, «*делегирование*» (delegates) — грид-сервис может делегировать выполнение запроса другому грид-сервису (например, обеспечение взаимодействия между разными виртуальными организациями), «*быть композитным*» (*composes*). Естественно, что грид-сервисы одновременно могут удовлетворять нескольким из введенных отношений.

2.3. Сервисы управления выполнением

Сервисы EMS предназначены для решения всего спектра задач, касающихся вопросов запуска и управления выполнением вплоть до завершения некоторых *единиц работы* (*units of work*). При этом само понятие единицы работы имеет

широкую трактовку. Это может быть запрос к грид-сервису или выполнение некоторого стороннего приложения. Наряду с понятием единицы работы вводится понятие *точки размещения* (location) — конкретный информационно-вычислительный ресурс, который будет использован для выполнения единицы работы.

Более формально, общий сценарий работы EMS-сервисов можно разделить на такую последовательность этапов. На первом этапе осуществляется поиск кандидатов (список точек размещения), которые по формальным критериям подходят (подходящая операционная система, установлен требуемый набор библиотек и так далее) для выполнения единицы работы. Второй шаг связан с выбором конкретной точки размещения из списка кандидатов для выполнения единицы работы. Могут использоваться разнообразные оптимизационные алгоритмы для обеспечения равномерной загрузки всех имеющихся информационно-вычислительных ресурсов. Третий этап связан с проведением всех подготовительных действий перед запуском. Он может включать сборку, конфигурирование и развертывание необходимых исполняемых файлов и библиотек, а также размещение исходных данных. Четвертый этап заключается в запуске единицы работы на выполнение, пятый этап — управление ходом ее выполнения. Управление ходом выполнения может включать постоянный мониторинг, реагирование на возникающие проблемы, создание контрольных точек, принятие решения о смене точки размещения для выполнения единицы работы.

В рамках EMS предполагается выделить три класса грид-сервисов: *контейнеры ресурсов* (resources service container), *сервисы управления заданиями* (job management), *сервисы выбора ресурсов* (resource selection services). Функциональные возможности этих сервисов следуют из их названия. Проясним только понятие *задания* (*job*). С одной стороны, под заданием понимается совокупность всей необходимой информации для управления ходом выполнения единицы работы, включая текущее состояние, исходные требования к аппаратно-программному окружению, информацию по результатам мониторинга и ряд других параметров. С другой стороны, задание представляет собой интерфейс для управления выполнением единицы работы.

Необходимо отметить, что изначально в технологии грид механизмы, соответствующие контейнерам ресурсов, были хорошо проработаны. В конце 1990-х гг. был разработан протокол GRAM (Grid Resource Access and Management) [Czajkowski et al., 1998], реализация которого с самого начало входила в состав инструментальных средств Globus Toolkit [Foster, Kesselman, 1997]. Несмотря на то, что GRAM представляет собой унифицированный интерфейс загрузки заданий для широкого спектра вычислительных сред, вопросы, связанные с управлением заданиями, были проработаны в рамках технологии грид недостаточно.

Технология управления вычислительными заданиями активно развивалась в рамках проекта Condor [Thain et al., 2005]. В частности, был развит подход к управлению композитными заданиями, получивший название DAGMan. К 2002 г. была осуществлена успешная попытка соединения технологий грид и Condor, выполненная в проекте по созданию программного средства Condor-G [Frey et al., 2002]. В рамках данного проекта осуществлена интеграция протокола GRAM в систему Condor.

Открытая архитектура грид-сервисов обозначила необходимость реализовать в рамках сервисов управления заданиями и выбора ресурсов технологию, аналогичную технологии Condor. Однако эта задача не была решена.

2.4. Сервисы данных

Открытая архитектура грид-сервисов декларирует необходимость использования разнообразных типов источников данных, включая коллекции плоских файлов, потоки, системы управления базами данных (реляционные, XML, объектно-ориентированные). Выделяются также сценарии работы с данными, включая удаленный доступ, временное размещение, репликацию, федеративные представления, создание представлений.

Изначально в рамках технологии грид был разработан ряд эффективных, правда «низкоуровневых», механизмов работы с данными, примером которых может служить GridFTP [Allcock et al., 2000]. Однако потребности в работе с данными, обозначенные в рамках открытой архитектуры грид-сервисов, удовлетворены не были. Исключением может служить проект OGSA DAI [Antonioletti et al., 2005], в рамках которого был реализован сценарий построения федеративных представлений над источниками данных. Следует отметить, что в настоящее время этот проект позиционируется отдельно от инструментальных средств грид.

2.5. Композитные сервисы

В рамках открытой архитектуры декларируется необходимость использования и построения композитных грид-сервисов. Под композитным понимается грид-сервис, реализующий программный механизм автоматизации сценария (схемы) взаимодействия грид-сервисов. Говорят также, что подобный программный механизм, необязательно оформленный в виде грид-сервиса, осуществляет их оркестровку.

В связи с необходимостью использования композитных грид-сервисов возникает ряд интересных вопросов. Архитектура и технология грид-сервисов базируется на некотором промежуточном программном обеспечении, лежащем в их основе. В настоящее время эту роль выполняют большие веб-сервисы. В рамках технологии больших веб-сервисов был создан и апробирован упоминавшийся ранее язык BPEL. Можно ли утверждать, что язык BPEL решает задачу построения композитного грид-сервиса? Ответ будет отрицательный. Данный язык позволит осуществить оркестровку грид-сервисов, результатом которой будет просто веб-сервис, по умолчанию не поддерживающий необходимых грид-интерфейсов.

Вопрос о целесообразности использования языка BPEL в рамках технологий грид остается открытым [Deelman et al., 2005; Zhao et al., 2008]. После внедрения спецификации WSDL 2.0 получен положительный ответ на другой вопрос: принятие решения о замене в рамках грид больших веб-сервисов на технологию RESTful-веб-сервисов не означает отказа от потенциальной возможности использования языка BPEL [Pautasso, 2008].

3. ОБЛАЧНЫЕ ВЫЧИСЛЕНИЯ

Начиная с 2006 г. стал развиваться новый подход к построению распределенных программных систем, на основе использования внешних информационных и вычислительных ресурсов. Подход получил название *облачные вычисления* (cloud computing). Учитывая схожесть задач, решаемых грид-системами, и задач, которые решаются в рамках облачных вычислений, подробнее остановимся на данном подходе. С этой целью проанализируем структуру и принципы реализации сервисов данных [Murty, 2008] компании Amazon, а также бизнес-модель их предоставления потребителям.

3.1. Сервисы данных компании Amazon

Базовые сервисы данных компании Amazon получили название S3 (Simple Storage Service). Схема их использования представлена на рис. 5. Эти сервисы используют очень простую модель данных, состоящую только из двух сущностей: *корзина* (в оригинале используется слово bucket) и *объект*. Объекты — это некоторые поименованные данные и метаинформация о них, а корзина представляет собой контейнер объектов. Имеется прямая аналогия с файловой системой: корзина соответствует директории, а объект — файлу. Отметим, что в рамках модели поддерживается только один уровень вложенности. Корзина не может содержать другие корзины. Однако это ограничение легко обходится, если самостоятельно придерживаться специальной нотации имен объектов, имитирующей вложенные друг в друга директории.

Пользователю предоставляется несколько вариантов взаимодействия с сервисом: через свободно распространяемые программные библиотеки, реализующие клиента сервиса в объектно-ориентированном стиле, через интерфейс RESTful-веб-сервиса и через интерфейс большого веб-сервиса. Реализация

Рис. 5. Сервисы данных в облачных вычислениях

RESTful-веб-сервиса несколько отличается от «канонической» и включает использование четырех методов протокола: HTTP — GET, HEAD, PUT, DELETE. Метод PUT одновременно используется и для создания, и для модификации соответствующего ресурса. Метод POST задействован только в реализации интерфейса большого веб-сервиса, а также используется для загрузки данных через веб-браузер.

Компания Amazon использует территориально-распределенную сеть дата-центров для репликации данных, обеспечивая таким образом их надежное хранение и высокую доступность. Выход из строя одного из дата-центров не принесет ущерба самим данным, а также не скажется на работе «внешних» программных систем, взаимодействующих с сервисами S3. Однако платой за такое решение является отказ от *строгой непротиворечивости* (strict consistency) данных в пользу *потенциальной непротиворечивости* (eventual consistency) [Таненбаум, Стеен, 2003]. Этот факт означает, что произведенные изменения данных будут «видны» пользователю в той же последовательности, в которой производились соответствующие операции изменения, однако не обязательно сразу. Как правило, изменения вступают в силу в течение одной секунды, в то же время никаких гарантий на этот счет не существует.

Сервисы S3 являются эффективным инструментом, положительно зарекомендовавшим себя во многих практически важных приложениях. Необходимо отметить, что на основе S3 были созданы более высокоуровневые сервисы, реализующие функциональные возможности систем управления базами данных. При этом подобные работы были проделаны как в самой компании Amazon, выпустившей сервис SimpleDB [Habeeb, 2010], так и сторонними разработчиками [Kraska et al., 2008].

3.2. Сравнение технологии облачных вычислений и грид-систем

Одной из задач [Foster et al., 2004], которая традиционно относится к технологии грид, является построение коммерческих дата-центров. Как было показано, данная задача нашла красивое решение в рамках другой технологии — технологии облачных вычислений.

Ранее был обозначен ряд сценариев работы с данными, которые являются важными с позиций открытой архитектуры грид-сервисов. Сценарии репликации данных, их временного размещения, а также использования структурированных и неструктурированных данных реализованы в сервисах компании Amazon в виде, идеологически близком технологии грид. Кроме того, реализована идея построения высокоуровневых сервисов, использующих для своей работы низкоуровневые сервисы и расширяющих их функциональные возможности.

Имеются ряд различий между технологиями грид и облачных вычислений. Эти отличия прежде всего касаются бизнес-моделей, лежащих в основе их использования. Технология грид направлена на построение в рамках межкорпоративного взаимодействия интегрированных программных систем. В то же время в основе облачных вычислений лежит целенаправленное накопление информационно-вычислительных ресурсов с целью предоставления их для использования максимально широкой аудиторией потребителей в виде стандартизованного

перечня услуг. Однако в основе этих технологий лежат схожие идеи. Это, прежде всего, касается использования временных программных сервисов, обладающих состоянием. Если продолжительность существования грид-сервиса задается при его создании и впоследствии может увеличиваться пользователем, то время жизни сервисов данных Amazon определяется периодически вносимой оплатой за их использование.

В связи с изложенными соображениями закономерным является вопрос: в чем причина успеха облачных вычислений и относительной неудачи технологии грид? Первое, что бросается в глаза, — сервисы данных компании Amazon не так жестко привязаны к технологии используемого промежуточного программного обеспечения. Одновременно поддерживается несколько API, между которыми можно переключаться в соответствии с собственными предпочтениями. Второе, на что следует обратить внимание, — в архитектуре сервисов Amazon отсутствуют «лишние» детали. Их архитектура усложняется постепенно, и ее сложность зависит исключительно от сложности (нетривиальности) реализуемой услуги.

В случае технологии грид все обстоит иначе. Каждый грид-сервис, независимо от сложности и важности предоставляемой им услуги, должен поддерживать базовый набор механизмов, соответствующий узкому горлу песочных часов протоколов грид. И это является сознательным решением основателей данной технологии [Foster et al., 2001]. Как видно, в некоторых случаях это может стать причиной задержки в развитии технологии или, во всяком случае, одного из ее направлений.

ЗАКЛЮЧЕНИЕ

В заключение приведем ряд выводов.

- Инструментальные средства построения грид-систем должны базироваться на некотором промежуточном программном обеспечении. В период 2001–2006 гг. альтернативы большим веб-сервисам в качестве промежуточного программного обеспечения не было.
- Формулировка основных концептуальных и архитектурных решений технологии грид приходится на период 2001–2006 гг. Говоря инженерным языком, в этот период было осуществлено эскизно-техническое проектирование данной технологии, результатом которого стала разработка открытой инфраструктуры и архитектуры грид-сервисов. При этом в качестве базового инструмента были выбраны большие веб-сервисы.
- После 2006 г. начался новый период в развитии веб-технологий. Он ознаменовался определенной стагнацией технологии больших веб-сервисов, пересмотром методики их использования и де-факто отказом от некоторых избыточных решений. Этот период связан с возникновением подхода к построению распределенных систем на базе RESTful-веб-сервисов.
- После 2006 г. возникла и начала активно развиваться технология облачных вычислений. В рамках облачных вычислений были предложены решения задач, традиционно относящихся к технологии грид. Другой отличительной чертой облачных вычислений стало использование гибридных веб-сервисов.

• Архитектурные решения, изложенные в открытой инфраструктуре и архитектуре грид-сервисов на языке больших веб-сервисов, жестко не привязаны к последним. В этой связи актуальным представляется проведение работ, направленных на исследование возможности построения грид-систем на базе гибридных веб-сервисов, и использования в них решений, выработанных в рамках технологии облачных вычислений.

ЛИТЕРАТУРА

[Кац и др., 2005] *Кац Г.* и др. W3C XML: XQuery от экспертов. Руководство по языку запросов. М.: Кудиц-образ, 2005. 480 с.

[Таненбаум, Стеен, 2003] *Таненбаум Э.*, *ван Стеен М.* Распределенные системы. Принципы и парадигмы. СПб.: Питер, 2003. 877 с.

[Allcock et al., 2000] *Allcock W.*, *Chervenak A.*, *Foster I.*, *Kesselman C.*, *Tuecke S.* Protocols and services for distributed data-intensive science // Proc. Advanced Computing and Analysis Techniques in Physics Research. 2000. P. 161–163.

[Antonioletti et al., 2005] *Antonioletti M.* et al. The design and implementation of Grid database services in OGSA-DAI // Concurrency and Computation: Practice and Experience. 2005. V. 17. N. 2. P. 357–376.

[Banks et al., 2004] *Banks T.* et al. Open Grid Service Infrastructure Primer // Global Grid Forum. 2004.

[Czajkowski et al., 1998] *Czajkowski K.* et al. A resource management architecture for metacomputing systems // Proc. IPPS/SPDP Workshop on Job Scheduling Strategies for Parallel Processing. 1998. P. 62–82.

[Deelman et al., 2005] *Deelman E.* et al. Pegasus: a Framework for Mapping Complex Scientific Workflows onto Distributed Systems // Scientific Programming J. 2005. V. 13. N. 3. P. 219-237.

[Emmerich et al., 2007] *Emmerich W.*, *Aoyama M.*, *Sventek J.* The impact of research on middleware technology // ACM SIGSOFT Software Engineering Notes 2007. V. 32. N. 1. P. 89-112.

[Fielding, 2000] *Fielding R. T.* Architectural Styles and the Design of Network-based Software Architectures: Dissertation doctor of philosophy in Information and Computer Science. Univ. of California, 2000.

[Foster, Kesselman, 1997] *Foster I.*, *Kesselman C.* Globus: A metacomputing intrastructure toolkit // Intern. J. Supercomputer Applications. 1997. V. 11. N. 2. P. 115–128.

[Foster et al., 2001] *Foster I.*, *Kesselman C.*, *Tuecke S.* The Anatomy of the Grid: Enabling Scalable Virtual Organizations // Intern. J. Supercomputer Applications. 2001. V. 15. N. 3. P. 200–222.

[Foster et al., 2002] *Foster I.*, *Kesselman C.*, *Nick J.*, *Tuecke S.* The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration // Global Grid Forum. 2002.

[Foster et al., 2004] *Foster I.*, *Gannon D.*, *Kishimoto H.*, *von Reich J.* Open Grid Services Architecture Use Cases // Global Grid Forum. 2004.

[Foster et al., 2005] *Foster I.* et al. The Open Grid Services Architecture. Version 1.0 // Global Grid Forum. 2005.

[Foster et al., 2006] *Foster I.* et al. The Open Grid Services Architecture. Version 1.5 // Global Grid Forum. 2006.

[Frey et al., 2002] *Frey J.*, *Tannenbaum T.*, *Foster I.*, *Livny M.*, *Tuecke S.* Condor-G: A computation management agent for multi-institutional grids // Cluster Computing. 2002. V. 5. P. 237–246.

[Habeeb, 2010] *Habeeb M.* A Developer's Guide to Amazon SimpleDB. Addison-Wesley, 2010. 1st ed. P. 288.

[Henning, 2006] *Henning M.* The Rise and Fall of CORBA // ACM Queue. 2006. V. 4. N. 5. P. 28–34.

[Henning, Spruiell, 2010] *Henning M., Spruiell M.* Distributed Programming with Ice. Revision 3.4 // Distributed Computing with Ice. ZeroC Inc. 2010.

[Kaufmann, Kossmann, 2009] *Kaufmann M., Kossmann D.* Developing an Enterprise Web Application in XQuery // ICWE 2009. LNCS. 2009. V. 5648 P. 465–468.

[Kraska et al., 2008] *Kraska T., Brantner M., Florescu D., Graf D., Kossmann D.* Building a Database on S3 // Proc. 2008 ACM SIGMOD Intern. Conf. Management of Data. P. 251–263.

[Murty, 2008] *Murty J.* Programming Amazon Web Services: S3, EC2, SQS, FPS and SimpleDB. O'Reilly Media, 2008. 1 ed. 608 p.

[Pautasso, 2008] *Pautasso C.* BPEL for REST // Proc. 6th Intern. Conf. Business Process Management. 2008. P. 278–293.

[Pautasso et al., 2008] *Pautasso C., Zimmermann O., Leymann F.* RESTful Web Services vs. Big Web Services: Making the Right Architectural Decision // Proc. 17th Intern. World Wide Web Conf. 2008. P. 805–814.

[Richardson, Ruby, 2007] *Richardson L., Ruby S.* RESTful Web Services. O'Reilly, 2007. 448 p.

[Thain et al., 2005] *Thain D., Tannenbaum T., Livny M.* Distributed Computing in Practice: the Condor Experience // Concurrency and Computation: Practice and Experience. 2005. V. 17 N. 2–4. P. 323–356.

[Tuecke et al., 2003] *Tuecke S.* et al. Open Grid Services Infrastructure (OGSI). Version 1.0 // Global Grid Forum Draft Recommendation. 2003.

[Zhao et al., 2008] *Zhao Y., Raicu I., Foster I.* Scientific Workflow Systems for 21st Century e-Science, New Bottle or New Wine? // IEEE Workshop on Scientific Workflows. 2008. P. 467–471.

ARCHITECTURAL AND TECHNOLOGICAL ASPECTS EVOLUTION OF THE GRID

V. A. Vasenin, A. S. Shundeev

Institute of Mechanics, Lomonosov Moscow State University, Moscow

Architecture and concept of GRID was developed in the period 2001-2006. This period, associated with rapid technological development of Big Web services, laid the foundation architecture of the GRID. After 2006, a number of alternative approaches to the construction of distributed software systems (such as RESTful Web services) and to solve problems that traditionally belong to the field of GRID (cloud computing) emerged. In this regard, current studies are aimed at analysis of the history and prospects of the GRID technology. The results of such a work are presented in this article.

Keywords: grid, cloud computing, middleware.

Vasenin Valery Alexandrovich — head of the laboratory, doctor of physical and mathematical sciences, e-mail: vasenin@msu.ru.

Shundeev Alexander Sergeevich — leading scientist, candidate of physical and mathematical sciences, e-mail: shundeev@msu.ru.

УДК 89.01.11

КОНЦЕПТУАЛЬНЫЕ ПОДХОДЫ К СОЗДАНИЮ ПЕРСПЕКТИВНЫХ КОСМИЧЕСКИХ СИСТЕМ

Алексей А. Романов, Александр А. Романов, Ю. М. Урличич, А. Е. Буравин

Открытое акционерное общество «Российская корпорация ракетно-космического приборостроения» (ОАО «Российские космические системы»), Москва

В обзорной статье излагаются основные проблемные вопросы, возникающие при проектировании космических систем на основе существующих технологий создания спутников. Анализируются текущее состояние и перспективные тренды в развитии базовых элементов. Рассматривается подход, предложенный Агентством передовых оборонных исследовательских проектов США DARPA (Defense Advanced Research Projects Agency), для преодоления возникших на сегодняшний день проблем, заключающийся в переходе от монолитных спутников к кластерам малоразмерных космических аппаратов с распределенным функционалом больших аппаратов на основе программы System F6. Рассматриваются предложения французского космического агентства CNES по принципиально новой архитектуре спутниковой системы дистанционного зондирования Земли e-CORCE.

Ключевые слова: спутниковые системы, дистанционное зондирование, малоразмерные космические аппараты, проектирование космических систем, проблемные вопросы.

ВВЕДЕНИЕ

В последние годы в связи с очевидным прогрессом, достигнутым в ряде наукоемких отраслей (микроэлектроника, мехатроника, робототехника, связь, навигация и др.), являющихся основными поставщиками базовых элементов и технологий для мировой ракетно-космической индустрии, значительно сократились сроки постановки продукции на серийное производство. В работе [Sweeting, 2007] показано, что темп достижения качественно нового уровня характеристик создаваемой космической аппаратуры (минимальный размер пространственного разрешения на поверхности Земли, скорость передачи информации по радиолиниям «борт–Земля», объем памяти бортовых запоминающих устройств и др.) подчиняется закону Мура, так же как нарастание количества элементов в электронных интегральных схемах. На рис. 1–3 (см. с. 93, 94)приведены графики изменения некоторых из них за последние 25...30 лет.

Наряду с этим, нормативные сроки создания базовых элементов для спутниковых информационных систем связи, навигации и дистанционного зондирования Земли (ДЗЗ) не меняются и часто превышают 10 лет. С учетом срока

Романов Алексей Александрович — заместитель генерального директора – генерального конструктора, доктор технических наук, профессор.

Романов Александр Алексеевич — начальник центра, доктор технических наук, e-mail: romulas@rniikp.ru.

Урличич Юрий Матэвич — генеральный директор – генеральный конструктор, доктор технических наук, профессор.

Буравин Андрей Евгеньевич — заместитель генерального директора.

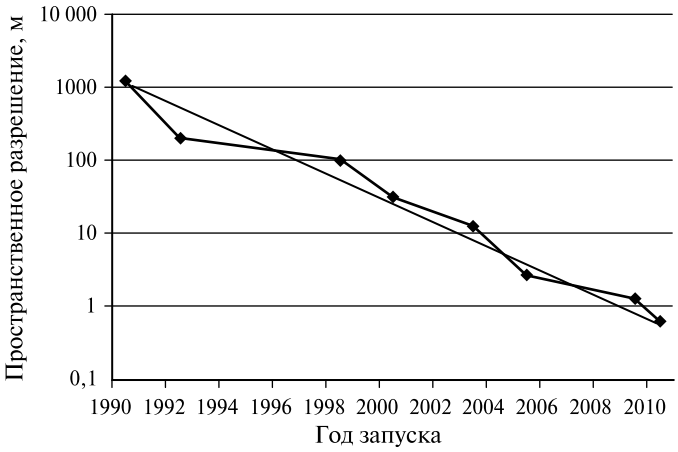


Рис. 1. Уменьшение пространственного разрешения аппаратуры

активного существования перечисленных комплексов около 10 лет указанное обстоятельство часто приводит к тому, что создаваемые космические средства морально устаревают еще на этапе проектирования.

Целью данной работы было проведение анализа современных подходов, позволяющих выйти на качественно новые показатели проектирования спутниковых информационных систем при сохранении, а иногда и превышении достигнутого уровня потребительских характеристик.

МАЛОРАЗМЕРНЫЕ КОСМИЧЕСКИЕ АППАРАТЫ — ПРОРЫВНАЯ ТЕХНОЛОГИЯ, МЕНЯЮЩАЯ МИРОВУЮ ЭКОНОМИКУ КОСМИЧЕСКОЙ ДЕЯТЕЛЬНОСТИ

Запуском первого искусственного спутника Земли 4 октября 1957 г. была открыта новая эра развития человечества, которая привела к появлению огромного числа прорывных технологий, позволяющих успешно решать многие вопросы, связанные с освоением околоземного космического пространства, а также и вполне земные задачи социально-экономического развития. Масса первого космического аппарата (КА) не превышала 100 кг и на тот момент времени это было обусловлено возможностями существовавшей космической техники. Однако по мере ее совершенствования масса выводимых на орбиту КА, а также сложность решаемых задач непрерывно нарастали и в 70–80 гг. прошлого столетия масса спутников стала превышать 1000 кг.

В первую очередь это было обусловлено тем, что используемые оптические материалы, размер апертуры оптических элементов, элементная база целевой аппаратуры, а также служебных систем, включая системы энергообеспечения и ориентации, имели значительные массогабаритные характеристики и большое энергопотребление. Требование обеспечения высокой надежности космической техники приводило к необходимости дублирования критических узлов бортовой

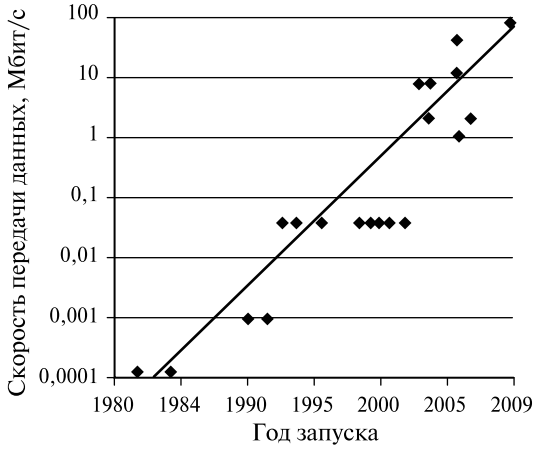


Рис. 2. Увеличение скорости передачи информации по радиолиниям «борт – Земля»

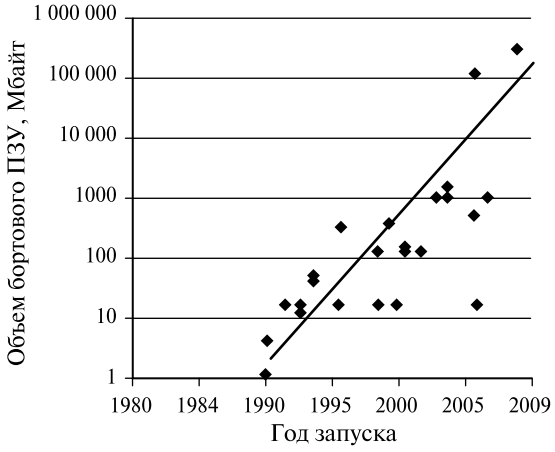


Рис. 3. Увеличение объема бортового ОЗУ

аппаратуры, что в свою очередь повышало ее сложность. В конечном итоге перечисленные причины и длительные сроки разработки привели к тому, что, по выражению Оуэна Брауна (Owen Brown) — менеджера программ Агентства передовых оборонных исследовательских проектов США DARPA, главный недостаток существующих больших КА состоит в попытке решать задачи завтрашнего дня с помощью вчерашних технологий [Magnuson, 2007].

С целью сокращения сроков разработки современных КА несколько небольших частных компаний попытались создать малоразмерные КА с использованием существующих на рынке микроминиатюрных устройств различного назначения. За прошедшие 20 лет эволюционного развития размеры и массога-

баритные характеристики КА претерпели значительные изменения в сторону уменьшения.

В табл. 1 приведена существующая градация КА [Sweeting, 2007].

Таблица 1

КА	Характеристики		
	масса, кг	цена, млн евро	время создания, годы
Большие	>1000	>300	>10
Малые	<1000	100	3...5
Мини	500	30	2
Микро	50	10	1,5
Нано	1...10	1	1
Пико	0,1	>0,1	~1

Однако сегодня данная таблица уже может быть дополнена новой категорией сверхмалых КА, так называемых фемптоспутников, массой от 10 до 100 г. На их основе создаются «спутники на кристалле» или «спутники в интегральной микросхеме» [Barnhart et al., 2007].

В работе основоположника малоразмерного спутникостроения профессора Сюррейского университета Мартина Свитинга [Sweeting, 2007] показано, что эффективность действующих КА определяется набором таких его характеристик как масса, время создания, стоимость и функциональные возможности.

Также убедительно продемонстрировано, что в области оптического ДЗЗ достигнутые характеристики малоразмерных КА практически уже ни в чем не уступают характеристикам больших КА. Более того, уже создана первая коммерческая система оптического дистанционного зондирования Rapid Eye на основе пяти малых КА с элементом пространственного разрешения на земной поверхности, ранее достижимым только военными разведывательными спутниками. Кроме того, разработана и успешно функционирует группировка оптических КА в интересах мониторинга чрезвычайных ситуаций DMC (Disaster Monitoring Constellation). Эволюция данной системы на ближайшую перспективу приведена на рис. 4 (см. с. 96).

В работе [Baker et al., 2009] предложена концепция малоразмерного КА, где полезной нагрузкой является радар с синтезированной апертурой. Планируемая дата запуска в 2015 г. позволяет рассчитывать, что в скором времени малоразмерные КА смогут занять значимую нишу в классе спутников круглосуточного всепогодного наблюдения с высоким пространственным разрешением. Причем все перечисленные КА относились к классу малых КА с массой около 100 кг. К числу наиболее важных достижений в указанных разработках следует отнести существенное сокращение сроков проектирования и создания спутников (около 3 лет), что позволяет обеспечить применение в разработках самых современных комплектующих. Кроме того, с использованием коммерчески доступных критических элементов конструкции КА существенно уменьшаются кооперация организаций, участвующих в его создании, и инфраструктура, необходимая для

проведения монтажных и испытательных работ. По существу, функция создания КА превращается в функцию создания электронного прибора, выполняющего роль полезной нагрузки, а сам спутник превращается в «прибор в космосе».

Очень интересна и тенденция отношения крупных предприятий мировой космической индустрии к данному направлению деятельности. Если с самого начала малоразмерное спутникостроение было в основном уделом технических университетов и малых коммерческих фирм, то сегодня в этом участвуют практически все основные игроки данного рынка: Boeing, Lokheed Martin, Orbital Science и EADS [Hoey, 2005]. Так EADS поглотила компанию SSTL путем присоединения.

В начале 2000 г. DARPA объявила конкурс на создание программы университетских наноспутников различного целевого назначения, в рамках которого были поддержаны предложения ряда ведущих университетов США. При этом главная задача состояла в том, чтобы предложить принципиально новый механизм создания дешевых малоразмерных КА в короткие сроки из коммерчески доступных элементов силами студентов и аспирантов. Именно в рамках этой программы профессором Стэнфордского университета Бобом Твиггсом [Twiggs, 1998] был предложен стандарт пикоспутников, названный CubeSat, унифицирующий основные элементы конструкции и служебных подсистем.

На протяжении последних пяти лет спутники сверхмалого класса (общей массой до 10 кг) из средства обучения студентов в технологических университетах основам проектирования КА, а также управления проектами в сфере косми-

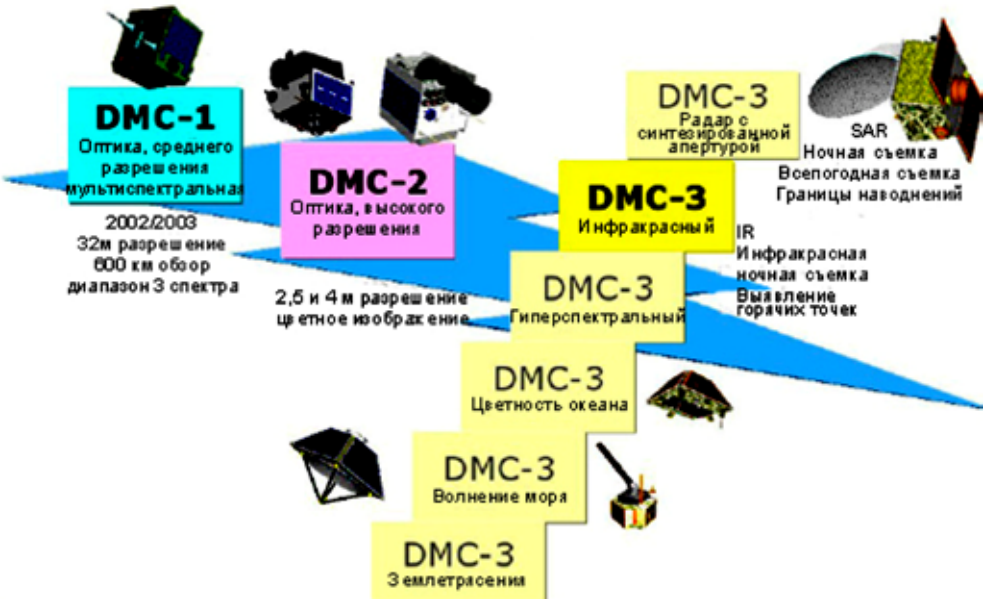


Рис. 4. Модифицирование спутников DMC

ческой деятельности превратились в мощный инструмент решения различных целевых задач.

В 2005 г. был запущен первый в России КА нанокласса ТНС-0, разработанный в ОАО «Российские космические системы» и предназначенный для отработки принципиально новой концепции управления КА [Urlichich et al., 2007]. В настоящее время на предприятии разработано целое семейство космических платформ нанокласса для решения спектра различных задач: дистанционного зондирования (ТНС-1), геофизического мониторинга ионосферы, автоматической идентификационной системы судов (ТНС-0 № 2) и др. [Urlichich et al., 2008].

На базе наноспутников развивается множество проектов, направленных на решение геофизических задач, в частности на изучение геомагнитного и электрического полей (CubeSTAR, Firebird, RAX), мониторинг электромагнитных предвестников землетрясений (QuakeSat). Появилась информация о создании наноспутников, основным предназначением которых является решение задач ДЗЗ с высоким пространственным разрешением от 1,5 до 6 м (KestrelEye, MISC-1), передачи текстовых сообщений (SMDC), мониторинга подвижных объектов и т. д. [London III et al., 2010].

Анализ развития рынка показывает, что мировое космическое приборостроение выходит на следующий виток развития, направленный на миниатюризацию полезной нагрузки и спутниковых платформ для решения традиционных задач, стоящих перед космической отраслью.

Катализатором данного процесса стало появление на коммерческом рынке стандарта КА сверхмалого класса — CubeSat, а также основных элементов спутниковой платформы этого стандарта (каркас платформы двух типоразмеров, бортовой компьютер, аккумуляторные батареи с контроллером питания, солнечные батареи, система ориентации и управления и пр.). Срок поставки компонент платформы CubeSat составляет 6...8 недель без учета сроков проведения таможенной очистки, а стоимость максимальной номенклатуры комплектации — около 120 000 евро без учета таможенных сборов.

В 2009 г. стали коммерчески доступны платформы спутников сверхмалого класса (MISC-1, -2), совместимые со стандартом CubeSat 3U (10×10×30 см), поставляемые на заказ и обеспечивающие средние энергетические характеристики из расчета 20...30 Вт за виток. Стоимость подобной платформы составляет около 300 000 дол. США, срок поставки — 12 месяцев без учета таможенных сборов и сроков растаможивания.

Появление коммерчески доступной стандартизированной спутниковой платформы позволяет сформулировать новую концепцию космического приборостроения — «спутник-прибор», делая ее реализацию коммерчески привлекательной.

В условиях громоздкой системы проектного управления, сложившейся в отрасли, системный инжиниринг целевых систем «под ключ» на базе аппаратов малого и сверхмалого классов мог бы стать одной из основных парадигм дальнейшей деятельности приборостроительных предприятий. В рамках данной магистральной линии можно выделить два наиболее перспективных направления развития.

Первое направление состоит в разработке и серийном изготовлении микроминиатюрных полезных нагрузок, обеспечивающих требуемые характеристики решения целевой задачи при условии малого энергопотребления: сканер видимого и инфракрасного диапазонов, ретранслятор автоматической идентификационной системы судов (АИС), специализированные приборы геофизического назначения и пр. *Но, что более важно для выхода на коммерческий рынок, разрабатываемая полезная нагрузка должна быть жестко унифицирована с платформой CubeSat по массогабаритным характеристикам и электрическим интерфейсам.*

Второе направление состоит в разработке специального программного обеспечения, включая программы управления отдельными элементами платформы спутника, наземного сегмента управления кластером КА, приема и обработки информации или решения всей целевой задачи (например, решения томографической задачи мониторинга ионосферы). Разрабатывая специальные программные продукты, можно создавать практически произвольную добавленную стоимость к стандартным компонентам платформы, доступным на коммерческом рынке. В качестве примера можно привести подход к ценообразованию своих продуктов компании IntelliTech Microsystems: комплект микроминиатюрных магнитных катушек можно заказать за 20 000 евро, тогда как система ориентации и управления на их базе стоит 45 000 евро. Основным отличием этих продуктов является программа управления гироскопами, позволяющая упростить процесс реализации алгоритмов управления спутниковой платформой.

Сравнительно небольшая стоимость спутников сверхмалого класса, а также сравнительно короткие сроки их разработки и изготовления позволяют решать многие задачи сегодняшнего дня при помощи технологий дня завтрашнего. Уже сегодня предполагается оперативное развертывание группировок сверхмалых спутников для решения на протяжении небольшого периода времени определенных задач видовой разведки или обеспечения передачи коротких сообщений. Примером такой системы может стать система наноспутников SMDC (*USASMDC/ARSTRAT — программа министерства обороны США, направленная на демонстрацию технологий наноспутников тактической связи*). Указанная программа включает разработку и запуск 10 спутников SMDC-ONE массой 4 кг, предназначенных для получения летной квалификации. Запуск первого КА размером 10×10×32 см состоялся в 2010 г. Предлагаемые КА предназначены для сбора данных от наземных центров управления и контроля и передачи их в штабы. Срок создания не превысил одного года. Стоимость менее 1 млн дол. за один спутник [Hawkins, 2011].

Запуск группировок и кластеров сверхмалых КА позволяет существенно повысить оперативность наблюдения земной поверхности с увеличением частоты просмотра одного и того же участка, а также в перспективе преодолеть фундаментальное физическое ограничение — размер оптической апертуры прибора наблюдения путем синтеза апертур с нескольких КА в кластере.

Следовательно, микроминиатюрный КА, кластер или система малых спутников как целостная система, решающая конкретную целевую задачу, из категории штучного, а потому не выгодного для разработки товара, переходит в категорию мелкосерийного изделия. Данное обстоятельство открывает возможность эффективной коммерциализации конкурентоспособных тематических разработок.

ИНФОРМАЦИОННЫЙ СЕТЕВОЙ ОБМЕН В КОСМОСЕ — ПУТЬ К ВИРТУАЛЬНОМУ КОСМИЧЕСКОМУ АППАРАТУ

В 2007 г. тактический технологический офис DARPA объявил конкурс на концепцию создания системы F6 (Future, Fast, Flexible, Fractionated, Free Flying Spacecraft united by Information eXchange (System F6)) [DARPA, 2010]. В переводе на русский язык название системы звучит следующим образом: перспективная, быстроформируемая, гибкая, разделяемая на части система свободно летающих КА, объединенных посредством информационного обмена. Данная концепция предполагает разделение единого монолитного КА с интегрированным набором различных функций на кластер независимых малоразмерных КА, каждый из которых в отдельности выполняет какую-либо функцию из этого набора. Главное достоинство предлагаемого подхода состоит в том, что надежность системы в целом значительно возрастает, поскольку в случае выхода из строя какого-либо из КА его функция может быть мгновенно продублирована любым КА из состава кластера. Кроме того, становится возможной оперативная замена вышедшего из строя модуля, что ведет к существенному увеличению срока активного существования системы в целом. В подобной системе возможно, наконец, перейти и к оперативной замене программного обеспечения, поскольку распределенный кластер КА представляет собой аналог наземной локальной вычислительной сети, каждый узел которой и является малоразмерным КА. Еще одним весьма ценным качеством является то, что при реализации данного подхода сроки формирования системы не попадают в жесткую зависимость от задержки с созданием какой-либо специфической полезной нагрузки. Сдвиг сроков ее готовности не приводит к задержке развертывания системы, поскольку отсутствующий элемент может быть запущен позже по мере готовности.

В 2008 г. DARPA провела конкурс на создание концепции, который выиграла компания Orbital Science. В октябре 2010 г. был объявлен новый конкурс на создание ключевых элементов System F6. Следует отметить, что DARPA поменяла парадигму, в соответствии с которой в конкурсах могли принимать участие только компании-резиденты США. Теперь в конкурсе могут участвовать любые международные компании. При этом, как и ранее, общий срок реализации выигранного конкурса не может превышать трех лет и проект должен обязательно заканчиваться действующим образцом, получившим летную квалификацию в реальном космическом полете.

Создав систему F6, DARPA собирается продемонстрировать возможность декомпозиции большого монолитного спутника на группу элементов или узлов, соединенных каналами беспроводной связи. Каждый из узлов выполняет специфическую функцию большого КА, например, один может быть компьютерным узлом, в то время как другой — передатчиком или «умным» датчиком. Эти узлы, функционируя совместно, создают единый распределенный «виртуальный» КА.

Разделяя тяжелый монолитный космический аппарат на части в соответствии с функционалом, можно увеличить надежность полученной распределенной системы, а также внедрять новые гибкие принципы проектирования и сборки. Например, возможно физически изолировать полезные нагрузки от остальных частей системы.

Технические принципы реализации System F6 включают сетевую организацию, беспроводной обмен данными и энергией, распределенные полезные нагрузки и вычисления, кластерную архитектуру и разумную эконометрику. Сетевая организация представляет собой самоорганизующуюся, надежную, легкодоступную, устойчивую и защищенную сеть узлов с уникальными адресами. Беспроводные связь и обмен энергией обеспечивают совместную эксплуатацию модулей, поддержание конфиденциальности, интеграцию и электромагнитную совместимость, внутри- и внемодульную передачу энергии, а также ориентацию по Солнцу. Распределенные полезные нагрузки и вычисления формируют независимые требования к полезным нагрузкам и распределенный вычислительный ресурс, включенный в составные модули. Кластерная архитектура поддерживает автономную совместную работу, «сближение» и поддержание межспутниковых расстояний во избежание столкновений, а также маневрирование при угрозах. Эконометрика обеспечивается рискоустойчивой методологией конструирования, основанной на достижении значимого результата и правильном учете, что приводит в конечном итоге к структурной гибкости и надежности.

Реализация подобного подхода позволит, по мнению специалистов DARPA, достигнуть принципиально нового уровня требований к проектированию, развертыванию и эксплуатации космических информационных систем, в первую очередь военного назначения. Для таких требований уже придуман новый термин — спутник по запросу. При этом на этапе проектирования время, затрачиваемое на разработку, создание и испытания космической техники, сокращается с сегодняшних 2...10 лет до 6...9 месяцев. Время, необходимое для установки на ракетоноситель (РН), запуска и развертывания космической системы с сегодняшних 3...12 месяцев сокращается до нескольких часов от момента получения запроса на поддержку (в случае готовности инфраструктуры космодрома и РН). На этапе эксплуатации время, необходимое для доставки информационного продукта конечному потребителю, сегодня в зависимости от назначения системы обычно составляет от нескольких часов до нескольких дней. По требованиям оперативно адаптируемых космических средств (Operative Response Space — ORS) результаты применения космических систем в ближайшей перспективе должны быть доступны практически в реальном времени в соответствии с темпом изменения ситуации (непрерывно/секунды) [Hoey, 2005].

ИНФОРМАЦИОННАЯ ТЕХНОЛОГИЯ E-CORCE — НОВЫЙ ПОДХОД К ПОЛУЧЕНИЮ ДАННЫХ ДЗЗ

Безусловно, основное применение малоразмерных КА в ближайшее время ожидается в создании оперативных группировок ДЗЗ оптического наблюдения, позволяющих ускорить процесс коммерциализации рынка ДЗЗ. К сожалению, существующая инфраструктура получения данных ДЗЗ высокого пространственного разрешения, за исключением военных применений, не позволяет осуществлять просмотр любой заданной территории на земной поверхности в реальном масштабе времени. Это обусловлено, в первую очередь, высокой стоимостью подобных КА, а также существующими ограниченными возможностями по сбо-

ру и распространению данных. В работе [Antikidis, Favier, 2008] предложено поменять парадигму организации системы ДЗЗ. По аналогии с развитием системы персональной мобильной связи и позиционирования GPS/ГЛОНАСС отмечается, что существенное снижение цены предоставляемой услуги возможно только после освоения массового рынка.

Понятно, что рынок ДЗЗ сегодня находится на начальной стадии профессионального использования данных ДЗЗ. Для того чтобы эти данные получили максимальное количество потребителей, необходимо, чтобы производство продукции ДЗЗ стало действительно массовым. Именно такая идея и была положена в основу системы e-CORCE (electronic — Continuous Observing system Relayed by Cellular processing Environment), в которой предполагается получать изображения всего земного шара с разрешением 1 м в течение 1 дня. При этом "Continuous Observing" означает «непрерывно обновляемые снимки поверхности Земли», а "Relayed by Cellular processing Environment" относится к тому, что концепция системы основана на тесном взаимодействии трех сотовых слоев среды обработки получаемых данных: космического, телекоммуникационного и сетевого (Grid). По существу данный подход означает усовершенствование сервиса Google Earth «я хочу увидеть любое место на земной поверхности в любое время» путем привнесения в него оперативной динамики обновления данных съемки земной поверхности.

Таким образом, в дополнение к возможности выбора любого элемента Google Earth в статике возникает его динамическая составляющая.

Три полностью интегрированных технических уровня, обеспечивающих единую концепцию e-CORCE, представляют собой:

- **космический сотовый уровень,** который обеспечивает использование полностью автоматизированной спутниковой группировки оптического наблюдения, постоянно снимающей всю земную поверхность в течение 1 дня с разрешением 1 м (предполагается наличие 100 или более КА);
- **телекоммуникационный сотовый уровень:** спутники постоянно передают сильно сжатый (например, в формате jpeg) поток отснятой информации в наземные центры; телекоммуникационные операции будут автоматизированы и предельно упрощены, что полностью устраняет участие оператора — человека на приемной станции; применение IP-адресации создает возможность организации общего канала передачи данных для сигналов управления и телеметрии;
- **наземный сотовый уровень Wide Area Grid (WAG),** из-за огромного количества информации, собираемой ежедневно, вся обработка будет полностью распределена по всей земной поверхности, обеспечивая приемлемое локальное покрытие данными для каждой наземной станции приема; грид-технология способна обеспечить подобную организацию распределенной обработки, учитывая то, что земная мозаика никогда не будет централизованно объединяться, приводя к созданию «Виртуального глобуса».

Основное достоинство предлагаемого подхода состоит в том, что для распространения конечного информационного продукта не требуется никакого распределения конечных потребителей или головного центра, нигде не создается законченная мозаика изображения земной поверхности. При этом конечный продукт распределен логически, но он никогда физически не возникает,

поскольку доступен как виртуальный продукт IP-Интернет или функционально эмулируется.

По оценкам авторов [Antikidis, Al., 2008] необходимая общая компьютерная мощность составляет 5000 персональных компьютеров (PC, из них 1000 — в 2012 г.), распределенных по 50...100 приемно-обрабатывающим центрам, способным генерировать на ежедневной основе глобальную мозаику с разрешением 1 м. Любой узел WAG будет использовать до 20 PC для создания своей локальной мозаики.

Поскольку вся Земля постоянно доступна для обзора, не требуется программирование работы съемочной оптической аппаратуры, она просто работает непрерывно.

За счет того, что изображение всей Земли постоянно доступно как «Виртуальный глобус», не нужен каталог метаданных, что сильно упрощает управление данными.

Вся система полностью автоматизирована, идеология ее работы сильно напоминает сервис коммерческой платформы магазина Amazon.

Хранение данных осуществляется каждым потребителем (для хронологии) и только интересных ему данных, а не системой e-CORCE. Это обусловлено в первую очередь тем, что данные постоянно обновляются и соответственно новые изображения дешевле старых.

Перечисленные достоинства подхода приводят к тому, что стоимость конечной информационной продукции уменьшается на четверть по сравнению с существующими ценами, обеспечивая в случае необходимости новые применения.

ЗАКЛЮЧЕНИЕ

В результате приведенного анализа были сформулированы перспективные направления развития мировой ракетно-космической отрасли. Бесспорно, малоразмерные КА в настоящее время не способны заменить собой большие и сложные системы, но определенные шаги и смещение тренда в сторону упрощения и миниатюризации КА наблюдаются в литературе уже сейчас.

Можно выделить следующие предпосылки для изменения парадигмы создания космических систем:

1. Закон Мура прогнозирует для малых КА быстрое уменьшение стоимости и времени создания с регулярным обновлением широко доступных элементов при постоянном наращивании функциональных возможностей, а это значит, что основу базовых элементов перспективных космических систем будут обеспечивать *коммерчески доступные технологии*.

2. Меняется концепция самого КА — теперь это «*прибор в космосе*» или «*спутник на кристалле/на печатной плате*».

3. «*Созвездия*» и «*Рои*», т.е. кластеры малоразмерных КА, дают реальные преимущества при создании новых космических систем на базе «виртуальных» КА (локальную компьютерную сеть в космосе) с возможностями большого КА.

4. *Малоразмерные КА* становятся основным направлением развития космического компонента наблюдения системы литосфера – атмосфера – ионосфера, а в скором времени и ДЗЗ.

В чем-то схожий подход предлагается использовать для построения перспективных систем дистанционного зондирования, рассмотренных на примере e-CORCE. Децентрализация системы приведет к упрощению каждого базового элемента, необходимости развертывания серийного производства, снижению их себестоимости, а следовательно, снижению стоимости самой информации. В перспективе информация ДЗЗ реального времени может предоставляться на условно бесплатной основе, тогда как операционную прибыль компания-оператор будет извлекать из сопутствующих источников. Подобную модель бизнеса успешно демонстрирует и развивает в последнее время корпорация Google.

Безусловно, рассмотренные концептуальные вопросы не являются четко сформулированной программой действий. Более того, многие аспекты могут быть подвергнуты серьезной критике специалистами. Однако нельзя не отметить и привлекательности предлагаемых подходов, поскольку, на первый взгляд, их реализация может действительно привести к прорыву при создании перспективных космических информационных систем.

ЛИТЕРАТУРА

[Antikidis, Al., 2008] *Antikidis J. P.*, *Al.* (July-August 2008). The one meter shop concept e-CORCE // Acta Astronautica. 2008. V. 63. Iss. 1–4. P. 156–164.

[Antikidis, Favier, 2008] *Antikidis J.-P.*, *Favier J.-J.* 2008.

[Baker et al., 2009] *Baker A.*, *Abbott B.*, *Whittaker P.*, *Bird R.*, *Bird R.*, *Curiel Alex da Silva*. Modular radar and optical constellation supporting commercial arctic operations // Intern. Astronautical Congress. Daejon, South Korea. 2009.

[Barnhart et al., 2007] *Barnhart D. J.*, *Vladimirova T.*, *Sweeting M. N.* Very-Small-Satellite Design for Distributed Space Missions // J. Spacecraft and Rockets. Nov.-Dec. 2007. V. 44. N. 6. P. 1294–1306.

[DARPA, 2010] DARPA. System F6. Broad Agency Announcement. 20 Oct. 2010. P. 1–49.

[Hawkins, 2011] *Hawkins K.* (2011). Flying into orbit in small package. Redstone Rocket. 26 Jan. 2011.

[Hoey, 2005] *Hoey Matthew*. Symposium on Non-proliferation and Disarmament — The Way Forward // Military Space Systems: the Road Ahead. 2005. V. 10.

[London III et al., 2010] *London III J. R.*, *Marley A. B.*, *Weeks D. J.* (2010). Army Nanosatellite Technology Demonstrations for the Tactical Land Warfighter. USASMDC/ARSTRAT Public Release, Sept. 2010. P. 1–8. [Электрон. текст]. Режим доступа: http://www.armyscience-conference.com/manuscripts/B/BP-013.pdf.

[Magnuson, 2007] *Magnuson S.* Scientists Pursue Flexible, Adaptable Space Systems // National Defense Magazin. 2007. V. 10.

[Selivanov et al., 2008] *Selivanov A.*, *Romanov A.*, *Vishnyakov V.*, *Vinogradov A.*, *Pavelyev A.*, *Yakovlev O.* Concept of space system for global radio occultation monitoring of lower atmosphere and ionosphere based on super-small satellites with GLONASS/GPS navigation signal receivers: научн. докл. [Международ. симп. United Nations / Austria / European Space Agency Symposium on Space Applications “Space Tools and Solutions for Monitoring the Atmosphere and Land Cover”. Graz, Austria, 9–12 Sept. 2008].

[Sweeting, 2007] *Sweeting M.* Moor’s Law and Small satellites // Proc. 7th IAA Simp. Small Satellites. Springer, 2007.

[Twiggs, 1998] *Twiggs R.* AMSAT-NA 16th Space Symp. Nanosatellite Program — A Challenge to AMSAT for Collaboration to Use the Amateur Bands. Vicksburg, MS. Oct. 16, 1998. P. 1–4.

[Urlichich et al., 2007] *Urlichich Yu.*, *Selivanov A.*, *Vishnyakov V.*, *Petushkov A.*, *Sergeev S.* Application of telecommunication system GLOBALSTAR for nanosatellite control // SP-648. Proc: 7th Intern. Symp. Reducing the Costs of Spacecraft Ground Systems and Operations (RCSGSO). 11–15 June 2007, Moscow.

THE CONCEPT APPROACH TO PERSPECTIVE SPACE SYSTEMS DEVELOPMENT

A. Romanov, A. Romanov, U. Urlichich, A. Buravin

Joint Stock Company "Russian Space Systems"

In the article the basic problem questions of the space satellites system development were considered. The current state and the perspective trends in the basics element advance have been analyzed. The approach for a modern problem solving in the field of transferring from the big and heavy satellites to cluster of microsatellites with distributed functionality had been proposed by DARPA in the framework of System F6 program was under a consideration. The proposals of CNES of the brand new architecture for the remote sensing satellite system were considered.

Keywords: satellite systems, remote sensing, small-sized satellites, space systems development, problem questions.

Romanov Алексей Александрович — assistant to the general director – general designer, doctor of technical sciences, professor.

Romanov Александр Алексеевич — chief of the center, doctor of technical sciences, e-mail: romulas@rniikp.ru.

Urlichich Юрий Матэвич — general director – general designer, doctor of technical sciences, professor.

Buravin Андрей Евгеньевич — assistant to the general director.

УДК 004.92+004.94

ЧТО ТАКОЕ СПО

В. П. Иванников

Институт системного программирования РАН, Москва

В статье* дается обобщенная характеристика свободного программного обеспечения (СПО) и обсуждаются вопросы его применения в различных областях деятельности в России.

Ключевые слова: программная индустрия, свободное программное обеспечение.

Прежде чем начать говорить о СПО, хотелось бы привести некоторые факты: как вообще возникли программное обеспечение (ПО), программная индустрия и как программа вдруг превратилась в продукт, в товар. Вроде бы это такая творческая область, где есть место фантазии, и вдруг такая прозаическая вещь как товар. Дело в том, что программы, конечно, возникли тогда же, когда в 1940–1950-х гг. появились первые компьютеры, но индустрии программного обеспечения, как самостоятельной отрасли, не существовало. В Советском Союзе ее вообще не было, т. е. всегда поставлялись компьютеры с программным обеспечением. Но, конечно, в научном сообществе накапливались огромные библиотеки, например ЦЕРН. И как-то незаметно эта индустрия возникла. Конечно, существует всегда много разных интерпретаций того, как это произошло, но многие связывают появление программной индустрии со знаменитой системой 360 IBM. Многие помнят, что в СССР было такое политическое решение во многих отраслях микроэлектроники, в компьютерах и т. д., как копирование. И были линии ЕС ЭВМ — это как раз система 360. Система 360 вышла в 1965–1967-х гг. и возникла очень сложная ситуация для всех других производителей вычислительной техники, таких как General Electric, Control Data Corporation и т. д., потому что было сделано приложение по управлению данными IMS, знаменитый монстр языков PL1, очень много приложений. Остальные производители были поставлены в очень тяжелые конкурентные условия. IBM захватило около 75 % рынка. В 1967 гг. Control Data Corporation обратилась в Министерство юстиции США с запросом о нарушении IBM антитрестового законодательства, что грозило IBM расчленением на компании, как, например, произошло с American Telephone & Telegraph Company. Разбирательство Министерства юстиции тянулось около двух лет, но уже в 1968 г. в IBM возникла идея о разделе продукции, с тем, чтобы отдельно продавать аппаратуру и программы, т. е. программа становится товаром, и отдельно — услуги, связанные с сопровождением ПО. IBM создала специальную Task Force — рабочую группу из 100 человек на full time, которая продумывала в течение года эту бизнес-модель, ценовую политику и т. д.

Иванников Виктор Петрович — директор, академик, профессор, e-mail: ivan@ispras.ru.

* Публикуемая статья представляет собой переведенную в текст аудиозапись доклада «Что такое СПО», сделанного директором института системного программирования В. П. Иванниковым на настоящем семинаре. Переведено в текст М. В. Григорьевой. Статья дана в основном в авторской редакции.

В 1969 г. IBM открыла архитектуру с системой 360, т. е. идея заключалась в том, что выпускается ряд машин разной производительности с почти одним и тем же ПО, и с совместимостью, хотя это не очень получилось. Фирма объявила о разделе продукции и тем самым на рынке появилась возможность независимым производителям разрабатывать программы. Нужно сказать, что те, кто покупал технику IBM или арендовал ее, были не очень довольны, потому что суммарная стоимость возросла. Были установлены достаточно высокие планки на стоимость программы независимо от числа копий — одна или миллион, но, тем не менее, лицензия за каждую копию стоит достаточно дорого. Именно так возникли такие монстры как Microsoft, Oracle. Все это производные от продуктов, которые первоначально были разработаны в IBM.

До этого было принято распространять программы с открытым кодом, например, в течение длительного времени выходил журнал «Алгоритмы», где масса программ была в открытых кодах. Это обычная практика для научного сообщества, Кнут, например, написал TeX открытым кодом. Закрытие кода создало определенные неудобства для научного и образовательного сообщества. Но появились разработки так называемого Free Software или Open Source полюса, т. е. с открытым кодом, например, FreeBSD (Unix, Berkley) и проект GNU (его начал Ричард Столман, основатель Free Software Foundation, он написал достаточно интересные и большие программы, вошедшие в этот проект, например, компилятор GCC, редактор EMACS), которые были особенно популярны в научной студенческой среде. Возникла целая философия, связанная с открытым кодом. Одновременно начали возникать некоторые юридические основания и документы, регламентирующие использование этого нового СПО. Возникли две основные правовые ветви: идущая от Berkley и от Ричарда Столмана. Первая ветвь имеет четыре степени свободы по отношению к коду:

- можно его использовать;
- распространять;
- изучать;
- модифицировать, что-то добавлять, изменять.

Ограничение второй ветви: если использовать взятый от сообщества свободный код и что-то туда добавлять, необходимо точно так же вернуть это сообществу. Это линия GNU General Public License (GPL). Другая линия, Berkley, говорит о том, что возможно внесение каких-то изменений, затем закрытие кода, распространение его только в бинарниках и взимание платы за лицензию. В открытом коде лицензии бесплатны. Сейчас много разных вариантов: MIT-лицензия, Apache-лицензия. Они немного отличаются, но есть два принципиальных момента — возможность превращения открытого кода в коммерческий продукт, проприетарный с закрытым кодом, либо невозможность этого. Когда возникают комбинации свободного софта, который выпущен по разного рода лицензиям, появляются юридические проблемы, связанные с тем, как их в дальнейшем использовать.

Расскажу о личном опыте с открытым софтом. В 1990 г. я познакомился с Ричардом Столманом по электронной почте, которая в это время возникла, получил все вещи, которые были сделаны в GNU, — отладчик, компилятор и т. д.

В то время я работал с Владимиром Андреевичем Мельниковым. Традиционно, инженеры делали машины, а мы — программное обеспечение для них, системное ПО. Я с удивлением узнал, что на СПО можно сделать какую-то модель бизнеса. В то время этим занималась только одна компания Sygnus, президентом которой был Майкл Тимман. С ним я встретился 1990 г. на конференции в Стенфорде. В ходе долгой беседы с использованием документации он пытался пояснить модель бизнеса. Компьютерных компаний в Калифорнии много, и иметь лабораторию, которая сопровождает или консультирует по GCC или отладчику, не имело смысла, потому что это дорого, поэтому компании заключали с Sygnus достаточно дешевые (по нынешним временам) контракты по 100 тыс. дол. Они имели около 10...15 контрактов в год, около десяти молодых сотрудников — один ходил по конторам, другой заключал договоры, а все остальные были программисты. Потом Sygnus создал несколько подразделений в Европе, в Торонто. В них работало около 100 программистов. Очень интересно, как позиционировали эти услуги. Сначала в розницу (сопровождение отдельной программы), потом потребовалось определить направленность компании, ее специализацию. Сейчас очень часто повторяется термин «Стек программных продуктов», т. е. дается некоторое решение, некоторая цепочка SDK (Software Development Kit), можно откомпилировать, отлаживать, редактировать — все это разные инструменты. Была еще система управления версиями — когда много человек работают и вносят много изменений — согласованное управление версиями. Потом эта компания была куплена очень крупной компанией RedHat.

Рассмотрим соотношение разработчиков и прочего персонала в софтверных компаниях. В общей сложности, в компании RedHat около 2000 сотрудников, из них разработчиков-программистов около 250. Это нормальное соотношение в софтверных компаниях, где 10% людей занимаются разработкой, а остальные — администрация и огромное подразделение маркетинга и продаж, которое занимается распространением ПО. Год тому назад на сайте RedHat были опубликованы бухгалтерские отчеты. Бизнес заключался в продаже услуг по сопровождению, идея Майкла Тиммана, т. е. предлагают стек. Они предлагают Linux (GNU Linux), хотя товарный знак Linux'а зарегистрирован на персональное имя Линуса Торвальдса. Но в этот стек входит еще система управления базами данных, Postgres и MySQL, не говоря уже о наборе компиляторов GCC, десять разных языков, полтора десятка разных платформ, несколько сотен драйверов на все виды внешних устройств — это все идет под Linux — две базы данных, JBoss — система поддержки документооборота, виртуальная машина. Есть несколько градаций, плата за сопровождение от 1500 до 5000 дол. Все зависит от того, как быстро вам отвечают, как быстро исправляют и т. д.

Это значит, что в группе из 250 человек есть эксперты по ядру Linux, по библиотекам Linux, GCC, Postgres и MySQL и т. д. Ядро Linux — это 10 млн строк, т. е. уровень этих экспертов очень высок. Ошибок в ПО на самом деле очень много, и, когда пользователь присылает обращение о каких-то неполадках, нужно искать и исправлять ошибки. Есть разработчики какой-то конкретной библиотеки и изменения должны быть посланы туда. Если ошибка найдена в ядре Linux, то для включения исправлений в мейнстрим Linux нужно договориться, чтобы этот патч взял Линус Торвальдс. Но вначале нужно эти исправления

выполнить, для чего и нужны эксперты, очень квалифицированные в каждой компоненте.

Есть еще один очень интересный источник доходов — RedHat. Судя по всему, СПО развиваться без финансирования со стороны мощных корпораций, типа IBM, Intel, в принципе, не может. Сопровождением не удается компенсировать все затраты компании, особенно на разработку, создание чего-то нового. Например, IBM потребовалось Real Time Linux, не жесткий Real Time (во всех крупных корпорациях, кроме Microsoft, площадка одна и та же — Linux, на мобильных, серверах, рабочих станциях, везде, где угодно), эту работу фирма IBM оплачивала RedHat.

Наш собственный опыт: проблема процессора Itanium — наличие очень большого числа регистров. Чтобы одна инструкция не ожидала другую, планируют код (планируют перестановки) — это называется scheduling. Это обычно происходит на линейных участках, т.е. там, где нет перехода, а поскольку очень много регистров и распределение идет не только на линейном участке, но и захватывает эти переходы, то планирование и распределение регистров становится достаточно сложным. Используется подход спекулятивного распределения, т.е. оцениваются вероятности перехода. Это можно выполнить в динамике, пропуская какие-то тестовые примеры, но статика лучше, например, если у вас стоит if x=0 then, очевидно, что по этому then вы никогда не пойдете, вероятность равна нулю. Такого рода эвристики дают возможность подсчитывать вероятности и в соответствии с ними распределять регистры. Эту работу финансировал Hewlett Packard, и патчи были приняты в мейнстрим, т.е. по-существу HP оплачивало развитие GCC.

Что сейчас происходит? Как правило, университетские или академические сообщества создают тот или иной код, разного рода приложения, большая корпорация Sun (сейчас не существует, ее купила Oracle) выпустила Open Office, на который уже принят наш российский стандарт. Microsoft его поддерживает. Это открытый код, который Sun поддерживает и разрабатывает. Еще один пример — это среда для разработки Eclipse, которую разрабатывает IBM. Это открытый проект, в который стекается масса новых идей, потому что разработчики по контракту или по собственной инициативе привносят новое и расширяют эту среду, тем самым постоянно ее развивая.

В мире сейчас очень сильно активизировалось движение открытого кода среди корпораций, и не последнюю роль здесь играет третья, новая модель бизнеса, так называемые «облачные вычисления», когда программа, в частности, есть сервис. Если человек платит за какую-то услугу, то ему все равно, какая она: проприетарная или открытый код. У Google есть система Map Reduce для обработки распределенных данных, на которой можно подвешивать некоторые свои собственные функции, для того чтобы обрабатывать распределенные данные. Yahoo купила компанию Hadoop, которая разрабатывает открытый код Map Reduce. Более того, Yahoo сейчас объявила о том, что она открывает все свои продукты. В том числе, у них есть своя линия Elastic Cloude, т.е. в Amazon можно запрашивать неограниченное число виртуальных машин и на них проводить вычисления. Очень много открытых пакетов, связанных, например, с математическим моделированием.

В нашей стране ситуация сложная. У нас есть квалифицированные программисты, но ландшафт очень большой. Например, Abby или Kaspersky делают очень интересные вещи, 1С Бухгалтерия насытила внутренний рынок. Но приблизительно на 2...3 млрд мы аутсорсим в России западные компании, которые потом на 5 млрд нам продают свой софт. Например, эти прикладные пакеты стоят несколько миллионов. долларов лицензии в год и повторить это не так просто, потому что, если код содержит 4 млн строк, а производительность труда хорошего программиста 15 тыс. строк, то возникает больше 200 человеко-лет. Проблема заключается еще и в том, что должна быть команда экспертов. Нужно создать проект, а это очень сложно. Недавно, вышло постановление правительства № 2299, подписанное В. В. Путиным, о тотальном переходе госучреждений и подведомственных им организаций на СПО. По существу, этим постановлением создан рынок программ в России, т. е. по крайней мере госучреждения и подведомственные им организации вынуждены будут использовать открытый код. Готово ли сообщество программистов? Есть несколько компаний: Alt Linux, компании в Санкт-Петербурге, Рейман купил Mandriv, еще один такой дистрибутив, но нет той экспертизы, которая существует в RadHat или в SuS.

В заключение хочу сказать о проблемах с Open Source:

• *Качество*. Например, число ошибок в MS Windows — 15 на 1000 строк кода. Это после того как все отлажено, но потом идет так называемый процесс тестирования и число ошибок снижается в 10 раз. Стоимость тестирования заложена в себестоимость продукта, который выпускает Microsoft, и составляет, по разным оценкам, от 50 до 70 %. Соотношение разработчиков и тестеров в Microsoft — 1:1,5. Это огромная работа. Интересно, что в Linux, в среднем, 7 ошибок на 1000 строк кода.

• *Сопровождаемость*. Должны быть эксперты высочайшего класса.

WHAT IS IT, OPEN SOURCE?

Victor P. Ivannikov

Institute for system programming of RAS

The paper gives description of first steps of software industry. There is discussing of two business models of software production: proprietary and open source ones, and also the importance open source for Russia in education, research and development.

Keywords: software industry, open source

Ivannikov Victor Petrovich — academician, professor, the director of ISP RAS, e-mail: ivan@ispras.ru.

УДК 004.89:004.4

МУЛЬТИАГЕНТНЫЕ ТЕХНОЛОГИИ ДЛЯ УПРАВЛЕНИЯ РАСПРЕДЕЛЕНИЕМ ПРОИЗВОДСТВЕННЫХ РЕСУРСОВ В РЕАЛЬНОМ ВРЕМЕНИ

П. О. Скобелев, А. В. Иващенко, М. В. Андреев, И. О. Бабанин

Научно-производственная компания «Разумные решения», Москва

В работе дается описание современных разработок в области автоматизации управления распределением производственных ресурсов в режиме реального времени на основе мультиагентных технологий, выполненных в научно-производственной компании «Разумные решения» в рамках ряда работ по созданию и внедрению мультиагентных систем на машиностроительных предприятиях. Рассматриваются новые принципы построения интеллектуальных систем поддержки принятия согласованных решений, основанные на аналогии с природными механизмами самоорганизации, и описываются основные преимущества, получаемые при практическом применении таких решений.

Ключевые слова: мультиагентные технологии, онтологии, сложные системы, производственное планирование, машиностроение

ВВЕДЕНИЕ

Управление современным производственным предприятием является сложной задачей, связанной с необходимостью обеспечения требуемого уровня качества как выпускаемой продукции, так и производственных процессов, постоянного совершенствования производства, применения современных методов оперативного планирования, основанных на использовании актуальной информации. В настоящее время существует много разнообразных автоматизированных систем, предназначенных для планирования производственных ресурсов [Загидуллин, 2005; Леньшин и др., 2003]. Однако современные требования по обеспечению адаптивного распределения ресурсов, сложность выявления единых критериев планирования для разных заказов обусловливают необходимость поиска новых способов решения данной задачи.

В частности, следует специально отметить тенденцию по построению систем автоматизированного управления производственными ресурсами с помощью технологий, основанных на имитации природных механизмов самоорганизации [Leitão, Restivo, 2008; Leitão, 2009]. В основе этих современных разработок лежат принципы построения децентрализованных сетецентрических самоорганизующихся систем, в которых задается логика функционирования отдельных элементов и обеспечивается взаимодействие между ними [Виттих, Скобелев, 2009].

Скобелев Петр Олегович — доктор технических наук, ведущий научный сотрудник, e-mail: petr.skobelev@gmail.com.

Иващенко Антон Владимирович — кандидат технических наук, доцент, руководитель направления, e-mail: anton.ivashenko@gmail.com.

Андреев Михаил Владимирович — руководитель проектов, e-mail: michael.v.andreev@gmail.com.

Бабанин Иван Олегович — ведущий разработчик, e-mail: babanin@gmail.com.

Такой подход оправдывает себя в случаях, когда логика взаимодействия участников процесса планирования и распределения ресурсов сложна для описания, а поступающие события постоянно приводят к небольшим изменениям условий принятия решений, что не позволяет использовать классические оптимизационные методы. В этом смысле, применение новых технологий, базирующихся на знаниях и аналогиях с природными механизмами эволюции, не противопоставляется классическим алгоритмам решения задач поддержки принятия решений, но дополняет их, позволяя строить адаптивные системы управления ресурсами, способные функционировать в условиях неопределенности и высокой динамики изменения условий принятия решений.

Одним из таких новых способов, получивших, однако, в последнее время достаточно широкое распространение, является применение мультиагентных технологий [Андреев и др., 2010*а*, *б*], позволяющих автоматизировать распределение производственных ресурсов и оперативное построение расписаний и создать основу для последующей оптимизации, контроля и развития производства.

1. ПРОБЛЕМА ОРГАНИЗАЦИИ УПРАВЛЕНИЯ ПРОИЗВОДСТВЕННЫМИ РЕСУРСАМИ В РЕАЛЬНОМ ВРЕМЕНИ

Особенности управления современным производством обусловлены его сложностью, наукоемкостью, повышенными требованиями к изготовляемой продукции, а также большими объемами мелкосерийных заказов. Производственные подразделения, как и специализированные предприятия, в рамках кооперации тесно взаимосвязаны между собой, что приводит к необходимости организации согласованного и координированного управления производством в режиме реального времени.

В частности, можно указать следующие особенности организации современного производства, определяющие необходимость разработки и внедрения новых принципов автоматизированного планирования ресурсов:

- высокая сложность технологий производства;
- необходимость принятия решений в режиме реального времени;
- наличие конфликтных интересов между различными участниками производственных процессов;
- необходимость индивидуального подхода к каждому заказу и ресурсу;
- поддержка постоянных инноваций и изменений;
- ориентация на мелкосерийное производство;
- разнообразие изделий, станков и квалификаций рабочих;
- необходимость ручной доводки производственных планов с учетом специфических особенностей производства;
- сочетание стадий планирования и исполнения плана;
- необходимость экономного использования ресурсов и обеспечения конкурентоспособности продукции.

Для того чтобы удовлетворить всем этим требованиям, необходимо автоматизировать процессы планирования, составления производственного расписания

и оперативного управления ресурсами. Это нужно, в первую очередь, для обеспечения своевременной и адекватной реакции на события (например, на поступление нового заказа, задержку изготовления деталей и сборочных единиц, поломку или ремонт станка и т. д.), планирования использования производственных ресурсов и распределения новых заказов по имеющимся мощностям, управления исполнением сгенерированных планов.

Производственное расписание требует согласования планов загрузки ресурсов между различными участниками процесса как на этапе подготовки производства, когда определяется планируемая загрузка участков, так и на этапе выполнения заказов, когда в ответ на поступающие данные о возникающих событиях должна проводиться корректировка расписания. Основной задачей оперативного планирования производства является организация слаженной работы всех участков и цехов для обеспечения равномерного, ритмичного выпуска продукции в установленном объеме и номенклатуре при максимально полном использовании производственных ресурсов.

Для этого руководством предприятия определяются плановые технико-экономические показатели на определенный период, и цеха должны их выполнять. В этом смысле автоматизированная система должна позволять определять эти показатели адекватно имеющимся возможностям производства и проводить мониторинг их исполнения в реальном времени, чтобы обеспечить своевременные мероприятия, направленные на выполнение плана. В подобных условиях применение классических методов планирования и оптимизации ресурсов чаще всего оказывается неприемлемым, так как изменение условий задачи, например, связанных с приходом новых событий, происходит до того, как сам оптимум будет найден по причине сложности и трудоемкости вычислений.

Решить поставленную задачу можно лишь при использовании новых подходов к построению автоматизированных систем поддержки согласованного принятия решений, обеспечивающих оперативное распределение ресурсов в реальном времени.

2. ОСОБЕННОСТИ ПРИМЕНЕНИЯ МУЛЬТИАГЕНТНЫХ ТЕХНОЛОГИЙ ДЛЯ ПЛАНИРОВАНИЯ ПРОИЗВОДСТВЕННЫХ РЕСУРСОВ

В отличие от классических ERP и MES-систем, в мультиагентных системах каждое предприятие или подразделение предприятия моделируется как динамическая сеть агентов потребностей и возможностей [Виттих, Скобелев, 2003]. В такой сети могут быть представлены различные подразделения, конкретные производственные заказы (на готовое изделие или его компоненты, отдельную операцию станка и т. д.) и ресурсы (например, рабочие, детали или станки).

Важным преимуществом этой технологии в планировании и оптимизации ресурсов является возможность адаптивного построения и исполнения планов в реальном времени, когда план не строится всякий раз заново при возникновении новых событий, а только корректируется по мере их появления. Такая адаптация осуществляется непрерывно путем выявления и разрешения конфликтов в расписаниях, проведения переговоров и достижения компромиссов

между агентами заказов и ресурсов. Другие преимущества такого подхода — возможность решения сложных задач управления ресурсами, например, планирования связанных расписаний, большая открытость, гибкость и оперативность, производительность и живучесть создаваемых систем, находящих все большее применение в самых разных сферах. Переход к работе в режиме реального времени позволяет повысить качество обслуживания клиентов, сократить затраты и уменьшить время производства изделий, снизить риски и возможные потери.

Применение мультиагентных технологий позволяет обеспечить оперативную реакцию на возникающие непредвиденные события и непрерывную оптимизацию работ. Система работает в интерактивном режиме, под управлением пользователями. Интерактивный характер взаимодействия подразумевает, что диалог с системой может быть инициирован каждым из участников взаимодействия в любой момент времени, что позволит адаптивно, т. е. без полного перепланирования «с нуля», достраивать или перестраивать производственное расписание по мере возникновения важных событий или с целью оптимизации.

В отличие от традиционных систем управления ресурсами предприятий, работающих преимущественно в пакетном режиме, предлагаемая система постоянно работает в реальном времени, адаптивно перестраивая план под действием любых заданных событий во внешнем мире. Она устанавливается и запускается на сервере и работает далее непрерывно, реагируя на события, вводимые оператором или приходящие из других систем.

При этом система постоянно стремится улучшать создаваемые планы операций путем не только использования свободных слотов времени станков или рабочих, но и цепочечных сдвижек ранее размещенных операций заказов или их переброски на другие ресурсы. Автоматизация такой адаптивной корректировки планов позволяет минимизировать ручные изменения и учитывать динамично изменяющуюся ситуацию, специфику заказов, особенности имеющихся станков и рабочих и многие другие факторы, которые делают задачу диспетчеризации ресурсов цеха столь сложной и трудоемкой.

Применяемый метод адаптивного планирования основан на воспроизведении работы самоорганизующегося «коллективного интеллекта», подобного колонии муравьев или роя пчел. Сотни и тысячи автономных программных агентов заказов и ресурсов, обладающих конфликтными интересами, кооперируя и конкурируя друг с другом, формируют расписание работы ресурсов цеха в ходе многочисленных взаимодействий путем переговоров для согласованного разрешения возникающих конфликтов и поиска баланса интересов, что позволяет системе всегда оставаться открытой к изменениям, гибко корректировать расписание по событиям, выполняя цепочки подвижек или переброски заказов с одного ресурса на другой в реальном времени.

3. МУЛЬТИАГЕНТНЫЙ АЛГОРИТМ ПРОИЗВОДСТВЕННОГО ПЛАНИРОВАНИЯ

Кратко алгоритм работы мультиагентной системы адаптивного планирования может быть описан следующим образом:

• каждый заказ, задача, операция, станок, рабочий или любой другой ресурс предприятия получает своего программного агента, у которого ведется свое расписание;

• приходящий новый заказ обращается к онтологии (базе знаний, отделенной от программного кода) и зачитывает оттуда технологический процесс своего исполнения;

• под каждую операцию создается свой агент, который получает требования и ограничения на планирование;

• агент начинает планирование путем поиска необходимых ему ресурсов в сцене, которая описывает текущую ситуацию в цехе, а именно, какой станок или рабочий какое расписание исполняет;

• если подходящие ресурсы заняты, то фиксируется конфликт, и начинаются переговоры по его разрешению путем сдвижек и освобождений слотов;

• в ходе переговоров возможны варианты: новый заказ уйдет на менее подходящий ресурс, предыдущий заказ уйдет или сдвинется;

• даже после решения своей задачи каждый агент не останавливается и продолжает пытаться улучшить свое положение.

При планировании заказов могут применяться различные стратегии, основные из них:

• планирование точно в срок (JIT — Just In Time);

• планирование как можно раньше (ASAP — As Soon As Possible).

Чтобы запланировать свои детали и сборочные единицы (ДСЕ), агент заказа с помощью сообщения посылает запрос на планирование первой ДСЕ, которая определяется согласно выбранной стратегии. Для стратегии планирования ASAP — это самые начальные ДСЕ, не имеющие предшественников, а для стратегии планирования JIT — самые поздние, которые необходимо изготовлять в последнюю очередь (обычно это сборочные единицы).

После того, как агенты технологических операций считывают из онтологии собственные требования, они начинают поиск подходящих рабочих и станков для выполнения каждой операции из числа тех, кто свободен или занят, но готов сделать подвижки в своем расписании.

После того, как каждый агент ресурса получает сообщение, он проверяет свои предпочтения (по недоступности, сменам и т. п.) и, если размещение возможно, возвращает ответ агенту операции, что все успешно. В случае, если агент ресурса не может запланировать выполнение в предпочитаемое время, он в ответном сообщении возвращает контрпредложение.

В случае успеха агент ДСЕ возвращает агенту заказа сообщение об успешном планировании. Агент заказа согласно выбранной стратегии планирования определяет следующую ДСЕ и посылает его агенту сообщение о необходимости планирования. При этом учитывается, что в расписании уже есть запланированные ДСЕ. После размещения нового заказа агенты ДСЕ получают возможность проактивно улучшить свое состояние согласно своим предпочтениям. В качестве критериев при этом могут использоваться: размещение на как можно более раннее время, выполнение в срок, обеспечения резерва времени на выполнение и т. п.

Отметим, что при отсутствии каких-либо предпочтений агент заказа может отправить запрос на планирование сразу всем ДСЕ, которые при этом будут планироваться параллельно, согласовывая время своего исполнения друг с другом. В этом случае расписание планируемого производственного процесса будет строиться, начиная от наиболее сильно ограниченных ДСЕ. Какие именно ДСЕ будут сильно ограниченными, как правило, не известно заранее, так как ограниченность здесь связана не только с самой структурой техпроцесса, но и с уже сложившимся расписанием.

В случае возникновения конфликта (например, если расписание станка уже распланировано: 08:00–12:00 ремонт станка, 12:00–17:00 производство других изделий) агенты оборудования могут вступить в переговоры для его разрешения: например, производство других изделий переходит на другой станок, который не подошел по техническим характеристикам для планируемой задачи.

В результате разрешения конфликта, возможно, потребуется подвижка во времени других операций или некоторые из них перейдут на другие станки. В худшем случае планирование наименее приоритетных заказов будет временно приостановлено, чтобы дать возможность исполнить более приоритетные.

В ходе процесса переговоров агентов строится квазиоптимальный, сбалансированный по многим критериям план производства с учетом индивидуальных ограничений и предпочтений, а также целей всего предприятия. В случае возникновения непредвиденных событий (поломка станка, опоздание рабочего), агенты могут динамически, в режиме реального времени, перераспределить задания на другие доступные ресурсы, без пересмотра всего плана производства.

Локальные изменения строятся не по жесткому централизованному алгоритму, а являются результатом совместной работы отдельных агентов, учитывающих свои состояния и действующих по обстоятельствам. Непрерывный поток событий на входе позволяет системе автоматически реагировать на изменения состояния заказов и ресурсов в реальном времени.

Агенты каждого заказа и ресурса строят свои собственные, но связанные в общую сеть расписания, в такой сети расписание может содержать сотни тысяч взаимосвязанных операций. Основой подхода к планированию становится не полный или частичный комбинаторный перебор вариантов, а выявление и разрешение конфликтов путем переговоров агентов и достижения компромиссов — так, как это делают люди в реальной жизни. Созданный план запускается на исполнение, в ходе которого система следит за выполнением и инициирует перепланирование в случае обнаружения расхождений между планом и фактом.

Основной особенностью мультиагентных систем является представление совокупности агентов в виде виртуального мира, в котором эти агенты существуют, моделируют различные возможные ситуации и решают конфликты путем переговоров.

Например, мир завода может содержать агента заместителя директора и агентов основных подразделений, которые могут вести переговоры на верхнем уровне, визуализировать план и идентифицировать проблемы верхнего уровня. Приходящий новый заказ поступает сюда и далее декомпозируется на основе онтологии на составляющие верхнего уровня, инициируя процессы планирования в каждом из других подразделений. Если некоторые события происходят в цехе

и могут быть разрешены путем переговоров внутри него, то это не требует взаимодействия с другими цехами.

Если же не удается решить процесс переговорами локально (например, из-за задержки, вызванной сбоем оборудования), то план должен попытаться сначала перепланироваться между подразделениями (по горизонтали). Если и это не помогает, необходимо информировать заместителя генерального директора и, возможно, выйти на заказчика для согласования сроков поставки изделия или получения дополнительных ресурсов с целью устранения последствий возникшей непредвиденной чрезвычайной ситуации.

Таким образом, система управления производством строится как сеть подсистем планирования (планировщиков) для каждого подразделения предприятия, поддерживающая их взаимодействие для согласования или координации планов. Такая система в любой момент времени, и со стороны любого из своих элементов, может пересматривать связи между этими элементами и согласованно менять их планы по принципу P2P. При этом обеспечивается автоматическое гибкое планирование всех имеющихся ресурсов в масштабе реального времени, как в автоматическом режиме, так и в диалоге с человеком.

4. РЕАЛИЗАЦИЯ МУЛЬТИАГЕНТНОЙ СИСТЕМЫ УПРАВЛЕНИЯ ЦЕХОМ

Разработанная система автоматизирует полный цикл управления цехом:

- ведение базы знаний по станкам, технологическим процессам и рабочим;
- загрузка технологических процессов из системы автоматизированного проектирования (САПР), с которыми обеспечивается интеграция;
- ввод заказов на производство изделий и других важных событий в любой момент времени;
- анализ того, как новый заказ скажется на предыдущих, уже распределенных и запущенных в выполнение, а также его стоимость;
- адаптивная гибкая и интерактивная разработка и корректировка производственного плана по событиям, возможность ручной доводки;
- проактивное улучшение плана в случае наличия ресурса времени;
- визуализация работы цеха в реальном времени;
- гибкая настройка алгоритмов планирования с учетом различных критериев качества производственного процесса;
- согласование производственного расписания со всеми участниками и его корректировка в случае необходимости;
- выдача сменно-суточных заданий по участкам и каждому рабочему;
- обратная связь с рабочими по выполнению заданий или операций;
- оперативный контроль исполнения планов производства;
- построение отчетов для мастеров, менеджеров и экономистов.

Разработанная система имеет распределенную архитектуру, содержит набор автоматизированных рабочих мест (АРМ) для разных ролей пользователей и включает следующие компоненты:

• модуль обработки заказов, с помощью которого пользователь вводит в систему заказы и который позволяет проводить их первичную обработку силами планово-диспетчерского бюро и технологического бюро цеха;

• модуль планирования заказов, который принимает заказы пользователя, планирует их на имеющиеся ресурсы согласно требованиям технологического процесса;

• база знаний (онтология), которая содержит сведения об оборудовании и человеческих ресурсах, графиках их работы и возможностях выполнения технологических операций;

• модуль интеграции, предназначенный для интеграции с САПР ТП.

В стандартную базовую версию системы включены следующие автоматизированные рабочие места:

• АРМ «Начальник цеха»: строит и корректирует месячный план производства подразделения и отображает текущий прогресс по каждому конкретному изделию (сколько изделий сделано, сколько в процессе, обеспеченность производства материалами, инструментами, оборудованием, рабочими);

• АРМ «Мастер»: позволяет строить месячный план производства мастерской (участка), анализировать загруженность ресурсов задачами, загрузку по рабочим, состояние выполнения планов. Реализуются различные стратегии планирования от «точно в срок» — до «как можно быстрее» или «как можно дешевле»;

• терминал рабочего: проводит выдачу сменно-суточных заданий рабочему, получение актуальной информации о выполнении техопераций по мере готовности, ввод других событий (досрочное выполнение или задержка, отсутствие материала, поломка оборудования и др.).

Примеры экранных форм системы приведены на рис. 1–4 (см. с. 118, 119).

Важно отметить, что при планировании учитываются различные особенности назначения технологических операций рабочим на исполнение, присущие конкретному производству.

Например, при изготовлении инструмента второго порядка необходимо обеспечить его готовность к началу соответствующей операции, где он используется (согласно указаниям технологического процесса). Это позволяет планировать его изготовление параллельно с производством изделия, в котором он используется, что должно учитываться алгоритмом планирования.

Другой особенностью является назначение рабочих. Кроме кода квалификации и разряда необходимо учитывать ряд особенностей. Все однотипные операции одного технологического процесса (например, слесарные) должны планироваться одному рабочему. Это связано с тем, что он, начиная с первой операции, лучше знает особенности сложного изделия и может на более поздних операциях компенсировать неточности выполнения работ в начале технологического процесса. Такое правило позволяет снизить брак.

Реализация такого подхода в разработанной системе выполнена с помощью описания для ресурсов онтологических свойств. Для одного рабочего может быть задано несколько онтологических свойств и в случае совпадения значения каждого из них со значениями соответствующих онтологических свойств планируемой операции приоритет данного ресурса при назначении будет изменяться.

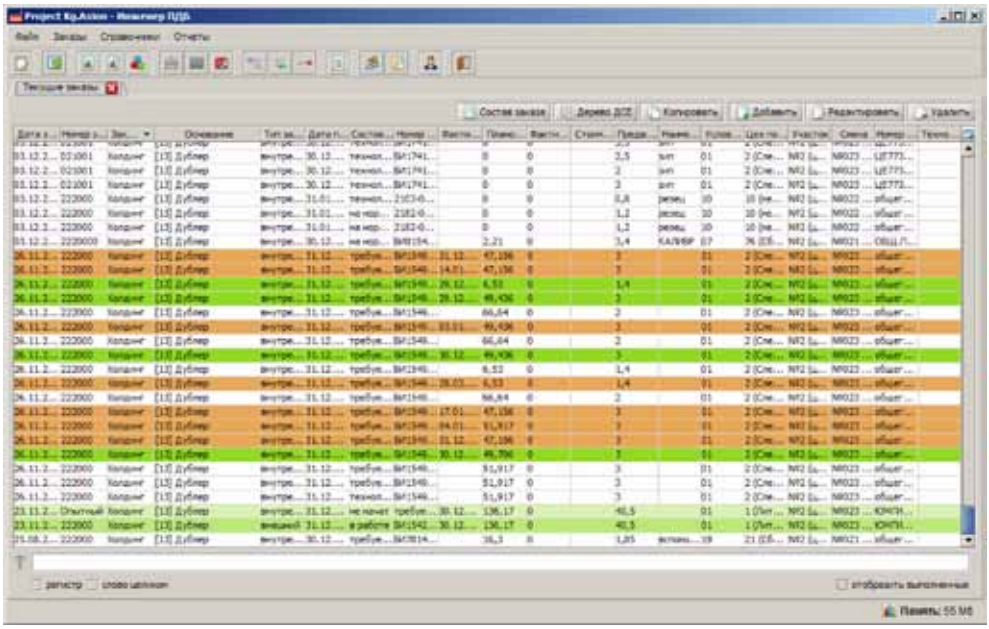


Рис. 1. Список заказов, поступающих в систему

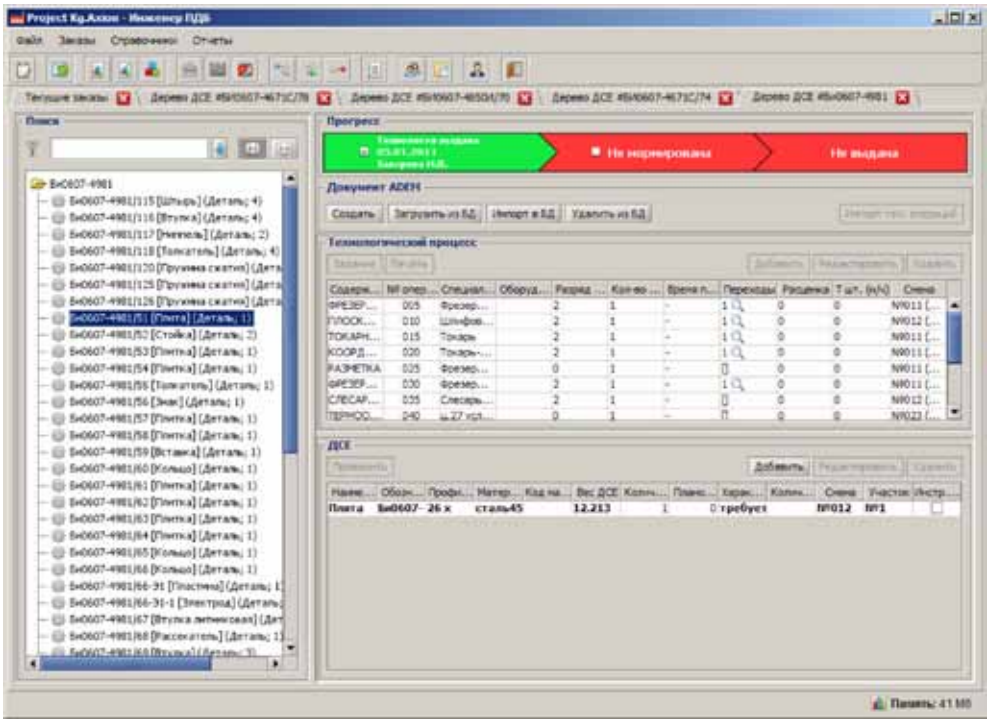


Рис. 2. Описание изделий, входящих в заказ, и технологических процессов их производства

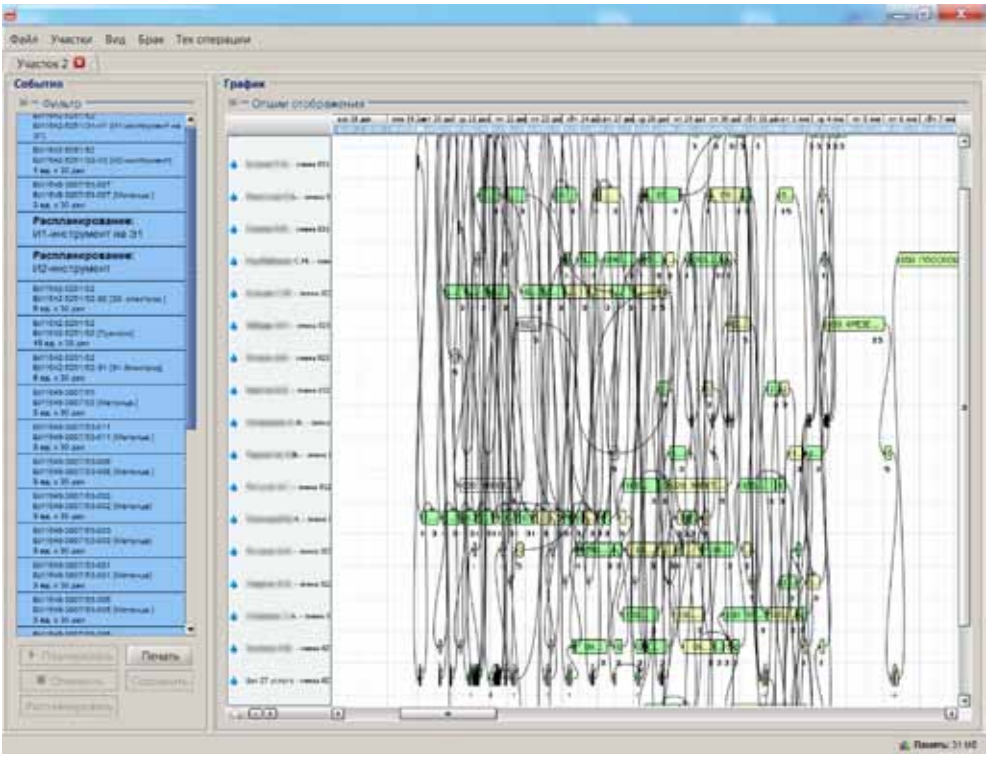


Рис. 3. Производственное расписание, полученное в результате мультиагентного планирования

Для ресурсов указываются свойства как повышающие этот приоритет, так и понижающие.

Онтологические свойства заполняются автоматически путем сравнения значений параметров технологической операции со значениями имеющихся свойств ресурсов. Например, допустим, у рабочего установлена в онтологическом свойстве марка материала, с которой он работает. В случае, если у технологического процесса, к которому относится планируемая операция, задан материал, в описании которого имеется указание соответствующей марки, приоритет этого рабочего при планировании будет повышен. Таким образом, при планировании кроме кода квалификации и разряда рабочих (имеющих наибольшее значение) учитываются онтологические свойства рабочих и операций, что позволяет повысить адекватность создаваемых расписаний.

Рис. 4. Терминал, на котором отображается сменно-суточное задание

5. РЕЗУЛЬТАТЫ ИССЛЕДОВАНИЯ

Результаты анализа производительности рассматриваемых алгоритмов планирования приведены на рис. 5, 6. Для эксперимента было создано необходимое число заказов, путем копирования одного из реальных: за основу был взят заказ на изготовление пуансона, состоящего из пяти компонентов — четырех деталей и одной сборки. Для сравнения алгоритмов было запланировано 100 заказов, что соответствует 500 изделий (3300 технологических операций).

На рис. 5 приведена зависимость времени t планирования каждого следующего заказа от его номера N в очереди. Линейный характер сохраняется и при увеличении числа заказов до 500. На рис. 6 приведена зависимость числа передвижек K, которые проводятся в ходе мультиагентных переговоров, от номера заказа N в очереди. Предпочитаемое время планирования здесь устанавливалось случайно с равномерным распределением по заданному интервалу, что оправдано свойствами предметной области. Видно, что, несмотря на рост времени планирования каждого следующего заказа, число передвижек операций в расписании варьируется в широком пределе.

В качестве вывода можно указать, что инкрементальное планирование позволяет обрабатывать поступающие события без полной перестройки расписа-

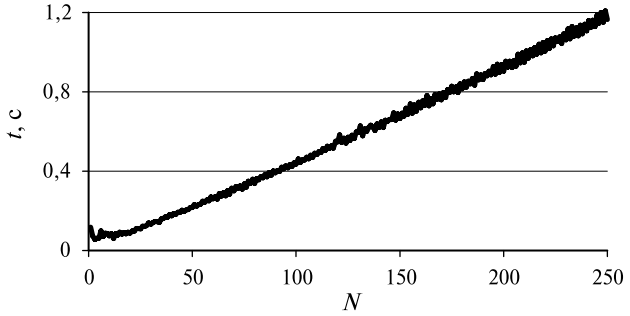


Рис. 5. Зависимость времени планирования очередного заказа

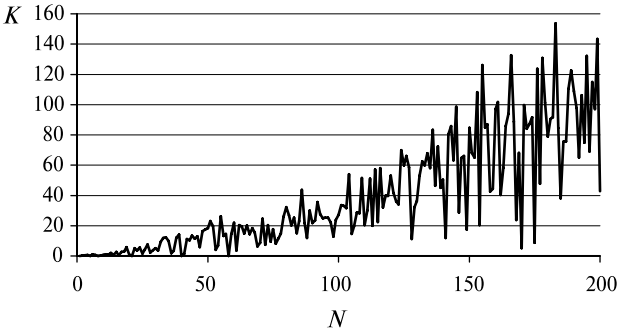


Рис. 6. Зависимость числа передвижек операций от порядка планирования заказов

ния, что определяет возможность их применения при необходимости управления производственными ресурсами в режиме реального времени.

ЗАКЛЮЧЕНИЕ

Полученные результаты позволяют сделать вывод о целесообразности применения алгоритмов управления распределением ресурсов в реальном масштабе времени с использованием мультиагентных технологий в машиностроении, особенно при организации управления мелкосерийными и опытными производствами. Развитие данного подхода видится в более глубокой проработке алгоритмов планирования и тиражировании найденных решений.

ЛИТЕРАТУРА

[Андреев и др., 2010*а*] *Андреев М. В.*, *Иващенко А. В.*, *Карсаев О. В.*, *Самойлов В. В.*, *Скобелев П. О.*, *Царев А. В.* Поддержка процессов коллективного принятия решений по управлению инструментальным производством на основе мультиагентной системы планирования ресурсов в реальном времени // Материалы научно-технического семинара «Управление в распределенных сетецентрических и мультиагентных системах. СПб.: ОАО «Концерн «ЦНИИ «Электроприбор», 2010. С. 75–80.

[Андреев и др., 2010*б*] *Андреев М. В.*, *Иващенко А. В.*, *Скобелев П. О.*, *Кривенок С. А.* Мультиагентная система распределения производственных ресурсов в тяжелом машиностроении // Программные продукты и системы. 2010. № 3. С. 100–104.

[Виттих, Скобелев, 2003] *Виттих В. А.*, *Скобелев П. О.* Мультиагентные модели взаимодействия для построения сетей потребностей и возможностей в открытых системах // Автоматика и телемеханика. 2003. № 1. С. 177–185.

[Виттих, Скобелев, 2009] *Виттих В. А.*, *Скобелев П. О.* Метод сопряженных взаимодействий для управления распределением ресурсов в реальном масштабе времени // Автометрия. 2009. № 2. С. 78–87.

[Загидуллин, 2005] *Загидуллин Р. Р.* Структура системы оперативно-календарного планирования в гибких производственных системах // Автоматизация и современные технологии. 2005. № 2. С. 44–46.

[Леньшин и др., 2003] *Леньшин В. Н.*, *Куминов В. В.*, *Фролов Е. Б.*, *Будник Р. А.* Производственные исполнительные системы (MES) — путь к эффективному предприятию // САПР и графика. 2003. № 6. [Электрон. текст]. Режим доступа: www.sapr.ru.

[Andreev et al., 2010] *Andreev M.*, *Ivaschenko A.*, *Skobelev P.*, *Tsarev A.* A Multi-agent platform design for adaptive networks of intelligent production schedulers // 10th IFAC Workshop on Intelligent Manufacturing Systems (IMS'10). Lisbon, Portugal. 2010. P. 87–92.

[Leitão, Restivo, 2008] *Leitão P.*, *Restivo F.* Implementation of a holonic control system in a flexible manufacturing system // IEEE Trans. Systems, Man and Cybernetics. Pt. C: Applications and Reviews. 2008. V. 38. N. 5. P. 699–709.

[Leitão, 2009] *Leitão P.* Holonic rationale and bio-inspiration on design of complex emergent and evolvable systems // Trans. Large-Scale Data and Knowledge Centered Systems I. LNCS 5740. Springer-Verlag, 2009. P. 243–266.

MULTI-AGENT TECHNOLOGY FOR REAL TIME MANUFACTURING RESOURCES MANAGEMENT

P. O. Skobelev, A. V. Ivaschenko, M. V. Andreev, I. O. Babanin

SEC "Smart solutions", Moscow

This paper gives brief overview of modern developments in the area of automation of manufacturing resources management in real time based on multi-agent technology. These developments were carried out in "Smart solutions" Software Engineering Company for a number of manufacturing enterprises and result in implementation and deployment of specialized multi-agent software. This paper describes new principles of intelligent software development for cooperative decision making support based on bio-inspired approach and points out basic benefits that are obtained due to such a technology practical application.

Keywords: multi-agent technologies, ontologies, complex systems, production scheduling, manufacturing.

Skobelev Petr Olegovich — doctor of technical sciences, chief researcher, e-mail: petr.skobelev@gmail.com.

Ivaschenko Anton Vladimirovich — candidate of technical sciences, business unit manager, e-mail: anton.ivashenko@gmail.com.

Andreev Michael Vladimirovich — candidate of technical sciences, business unit manager, e-mail: michael.v.andreev@gmail.com.

Babanin Ivan Olegovich — chief developer, e-mail: babanin@gmail.com.

УДК 519.6+621.039.61

СТРУКТУРА И ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ КОМПЛЕКСА ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ «ВИРТУАЛЬНЫЙ ТОКАМАК»

Ф. С. Зайцев [1,2], А. Г. Шишкин [1], Д. Ю. Сычугов [1], В. Э. Лукаш [3], Ю. В. Митришкин [4], Р. Р. Хайрутдинов [5], С. В. Степанов [1], Е. П. Сучков [1]

[1] *Московский государственный университет имени М. В. Ломоносова*
[2] *Учреждение Российской академии наук Научно-исследовательский институт системных исследований РАН* (*НИИСИ*), *Москва*
[3] *Институт ядерного синтеза РНЦ «Курчатовский институт»*, *Москва*
[4] *Институт проблем управления Российской академии наук*, *Москва*
[5] *Троицкий институт инновационных и термоядерных исследований*, *Троицк*

В работе описаны концепция комплекса программ «Виртуальный токамак», его функциональные возможности, основные компоненты и способ использования. Обсуждены место и роль проекта в развитии промышленного применения управляемого термоядерного синтеза в России и за рубежом для производства энергии. Рассмотрены проблемы применения современных компьютерных систем решения задач комплексного моделирования плазмы. Сформулированы требования к вычислительным системам и подготовке кадров. Россия, имеющая признанные теоретическую и вычислительную школы, может стать мировым лидером в имитационном моделировании токамака путем относительно небольших инвестиций с целью создания инновационных технологий для последующего внедрения в промышленность.

Ключевые слова: математическое моделирование, токамак, комплекс программ, управляемый термоядерный синтез.

ВВЕДЕНИЕ

В последние десятилетия практически во всех развитых странах ведутся интенсивные исследования в области управляемого термоядерного синтеза (УТС). Конечная цель исследований — построение относительно безопасных электростанций с практически неисчерпаемым источником энергии. В основе принципа действия термоядерной электростанции лежит ядерная реакция синтеза изотопов водорода. В результате такой реакции образуются ионы инертного газа

Зайцев Федор Сергеевич — профессор, доктор физико-математических наук, e-mail: zaitsev@cs.msu.su.

Шишкин Алексей Геннадьевич — старший научный сотрудник, доктор физико-математических наук, e-mail: zaitsev@cs.msu.su.

Сычугов Дмитрий Юрьевич — доцент, кандидат физико-математических наук, e-mail: sychugov@cs.msu.su.

Лукаш Виктор Эммануилович — ведущий научный сотрудник, профессор, доктор физико-математических наук, e-mail: lukash@nfi.kiae.ru.

Митришкин Юрий Владимирович — ведущий научный сотрудник, профессор, доктор технических наук, e-mail: yvm@mail.ru.

Хайрутдинов Рустам Рашитович — ведущий научный сотрудник, доктор физико-математических наук, e-mail: khayrutd@mail.ru.

Степанов Сергей Владимирович — аспирант, e-mail: sergey.v.stepanov@gmail.com.

Сучков Егор Петрович Сучков — аспирант, e-mail: suchkov.egor@gmail.com.

гелия и выделяется большое количество энергии. Аналогичные процессы служат источником энергии Солнца и других звезд. Несколько десятков килограммов дейтерия и трития достаточно для обеспечения России энергией в течение года.

Основная задача современных экспериментов по УТС состоит в нагреве вещества до высокой температуры, при которой начинается термоядерная реакция, и удержании его в течение необходимого времени в некотором изолированном от внешней среды объеме.

Выдающиеся успехи достигнуты на установках с удержанием плазмы магнитным полем тороидальной конфигурации. В 1991 г. на самой большой в мире установке токамак JET, расположенной в Великобритании, впервые экспериментально получена управляемая термоядерная плазма со значительным количеством реакций синтеза. Ведущие страны мира, включая Россию, приступили к сооружению во Франции реактора-токамака ITER в рамках международного проекта. Одна из главных задач проекта — демонстрация экономической целесообразности термоядерных электростанций с магнитным удержанием плазмы.

В России разработана Федеральная целевая программа «Овладение энергией термоядерного синтеза, создание научно-технологической базы термоядерной энергетики в России на 2009–2015 гг.». Финансирование до 2050 г. должно составить 515 млрд р. [Ведомости, 2007].

Важную роль в управляемом термоядерном синтезе играют теоретические исследования и математическое моделирование [Днестровский, Костомаров, 1993; Зайцев, 2005]. Основываясь на надежном физическом и математическом фундаменте, вычислительном эксперименте на суперЭВМ, теория позволяет правильно спроектировать установку, объяснить наблюдаемые явления, рассчитать различные характеристики процессов, в том числе не измеряемые непосредственно, решать задачи управления и оптимизации, дать обоснованный прогноз свойств плазмы в проектируемых реакторах, предсказать ее поведение в экстремальных условиях, повысить эффективность использования уникального дорогостоящего оборудования, избежать его разрушения, снизить расходы на строительство новых установок, оптимизировать режимы работы термоядерной электростанции, понизить себестоимость производства энергии.

1. КОНЦЕПЦИЯ КОМПЛЕКСА

Как уже отмечалось, ведущие страны мира, включая Россию, приступили к строительству международного термоядерного реактора-токамака ITER. Помимо создания реактора разрабатывается и модернизируется ряд малых и средних установок с целью проведения поддерживающих физических исследований, обучения персонала, отработки новых диагностик, методов управления, нагрева плазмы и поддержания тока, проверки конструктивных решений и т. д. В мире насчитывается несколько десятков токамаков. Практически каждый из них уникален по характеристикам создаваемой плазмы и элементам конструкции.

Проведению экспериментальных кампаний на действующих установках и проектированию новых предшествуют численные расчеты магнитной конфигурации, равновесия, устойчивости, ожидаемых параметров плазмы, разработка

и настройка систем управления. Одной из важных задач является прогнозирование поведения плазменного шнура в ходе эксперимента — так называемый расчет сценария разряда. Разработка соответствующих компьютерных кодов и проведение вычислительных экспериментов с их помощью является задачей чрезвычайной сложности.

В настоящее время в рамках исследовательских программ российскими учеными накоплен колоссальный объем математических методов, программного обеспечения, различных информационных подходов. Созданы и успешно эксплуатируются численные коды, моделирующие наиболее важные процессы в плазме.

Прикладное программное обеспечение УТС разрабатывалось в разное время многими авторами и имеет свои специфические особенности, что влечет неизбежные трудности при совместном применении кодов. Длительность разработки одного кода обычно составляет 5...7 лет. Коды характеризуется многомодельностью и многовариантностью. Исследователям приходится работать с десятками различных моделей физических процессов и десятками вариантов численных алгоритмов. Общее число расчетных программ для решения конкретной задачи обычно составляет сотни. При проведении вычислительного эксперимента, анализе его результатов возникает множество рутинных операций: подготовка текстов программ для решения конкретных задач, формирование исходных данных, распределение задач по сети компьютеров, мониторинг процесса прохождения задач, визуализация входных и выходных данных.

Эффективное использование и развитие столь сложного программного обеспечения и поддержка связанных с его эксплуатацией процессов требует разработки и применения специальных системных программных средств. Общепринятые стандарты здесь пока не сформировались, ведется поиск оптимальных форм.

Естественным образом возникает идея создания единой программной среды для проектирования, сопровождения, планирования и интерпретации экспериментов на установках токамак. Первым шагом для ее создания является стандартизация уже имеющихся численных алгоритмов (кодов), предназначенных для моделирования процессов в установках токамак, и объединение их в библиотеку стандартных программ «Виртуальный токамак»; вторым — создание удобного графического интерфейса для решения задач имитационного моделирования.

В широком смысле «Виртуальный токамак» — это полномасштабная, детализированная, интерактивная компьютерная модель реального термоядерного реактора.

Комплекс имитационного моделирования «Виртуальный токамак» имеет целью:

- применение и развитие новых информационных технологий, математических моделей и методов управления многосвязными динамическими объектами для всестороннего изучения источника энергии, построенного на основе реакции термоядерного синтеза в системах с удержанием плазмы магнитным полем тороидальной конфигурации;
- проверку и настройку многочисленных используемых методов и систем диагностики плазмы, анализ эффективности новых подходов;

• комплексное расчетное сопровождение проектов реакторов ITER, Ignitor, DEMO в части описания физических процессов в вакуумной камере, диагностики плазмы, создания систем управления плазмой;

• проектирование новых установок;

• проведение широкомасштабного поискового, предсказательного, оптимизационного и диагностического вычислительного эксперимента на суперЭВМ производительностью вплоть до петафлопного и экзафлопного класса;

• сопровождение и интерпретацию натурных экспериментов.

Первоначальное ядро комплекса строилось из относительно небольшого числа хорошо апробированных на практике кодов. Ядро снабжено общей интерактивной оболочкой, позволяющей эффективно работать с объемными программами. Далее расширение системы осуществлялось путем подключения новых программ (модулей) с соответствующей модификацией оболочки.

Основа реализации и применения комплекса имитационного моделирования — построение развитого информационно-вычислительного портала, позволяющего в удаленном режиме через стандартный веб-браузер использовать локально хранящиеся и поддерживаемые наукоемкое программное обеспечение и системы имитационного моделирования с автоматическим распределением вычислительной нагрузки по сети компьютеров. Недавно появившиеся технологии веб-программирования, новые компьютерные протоколы и средства связи предоставляют базовые средства для создания такого портала.

Акцент ставится именно на локальное хранение, поддержку и предоставление кодов в связи с большой наукоемкостью, сложностью, объемностью, многомодельностью, многовариантностью, высокой динамикой развития программного обеспечения. Обычно использование и развитие кодов, адаптация их к конкретной вычислительной платформе требуют участия авторов в той или иной форме.

Современные реалистичные модели плазмы, алгоритмы управления, предполагаемая вычислительная нагрузка портала требуют применения суперЭВМ с быстродействием вплоть до петафлопного и экзафлопного класса. Использование такой техники особенно важно для детального трехмерного предсказательного моделирования термоядерных реакторов, проверки стратегий управления и устранения ситуаций, при которых возможно полное разрушение конструктивных элементов устройств.

В комплексе нашли отражение базовые компоненты токамака, основные модели плазмы и системы управления ей, такие как: конструктивные элементы (вакуумная камера, катушки тороидальных и полоидальных магнитных полей и т. д.), разнообразная диагностика плазмы, системы магнитного и кинетического управления плазмой, модели исполнительных устройств, комплексы дополнительного нагрева плазмы и генерации тока, дивертор и др.

Основной способ работы пользователя с системой — посредством стандартного веб-браузера через Интернет. При этом важно, что от пользователя не требуется устанавливать или настраивать какое-либо специальное прикладное и системное программное обеспечение. Запуск кодов, мониторинг вычислений и визуализация данных происходят в единой среде SCoPEShell [Зайцев, 2005].

В состав прикладной библиотеки комплекса входят коды, моделирующие процессы, происходящие в токамаке в различные моменты разряда, поэтому большое внимание уделено синхронизации во времени последовательности выполнения кодов. Возникающая вычислительная нагрузка автоматически распределяется по сети компьютеров с помощью системы TADISYS [Зайцев, 2005].

Программы ориентированы на работу в операционной среде Linux. Выполнение созданных под ОС Windows компьютерных кодов возможно под эмулятором VMware или пакетом Windows API Wine.

Структура информационно-вычислительного портала «Виртуальный токамак» изображена на рис. 1. Система управления контентом портала основана на свободно распространяемой системе Drupal, дополненной специализированными компонентами, обеспечивающими надежность, безопасность, быстродействие, универсальность, гибкость и простоту обслуживания портала.

Пользователю портала может быть назначен один их трех уровней доступа к программному обеспечению: просмотр информационных материалов, ограниченный доступ, полный доступ. Ограниченный доступ предоставляет только демонстрационное использование некоторых кодов библиотеки. При полном доступе результаты работы с библиотекой сохраняются для каждой сессии и в последующем могут быть использованы. Кроме того, при полном доступе вычислительные задачи получают более высокий приоритет.

Рис. 1. Структура информационно-вычислительного портала «Виртуальный токамак». Стрелками показаны основные информационные потоки

2. ИНТЕРФЕЙС ПОЛЬЗОВАТЕЛЯ

В состав комплекса входят коды, написанные различными коллективами авторов и предназначенные для решения большого числа специфических задач. Поэтому значительное внимание при создании интерактивной оболочки системы уделено разработке удобного, обладающего многочисленными функциями и в то же время интуитивно понятного графического интерфейса пользователя. Для обеспечения платформенной независимости в качестве языка разработки интерфейса выбран язык Java. Общая схема, лежащая в основе работы графического интерфейса, представлена на рис. 2.

Первым элементом верхнего уровня интерфейса является блок с конкретными установками токамак — ITER, JET, MAST и др. Пользователь с административными привилегиями имеет возможность добавлять новые установки, редактировать и удалять уже имеющиеся. Обычный пользователь также может добавлять новые установки, редактировать и удалять данные, но относящиеся только к тем токамакам, которые добавлены непосредственно этим пользователем. Такие установки и данные отображаются только в рабочем пространстве внесшего их пользователя, но могут стать доступными для всех после тщательной верификации администратором. На верхнем уровне задаются данные, относящиеся к конструкции (геометрии) установки.

Следующими элементами верхнего уровня интерфейса являются так называемый «Сценарий», т. е. ряд отдельных экспериментов, объединенных общим направлением исследования, например, анализом эффективности генерации тока в плазме электромагнитными волнами, и «Разряд» — конкретный эксперимент. На верхнем уровне определяются все глобальные параметры, необходимые для проведения вычислительного эксперимента.

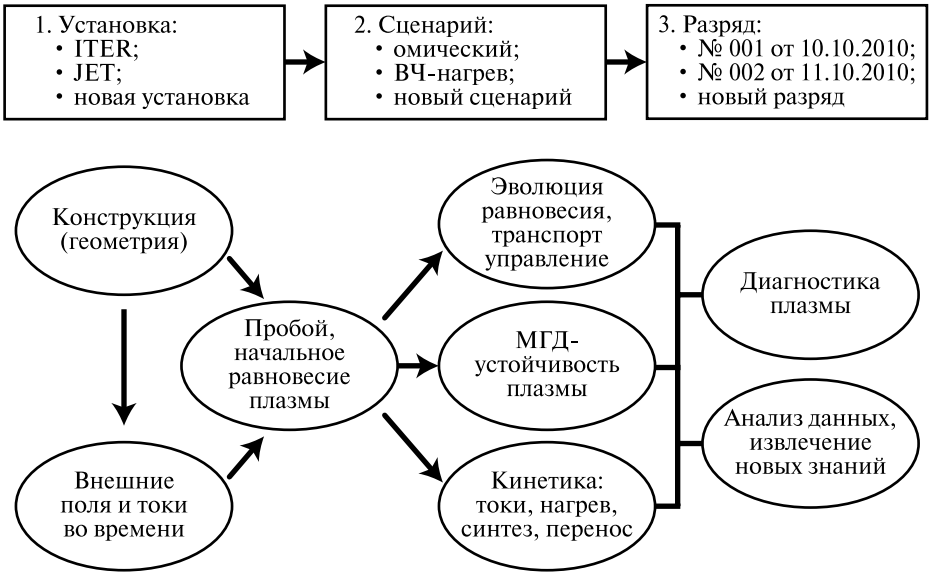


Рис. 2. Общая схема пользовательского интерфейса комплекса «Виртуальный токамак»

После задания входных параметров пользователь может начать моделирование отдельных процессов физического эксперимента или эксперимента целиком. Результаты работы одних кодов, как правило, служат входными данными для других. Поэтому функционирование компонентов комплекса строго синхронизировано. Так, например, нельзя начать моделирование эволюции равновесия плазменного шнура, пока не рассчитаны пробой и начальное равновесие. Группы взаимосвязанных кодов показаны на рис. 2 в виде элементов на общей шине.

3. ПРИКЛАДНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Большой разброс характерных времен и пространственных масштабов развития явлений в плазме токамака делает невозможным в настоящее время создание единого кода, который бы охватил все процессы. Именно поэтому требуется объединение в единую систему различных специализированных кодов, которые в совокупности позволяют спроектировать установку токамак и проанализировать ее работу.

В библиотеку комплекса «Виртуальный токамак», прежде всего, вошли имеющиеся в распоряжении коллектива разработчиков численные коды (см. далее), которые были стандартизированы в соответствии с общей концепцией библиотеки. Коды широко известны, неоднократно применялись при проектировании, моделировании, интерпретации и сопровождении экспериментов на установках токамак. В целом коды библиотеки комплекса можно разделить на следующие большие группы в соответствии с выполняемыми ими задачами.

1. *МГД-равновесие* — состояние, при котором газокинетическое давление плазмы уравновешено силой Лоренца. Характерное время — от сотни миллисекунд на малых токамаках до часа и более на больших проектируемых установках. Коды TOKAMEQ [Сычугов, 2008], SCoPE [Зайцев, 2005], DINA-EQDSK, DINA-SVD [Khayrutdinov, Lukash, 1993; Хайрутдинов, Лукаш, 2008, 2009].

2. *Эволюция равновесия* — медленное изменение свойств плазмы, при котором в каждый момент сохраняется МГД-равновесие. Характерное время — от десятков миллисекунд и выше до сотых и тысячных долей времени разряда. Коды SCoPE, DINA.

3. *Транспорт заряженных частиц* — эволюция с учетом нагрева плазмы, потоков теплоты и частиц в радиальном направлении, выделения и поглощения энергии. Характерное время — как в моделях эволюции. Коды ASTRA [Pereverzev et al., 1992], DINA-TRANSP [Хайрутдинов, Лукаш, 2010], FPP-3D [Зайцев, 2005].

4. *Кинетика* — описание поведения частиц высоких энергий с помощью функции распределения. Расчет эффектов термоядерного синтеза, нагрева плазмы и генерации тока инжекцией и высокочастотными волнами, потерь и источников частиц, тепловой нагрузки элементов конструкции. Характерное время — от сотен миллисекунд до часа и более. Коды FPP-3D [Зайцев, 2005], QUARK3D.

5. *МГД-устойчивость* — свойство МГД-равновесной плазмы быть устойчивой по отношению к малым возмущениям. Характерное время — от микро- до

миллисекунд в зависимости от вида неустойчивости. Известно около 15 типов основных неустойчивостей тороидальной плазмы. Код TOKSTAB [Сычугов и др., 2010*а*].

6. *Диагностика* — восстановление внутренних параметров плазмы по внешним измерениям. Обработка и анализ данных, извлечение новых знаний. Задачи обработки данных в реальном времени. Коды SCoPE, VIP, SDSS, NNTMM, CLUNAVT, DaMa, FPP-3D, FIRe и др. [Зайцев, 2005; Lukyanitsa et al., 2007, 2008].

7. *Управление* — целенаправленное изменение параметров плазменного шнура на относительно медленных временах (миллисекунды, сотые и тысячные доли времени разряда). Коды SCoPE, DINA.

4. СИСТЕМНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

В качестве среды выполнения модулей комплекса, как уже отмечалось, выбрана ОС Linux. Для программ, разработанных в других операционных системах, предоставлена возможность их выполнения с помощью эмулятора VMware или альтернативной реализации Windows API Wine.

Удаленный доступ по протоколу Remote FrameBuffer (RFB) к рабочему столу компьютера реализован с помощью платформонезависимой системы Virtual Network Computing (VNC). Ее достоинством является приемлемая скорость обновления графического экрана даже на относительно медленных линиях связи.

Система безопасности данных основана на SELinux. Пользователям назначаются предопределенные роли таким образом, чтобы они не могли получить доступ к файлам и процессам других владельцев. Отличие от обычной системы Unix-привилегий состоит в том, что определенные пользователям роли, или контексты безопасности, в которых они находятся, имеют ограниченный и более управляемый доступ к файлам и ресурсам компьютера.

Входящие в состав комплекса имитационного моделирования коды предъявляют весьма серьезные требования к вычислительным ресурсам. Возникает задача распределения вычислений по сети доступных компьютеров. Для ее решения используется разработанная авторами проекта система TADISYS [Зайцев, 2005], позволяющая автоматизировать рутинные операции распределения независимых задач по сети, включая выбор оптимального узла сети, загрузку и запуск задачи, возврат данных на базовый компьютер. Система TADISYS и ее графический интерфейс реализованы на языке Java. Применен подход клиент – сервер. Сервер выполняется на базовом компьютере пользователя. Клиенты рассылаются сервером по узлам сети и запускаются на выполнение. Доступ к удаленным компьютерам и обмен данными организован по защищенному протоколу ssh.

Автоматизация рутинных операций настройки входных и выходных данных численных кодов комплекса, их компиляция и запуск, мониторинг формирования данных, преобразование формата данных, построение дву- и трехмерных графиков, обеспечение доступа к распределенным по сети компьютеров файлам осуществляется с помощью графической среды SCoPEShell [Костомаров и др., 2010; Зайцев 2005].

Рис. 3. Архитектура вычислительной системы комплекса «Виртуальный токамак»

Работа с комплексом требует выполнения многочисленных рутинных операций процесса настройки исходных файлов прикладного программного обеспечения на решение конкретной задачи. Для их автоматизации разработана система UMB [Зайцев, 2005], реализующая следующие основные функции:

- интерпретация специального языка настройки модулей, описывающего диалог с пользователем и процесс сборки расчетной программы из набора модулей;
- диалог с пользователем в виде удобного графического интерфейса;
- формирование расчетной программы из модулей с преобразованием текста их тела, включающим осуществление макроподстановок, выбор фрагмента из числа альтернативных и других операций редактирования;
- документирование процесса сборки расчетной программы, повторение процесса сборки в упрощенном режиме диалога на основе данных из архива, предоставление разнообразной справочной информации.

Система UMB полностью написана на Java.

5. АРХИТЕКТУРА ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ

Архитектура вычислительной системы комплекса «Виртуальный токамак» изображена на рис. 3. Стрелками отмечены направления информационных потоков. Пользователи комплекса работают с сервером портала через сеть. Сервер обеспечивает доступ к прикладному и системному программному обеспечению, хранилищу данных, распределяет вычислительную нагрузку по компьютерам.

На рис. 3 также представлена оценка необходимых вычислительных ресурсов при штатной нагрузке комплекса. Проведение масштабных вычислительных экспериментов требует использования нескольких мощных MPI-кластеров и суперЭВМ вплоть до петафлопного и экзафлопного класса.

ЗАКЛЮЧЕНИЕ

В ходе начальной эксплуатации комплекса «Виртуальный токамак» подтверждены преимущества интеграции кодов в единую систему. Например, при расчете сценария разряда на строящемся в республике Казахстан токамаке КТМ комплекс позволил оперативно сопоставлять результаты вычислений по нескольким моделям, что существенно повысило адекватность и надежность результатов [Сычугов и др., 2010*б*].

«Виртуальный токамак» значительно уменьшит затраты на проектирование, строительство и введение установок в эксплуатацию, а также существенно сэкономит очень дорогое экспериментальное время, которое на действующих крупных токамаках оценивается в 100 тыс. дол. в день, а на проектируемом токамаке ITER — в 1 млн дол. в день.

Комплекс «Виртуальный токамак» впервые объединил в единую систему коды, моделирующие различные процессы в плазме установок токамак: МГД-равновесие, МГД-устойчивость, транспорт, эволюцию, влияние различного рода внешних источников, а также коды, решающие задачи диагностики и управления плазмой. Наличие общей программной оболочки позволяет эффективно осуществлять комплексное моделирование процессов в токамаке, рассматривая установку и происходящие в ней процессы в целом. В ряде зарубежных научных центров по управляемому термоядерному синтезу ведется создание подобного рода библиотек. Однако вследствие слабого вовлечения специалистов по прикладной математике и программированию этим разработкам присущ ряд существенных недостатков. В России аналогов комплекса «Виртуальный токамак» нет.

Комплекс частично реализован в рамках специализированного информационно-вычислительного портала (http://leader.cs.msu.su) с использованием и развитием платформонезависимых технологий, сетевого протокола VNC, различных подходов веб-программирования, методов распределенных вычислений, подходов теории управления и принятия решений, методов защиты информации и обеспечения безопасности компьютерных систем. Совокупность методики и технологий впервые применена к решению задач в области моделирования высокотемпературной плазмы. Новые подходы позволили заметно повысить производительность труда исследователя, сделать уникальное наукоемкое прикладное программное обеспечение доступным специалистам, не занятым непосредственно в его разработке.

Воплощение и развитие комплекса «Виртуальный токамак» представляется одним из стратегических направлений в решении проблемы УТС, существенно повышающим эффективность и сокращающим сроки перехода к термоядерной энергетике. Комплекс представляет конкретную реализацию прорывных технологий информационно-вычислительных порталов, управления сложными

динамическими объектами, супервычислений, имитационного моделирования, стимулирует подготовку кадров для решения прикладных задач на суперЭВМ следующего поколения, открывает возможность применения создаваемых новых технологий, алгоритмов управления и программного обеспечения во многих отраслях науки и промышленности.

Россия, не обладающая современными установками токамак, но имеющая признанные теоретическую и вычислительную школы, может стать мировым лидером в имитационном моделировании токамака путем относительно небольших инвестиций с целью создания инновационных технологий для последующего внедрения в промышленность.

Проект «Виртуальный токамак» реализуется при поддержке известных ученых: академиков С. К. Коровина и В. Б. Бетелина, члена-корреспондента Д. П. Костомарова, профессоров Ю. Н. Днестровского и А. Г. Кушниренко.

Исследования поддерживались грантами Президента РФ, Российского фонда фундаментальных исследований, Правительства Москвы, совместными проектами России с зарубежными странами, группой FARG Ltd.

ЛИТЕРАТУРА

[Ведомости, 2007] Ведомости. [Электрон. текст]. Режим доступа: http://www.vedomosti.ru. 2007. № 152.

[Днестровский, Костомаров, 1993] *Днестровский Ю. Н.*, *Костомаров Д. П.* Математическое моделирование плазмы. М.: Наука, 1993. 336 с.

[Зайцев, 2005] *Зайцев Ф. С.* Математическое моделирование эволюции тороидальной плазмы. М.: МАКС Пресс, 2005. 524 с.

[Костомаров и др., 2010] *Костомаров Д. П.*, *Зайцев Ф. С.*, *Шишкин А. Г.*, *Степанов С. В.* Графический интерфейс ScopeShell: поддержка вычислительного эксперимента и визуализация данных // Вестн. Моск. ун-та. Вычисл. математика и кибернетика. 2010. Сер. 15. С. 42–48.

[Сычугов, 2008] *Сычугов Д. Ю.* Код для расчета МГД-равновесия TOKAMEQ (модуль библиотеки программ «Виртуальный токамак») // Вопросы атомной науки и техники. Сер. Термоядерный синтез. 2008. Вып. 4. С. 85–89.

[Сычугов и др., 2010*а*] *Сычугов Д. Ю.*, *Шаповалов Г. В.*, *Волынкина Ю. В.*,. *Садыков А. Д.*, *Чектыбаев Б. Ж.*, *Шипилов Д. В.*, *Шумайлова О. Н.* Численное моделирование омического сценария разряда в токамаке KTM // Вопросы атомной науки и техники. Сер. Термоядерный синтез. 2010. Вып. 1. С. 38–45.

[Сычугов и др., 2010*б*] *Сычугов Д. Ю.*, *Амелин В. В.*, *Гасилов Н. А.* Модуль TOKSTAB (модуль библиотеки программ «Виртуальный токамак») // Вопросы атомной науки и техники. Сер. Термоядерный синтез. 2010. Вып. 3. С. 46–49.

[Хайрутдинов, Лукаш, 2008] *Хайрутдинов Р. Р.*, *Лукаш В. Э.* Программа расчета МГД-равновесия плазмы в токамаке DINA-SVD (модуль библиотеки программ «Виртуальный токамак») // Вопросы атомной науки и техники. Сер. Термоядерный синтез. 2010. Вып. 3. С. 87–89.

[Хайрутдинов, Лукаш, 2009] *Хайрутдинов Р. Р.*, *Лукаш В. Э.* Модуль DINA-EQDSK (модуль библиотеки программ «Виртуальный токамак») // Вопросы атомной науки и техники. Сер. Термоядерный синтез. 2009. Вып. 3. С. 57–59.

[Хайрутдинов, Лукаш, 2010] *Хайрутдинов Р. Р.*, *Лукаш В. Э.* Модуль DINA-Transp (модуль библиотеки программ «Виртуальный токамак») // Вопросы атомной науки и техники. Сер. Термоядерный синтез. 2010. Вып. 3. С. 50–54.

[Khayrutdinov, Lukash, 1993] *Khayrutdinov R. R.*, *Lukash V. E.* Studies of plasma equilibrium and transport in a tokamak fusion device with the inverse-variable technique // J. Comp. Phys. 1993. V. 109. N. 2. P. 193–201.

[Lukianitsa, Zaitsev, 2008] *Lukianitsa A. A.*, *Zaitsev F. S.* Advanced Methods for Analysis of Plasma Diagnostics Data // Proc. 8th Intern. FLINS Conf. Computational Intelligence in Decision and Control. Madrid, 12–24 Sept. 2008. P. 43–48.

[Lukyanitsa et al., 2007] *Lukyanitsa A. A.*, *Zaitsev F. S.*, *Shishkin A. G.*, *Nosov S. V.*, *Morozov A. V.*, *Zhdanov F. M.*, *Zlobin V. V.* Data mining methods in controlled thermonuclear fusion // 1st Korean-Russian Workshop on Data Mining. M.: MAX Press, 2007. P. 17–25.

[Lukyanitsa et al., 2008] *Lukianitsa A. A.*, *Zhdanov F. M.*, *Zaitsev F. S.* Analyses of ITER operation mode using the support vector machine technique for plasma discharge classification // Plasma Phys. Control. Fusion. 2008. V. 50. P. 1–14. Ref. N. 065013.

[Pereverzev et al., 1992] *Pereverzev G. V.*, *Yushmanov P. N.*, *Dnestrovskiy A. Yu.* et al. ASTRA, an Automatic System for Transport Analysis in a Tokamak: Preprint IAE-5358. Kurchatov Inst. Moscow, 1992.

STRUCTURE AND FUNCTIONAL CAPABILITIES OF IMITATION MODELING SYSTEM VIRTUAL TOKAMAK

F. S. Zaitsev [1,2], D. Y. Sychugov [1], A. G. Shishkin [1], V. E. Lukash [3], Y. V. Mitrishkin [4], R. R. Khayrutdinov [5], S. V. Stepanov [1], E. P. Suchkov [1]

[1] *Moscow State University*
[2] *Scientific Research Institute of System Development Russian Academy of Science*
[3] *Institute of Fusion Research "Kurchatov Institute" Russian Scientific Centre*
[4] *Institute of Control Problems Russian Academy of Science*
[5] *Troitsk Institute of Innovations and Thermonuclear Research*

The paper describes the concept of the software system "Virtual Tokamak", its functional capabilities, the main components and methods of usage. The role and place of the project in development of industrial application of controlled thermonuclear fusion are discussed. The problems of modern computer systems usage for integrated plasma simulation are considered. Requirements for computational systems and education are formulated. Russia, having acknowledged theoretical and computational school, can become the leader in tokamak simulations at the relatively small investments for production of the new innovative technologies in science and industry.

Key words: mathematical modeling, tokamak, software systems, controlled thermonuclear fusion.

Zaitsev Fedor Sergeyevich — professor, doctor of physical and mathematical sciences, e-mail: zaitsev@cs.msu.su.

Шишкин А. Г. — senior scientist, doctor of physical and mathematical sciences, e-mail: zaitsev@cs.msu.su.

Sychugov Dmitrii Yurevich — docent, candidate of physical and mathematical sciences, e-mail: sychugov@cs.msu.su.

Lukash Victor Emmanuilovich — senior scientist, professor, doctor of physical and mathematical sciences, e-mail: lukash@nfi.kiae.ru.

Mitrishkin Yuri Vladimirovich — senior scientist, professor, doctor of technical sciences, e-mail: yvm@mail.ru.

Khayrutdinov Rustam Rashitovich — senior scientist, doctor of physical and mathematical sciences, e-mail: khayrutd@mail.ru.

Stepanov Sergei Vladimirovich — PhD student, e-mail: sergey.v.stepanov@gmail.com.

Suchkov Egor Petrovich — аспирант, e-mail: suchkov.egor@gmail.com.

УДК 004.45

ТЕКУЩИЕ И ПЕРСПЕКТИВНЫЕ ТЕХНОЛОГИИ ВИРТУАЛИЗАЦИИ КАК ПЛАТФОРМА ДЛЯ ОРГАНИЗАЦИИ ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ: СРАВНЕНИЕ И АНАЛИЗ

В. В. Коноплев

Учреждение Российской академии наук
Институт космических исследований РАН (*ИКИ РАН*), *Москва*

В настоящей работе приводится классификация и обзорный сравнительный анализ существующих технологий виртуализации; аргументируются преимущества виртуализации как средства консолидации вычислительных ресурсов; рассматриваются причины потерь производительности в случае применения технологий виртуализации.

Работа выполнена при поддержке Президиума Российской академии наук (программа П-21).

Ключевые слова: виртуализация, облачные вычисления, Xen, OpenVZ.

ВИРТУАЛИЗАЦИЯ И ОБЛАЧНЫЕ ВЫЧИСЛЕНИЯ

Технологии облачных вычислений являются одним из основных трендов в области развития современных информационных технологий. И это неслучайно. Производительность вычислительного потока в современных процессорах выходит на свой технологический предел, и дальнейший рост вычислительной мощности происходит уже с увеличением числа процессорных ядер. В то же время программное обеспечение не успевает за растущим аппаратным параллелизмом вычислительных систем. Во многом это объясняется тем, что разработка программного обеспечения происходит преемственно, базируясь на старом программном коде, который зачастую не распараллеливается. Таким образом, возникает серьезная опасность того, что доступные вычислительные возможности окажутся незадействованными. Облачные вычисления, как одно из средств коллаборативного использования вычислительных ресурсов, позволяют эффективно решить эту проблему.

Текущий подход к построению облачных вычислений базируется на трех парадигмах: 1) «инфраструктура как сервис» (Infrastructure as a Service); 2) «платформа как сервис» (Platform as a Service) и 3) «программное обеспечение (ПО) как сервис» (Software as a Service) (рис. 1, см. с. 136). В рамках первой парадигмы пользователю «облака» доступны виртуальные элементы вычислительной инфраструктуры, где он может разворачивать образы виртуальных систем с пакетами приложений. В рамках второй парадигмы пользователю доступен программный стек, где он может разрабатывать и внедрять свои приложения. В рамках третьей парадигмы пользователю доступно только конечное приложение, на котором он может решать свои задачи.

Зачастую промежуточное программное обеспечение, управляющее «облаком», может работать с разными технологиями виртуализации.

Коноплев Вениамин Викторович — главный специалист, e-mail: v.konoplev@rssi.ru.

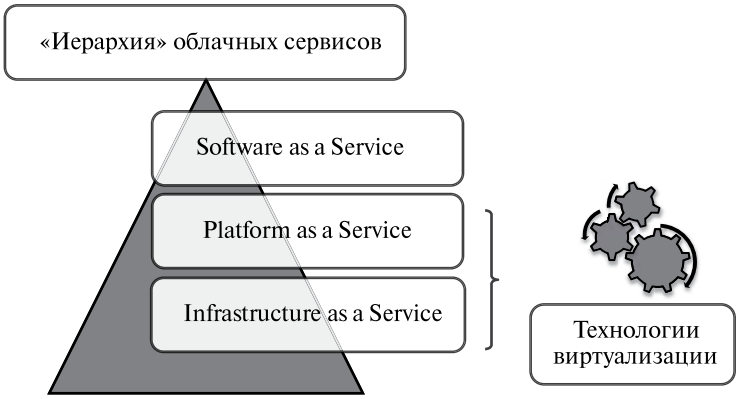


Рис. 1. Парадигмы облачных вычислений

В частности, пакет OpenStack (программное обеспечение с открытым исходным кодом для организации «облачных» вычислений: http://www.openstack.org/) предлагает системному администратору целых пять вариантов: KVM, UML, Xen (индустриальный стандарт для виртуализации с открытым исходным кодом: http://www.xen.org/), HYPER-V и QEMU.

Виртуализация непосредственно затрагивает два нижних уровня: «инфраструктура как сервис» и «платформа как сервис». Именно виртуализация является той технологической прослойкой, которая обеспечивает совместное, прозрачное и безопасное использование общих аппаратных ресурсов при построении облачных вычислений. Поэтому представление возможностей и особенностей данной технологии, способствующее выбору адекватного продукта, является залогом успеха конечного проекта.

ПРЕИМУЩЕСТВА ВИРТУАЛИЗАЦИИ

Далее будут рассмотрены некоторые преимущества, которые дает использование технологии виртуализации в центрах обработки данных (ЦОД).

Преимущества с точки зрения администратора ЦОД

Консолидация вычислительных ресурсов — виртуализация позволяет собрать на общей аппаратной платформе различные сервисы, которые в других случаях пришлось бы разносить по разным физическим вычислительным машинам. Например, публичные и закрытые сетевые службы по соображениям безопасности должны работать на разных системах независимо от нагрузки. Другой пример — приложения могут предъявлять разные, порой несовместимые требования к операционным системам.

Решение проблем безопасности — даже если обстоятельства допускают работу приложений в рамках одной операционной системы, их изоляция в рамках разных виртуальных серверов может значительно снизить уязвимость проекта.

Управляемость — использование виртуализации предоставляет администратору полный дистанционный доступ к гостевым системам, включая инсталляцию, остановку и пуск, что может быть особенно полезно для поддержки удаленных филиалов, а также аутсорсинга.

Преимущества с точки зрения конечного пользователя

Свобода выбора системного ПО и доступ к функциям администрирования — конечному пользователю предоставляется в распоряжение виртуальный сервер, позволяющий по желанию выбрать операционную систему, установить необходимые версии системных библиотек, добавлять и удалять других пользователей.

Предоставление гарантий на вычислительные ресурсы — текущие технологии виртуализации предоставляют конечным пользователям уровень изоляции с определенными гарантиями на вычислительные ресурсы — дисковое пространство, процессор и память.

КОНСОЛИДАЦИЯ РЕСУРСОВ — ЭКОНОМИЧЕСКИЕ ПРЕДПОСЫЛКИ

Виртуализация, как и любой промежуточный слой, неизбежно влечет за собой накладные расходы, что снижает конечную эффективную производительность вычислительной системы. Тем не менее, при использовании виртуализации отношение стоимости конечной системы к ее производительности может оказаться не хуже или даже лучше, чем без нее. Это становится возможным благодаря консолидации ресурсов. Виртуализация позволяет объединить несколько независимых логических вычислительных систем на общей высокопроизводительной аппаратной платформе. В этом случае, в силу статистического сглаживания, максимальная допустимая средняя загрузка системы может быть повышена. Далее это проиллюстрировано на модельном примере для семейства процессоров Intel Xeon.

Сводные данные по стоимости и индексу производительности процессоров фирмы Intel приведены в табл. 1.

Таблица 1

Характеристики процессоров Intel

Модель процессора	Стоимость, дол.	Индекс производительности
Intel Xeon X5680	1676	8991
Intel Xeon X5650	1003	7923
Intel Xeon E5630	554	4824

Рассмотрим некоторую модельную задачу, где максимальная производительность вычислительной системы, выраженная в запросах в секунду (K), пропорциональна индексу производительности процессора (I) и связана с ним следующей формулой:

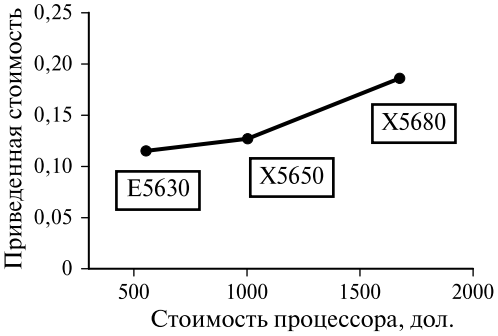


Рис. 2. Стоимость процессора, приведенная к его производительности

$$K = \frac{I}{100}. \qquad (1)$$

Предположим также, что поступающие запросы удовлетворяют распределению Пуассона $P(K, \lambda)$. Под перегрузкой будем подразумевать наступление события, при котором число поступивших запросов в единицу времени превышает производительность системы. Зададим вероятность перегрузки системы равной 0,1 % и получим соотношение для параметра λ, который соответствует средней загрузке вычислительной системы:

$$P(K, \lambda) = 0{,}999. \qquad (2)$$

Максимальная допустимая средняя загрузка системы будет определяться как отношение λ / K. Результаты приведены в табл. 2.

Таблица 2

Максимальная допустимая средняя загрузка системы

Модель процессора	**Максимальная загрузка *K*, 1/с**	**Средняя загрузка λ, 1/с**	**Допустимая загрузка λ/*K*, %**
Intel Xeon X5680	90	64	71
Intel Xeon X5650	79	55	69
Intel Xeon E5630	48	30	62

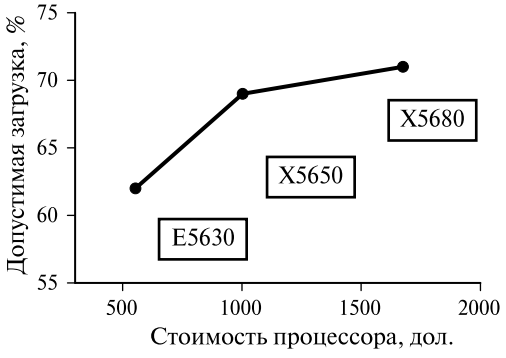


Рис. 3. Модельная максимальная допустимая загрузка системы

Далее показаны три производных графика:

1) стоимость процессора, приведенная к его индексу производительности (рис. 2);

2) модельная допустимая загрузка системы с данным процессором (рис. 3);

3) приведенная стоимость процессора с учетом допустимой загрузки (рис. 4, см. с. 139).

Как видно, каждая следующая единица в индексе производительности процессора обходится пользователю все дороже (см. рис. 2), что может способствовать выбору

процессоров низших ценовых категорий. Однако из-за статистического сглаживания в более мощных системах можно позволить себе более высокий допустимый средний уровень загрузки. Это обстоятельство позволяет в некоторых случаях получить оптимум при разработке бизнес-плана на этапе проектирования центра обработки данных. Так, в данном модельном случае оптимальным вариантом будет процессор X5650 (см. рис. 4), который занимает среднее положение из рассмотренных как по цене, так и по производительности.

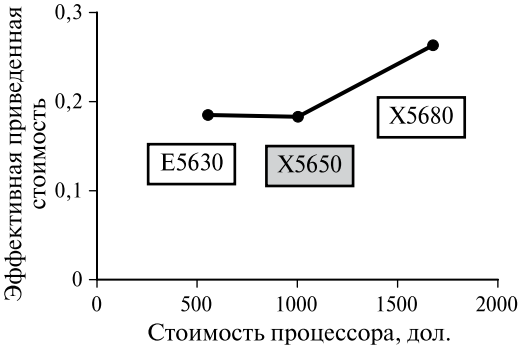


Рис. 4. Приведенная стоимость процессора с учетом допустимой загрузки системы

ХАРАКТЕРИСТИКИ ТЕХНОЛОГИЙ ВИРТУАЛИЗАЦИИ

Рассматривая технологии виртуализации, полезно выделить основные характеристики, которые потом можно будет использовать в качестве базиса для сравнения.

Управление доступом к ресурсам

Виртуализацию можно рассматривать как одно из средств организации доступа к аппаратным вычислительным ресурсам в рамках парадигмы облачных вычислений. В этом случае можно выделить две основные стратегии (рис. 5, см. с. 140):

1) безопасный конкурентный доступ к вычислительным ресурсам — наиболее подходит для научных приложений и позволяет эффективно использовать доступные ресурсы организаций;

2) изолированный доступ с резервированием и гарантиями — предпочтителен для бизнес-приложений, где пользователи ожидают получить определенные гарантии на заказанный ими продукт.

С технической точки зрения две описанные стратегии можно проиллюстрировать на примере доступа к оперативной памяти для систем виртуализации Xen и OpenVZ (контейнерная технология виртуализации для Linux: http://wiki.openvz.org/). Система Xen предполагает сегментацию оперативной памяти на части заданного размера и выделение этих частей гостевым системам. В случае OpenVZ, напротив, все гостевые системы пользуются общим пулом виртуальной памяти системы (оперативная память плюс своп); администратор может задавать ограничения на использование этого ресурса для гостевых систем, однако он не может ограничить использование именно оперативной памяти или зарезервировать ее за определенной гостевой системой.

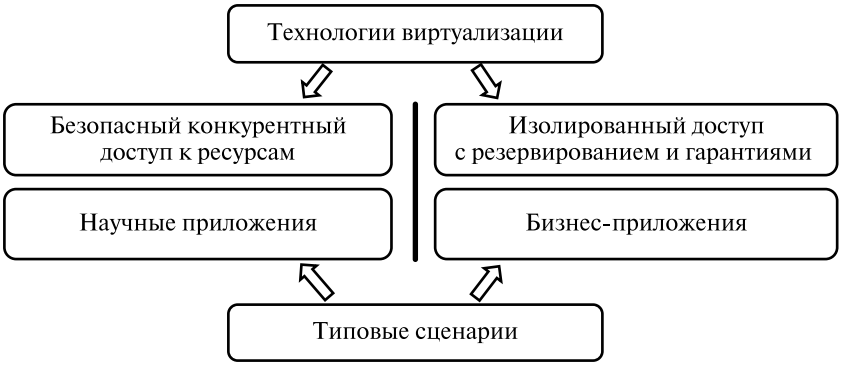


Рис. 5. Основные стратегии доступа к ресурсам

В настоящее время прослеживается определенная тенденция движения технологий виртуализации к комбинированной модели доступа к оперативной памяти: изоляция (включая резервирование) плюс конкурентный доступ к свободным ресурсам. Так, недавно появившаяся технология контейнеров в OS Linux-LXC (контейнеры для Linux: http://lxc.sourceforge.net), позволяет задавать ограничение на использование оперативной памяти со стороны гостевых систем, что можно считать шагом в сторону изоляции и резервирования ресурсов. С другой стороны, для системы XEN появилась технология ballon driver, изначально реализованная в VMware ESX [Waldspurger, 2002] и позволяющая перераспределять оперативную память между работающими гостевыми системами, что можно считать шагом в сторону конкурентного доступа. В системе XEN перераспределение памяти происходит вручную, запуском специальной команды. В то же время в системе VMware ESX эту функцию может брать на себя гипервизор, в случае нехватки оперативной памяти одной из гостевых систем.

Накладные расходы

Накладные расходы, неизбежно связанные с использованием виртуализации, обусловлены следующими факторами.

1. *Переключение контекста*. Наличие нескольких гостевых систем неизбежно приведет к увеличению переключений контекста процессора, которые могут оказать ощутимое влияние на производительность системы.

2. *Программная эмуляция*. В подавляющем большинстве технологий виртуализации процессор исполняет преимущественно «родной» (англ. native) программный код. Однако выполнение некоторых критических инструкций гостевых операционных систем гипервизор вынужден перехватывать специальными обработчиками (бинарная трансляция, описанная далее).

3. *Программная диспетчеризация ввода-вывода*. К сожалению, средства прямого ввода-вывода (DMA) обычно не могут работать напрямую с виртуальной памятью гостевых систем. Для этого необходима специальная аппаратная поддержка и соответствующая программная поддержка со стороны гипервизора.

4. *Дублирование программного кода*. В ходе работы пользовательских задач помимо кода приложения в системе выполняются коды ядра операционной системы и системных библиотек. При использовании виртуализации код ядра и системных библиотек дублируется между гостевыми доменами. Этот фактор, рассмотренный далее более подробно, может оказать существенное влияние на производительность системы.

Функциональные возможности

1. *Свобода выбора гостевой системы*. Некоторые технологии виртуализации предъявляют специальные требования к гостевым операционным системам, что ограничивает список возможных вариантов. Так, в технологиях виртуальных контейнеров требуется, чтобы гостевые домены работали на том же ядре, что и основная система. Технологии паравиртуализации требуют использования специальным образом модифицированных гостевых операционных систем.

2. *Возможности виртуализации оборудования*. Многие технологии виртуализации накладывают серьезные ограничения на виртуализованное аппаратное окружение, которое будет доступно гостевым операционным системам.

3. *Прямой доступ к оборудованию*. Некоторые системы виртуализации позволяют приписать определенные аппаратные устройства к определенным гостевым системам. Однако это требует дополнительной аппаратной поддержки со стороны хост-системы. В противном случае, устройство ввода-вывода не сможет получить прямой доступ (DMA) к памяти гостевой системы.

ВИДЫ ВИРТУАЛИЗАЦИИ

Технологии виртуализации можно разделить на следующие группы (рис. 6, см. с. 142):

1. *Контейнерные операционные системы*. Эти технологии с трудом можно назвать полноценными технологиями виртуализации. Они позволяют изолировать пространства имен операционной системы; благодаря этому у пользователя создается видимость того, что он работает на отдельной системе. В частности, пользователь гостевой системы видит только «свою» часть общей файловой системы и только «свои» процессы. В действительности, все гостевые операционные системы находятся под управлением общего ядра, что, естественно, снижает уровень безопасности. С другой стороны, контейнерные технологии обладают минимальными накладными расходами, которыми в большинстве случаев можно пренебречь.

2. *Технологии паравиртуализации*, использующие модифицированные гостевые операционные системы, в которых удалены или переписаны критические вызовы и команды. Кроме того, в гостевых операционных системах в данном случае отсутствуют полноценные драйверы, они заменены «заглушками», взаимодействующими с драйверами основной системы. Согласно большинству наблюдений, паравиртуализация занимает второе место по производительности после контейнерных технологий для систем, у которых отсутствует специальная аппаратная поддержка виртуализации.

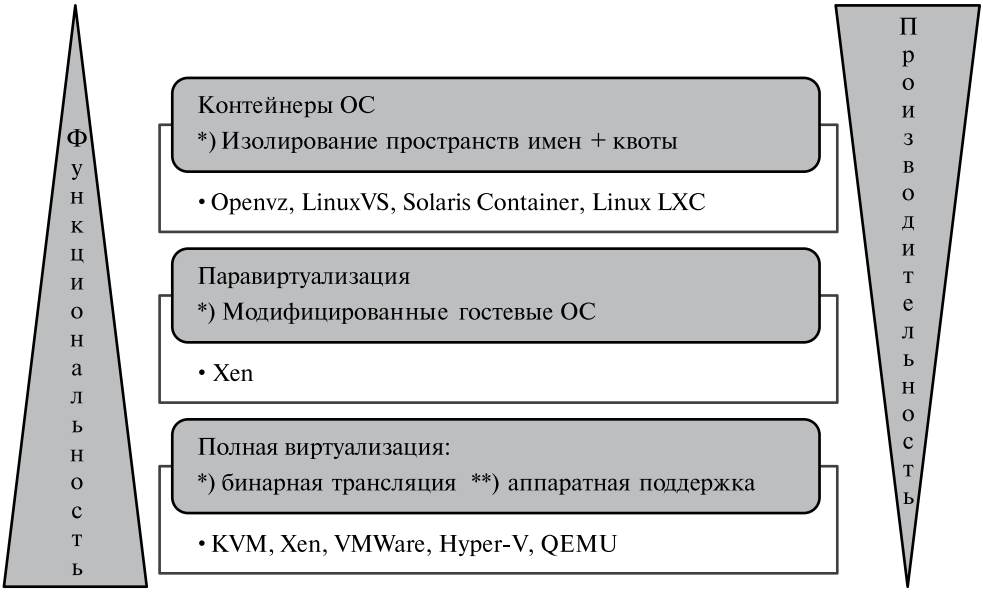


Рис. 6. Технологии виртуализации

3. *Полная виртуализация*, позволяющая запускать немодифицированные образы гостевых операционных систем. Для этой цели может использоваться одна из технологий:

а) бинарная трансляция, суть которой заключается в том, что гипервизор исследует запускаемый программный код и заменяет критические участки соответствующими обработчиками; это задерживает запуск кода, а также уменьшает скорость его выполнения;

б) аппаратная поддержка, включающая: 1) организацию дополнительных уровней безопасности процессора, которые позволяют ему перехватывать критические участки кода ядра гостевых систем; 2) поддержку диспетчеризации потоков ввода-вывода, которая позволяет драйверам гостевых операционных систем работать напрямую с оборудованием.

ДУБЛИРОВАНИЕ ИСПОЛНЯЕМОГО КОДА

В случае полной и паравиртуализации помимо пользовательского кода дублируются коды системных библиотек и ядра (рис. 7, см. с. 143). В случае контейнерных технологий дублируется только код системных библиотек. Некоторые контейнерные технологии позволяют избежать также дублирования кода системных библиотек. В частности, система виртуализации Virtuozzo (Parallels Virtuozzo Containers: http://www.parallels.com/products/virtuozzo/) в паре со специализированной файловой системой vzfs позволяет гостевым доменам использовать общий системный раздел.

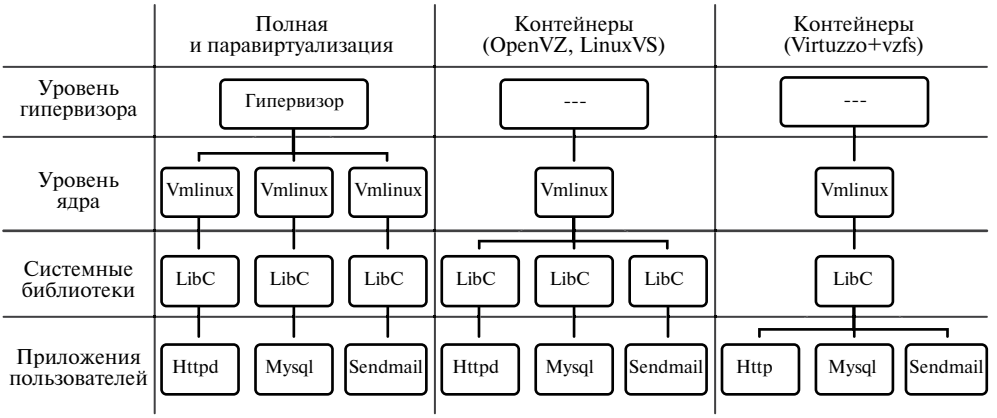


Рис. 7. Дублирование исполняемого кода для разных типов виртуалзации

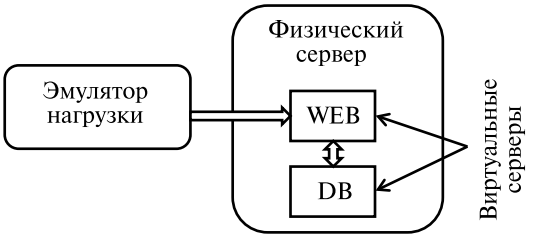


Рис. 8. Схема эксперимента

Основное негативное влияние эффекта дублирования кода связано не с расходом дискового пространства и оперативной памяти, как может показаться на первый взгляд, а с резко возрастающей нагрузкой на КЭШ центрального процессора.

В работе [Pradeep Padala et al., 2008] проводится подробный анализ данного эффекта. Далее воспроизведены некоторые наиболее характерные результаты, полученные авторами работы [Pradeep Padala et al., 2008]. На одном физическом сервере подняты две гостевые системы (рис. 8), на одной из них под управлением веб-сервера работает веб-приложение, написанное на PHP, которое в ходе выполнения запросов обращается к серверу системы управления базами данных (СУБД), расположенному на другой гостевой системе.

На рис. 9 (см. с. 144), заимствованном из работы [Pradeep Padala et al., 2008], показаны результаты анализа промахов в процессорном КЭШе при использовании разных технологий виртуализации (Xen и OpenVZ), а также для системы без виртуализации (базовый вариант). Как видно, число промахов при использовании OpenVZ незначительно отличается от базового варианта. В то же время при использовании XEN эта величина резко возрастает и, в основном, приходится на код ядра системы (колонка “Vmlinux”).

На практике увеличение промахов в процессорном КЭШе приводит к резкому падению производительности системы.

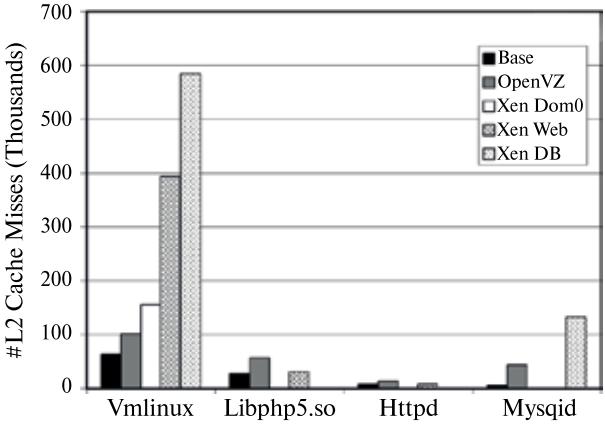


Рис. 9. Анализ «промахов» в КЭШе процессора (заимствован из работы [Pradeep Padala et al., 2008])

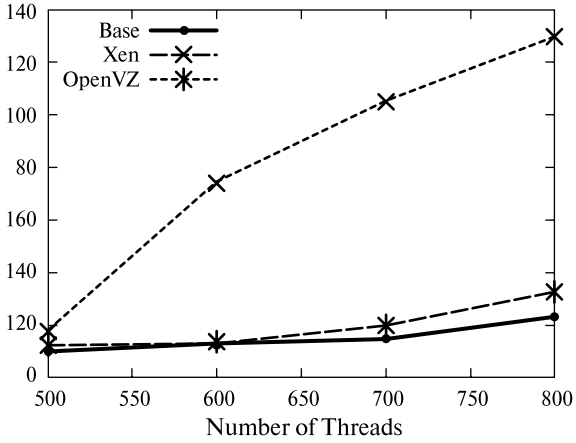


Рис. 10. Среднее время выполнения запроса в зависимости от загрузки (заимствован из [Pradeep Padala et al., 2008]).

На рис. 10, заимствованном из работы [Pradeep Padala et al., 2008], показано среднее время выполнения запроса в зависимости от нагрузки для трех вариантов: базовый, OpenVZ и Xen. Как видно, результаты для OpenVZ незначительно отличаются от базового варианта. Для Xen наблюдается увеличение времени выполнения запроса в несколько раз с ростом нагрузки.

ВИРТУАЛИЗАЦИЯ ВВОДА-ВЫВОДА

Виртуальные машины не имеют прямого доступа к физической памяти основной системы. Это одна из причин, которые затрудняют взаимодействие гостевых систем с устройствами ввода-вывода. Современное аппаратное обеспечение

поддерживает режим прямого доступа к памяти (DMA), который позволяет организовать потоки данных, минуя центральный процессор. Проблема в том, что обычное устройство DMA не имеет сведений об организации виртуальной памяти, а следовательно, о сегментах, которые выделены виртуальным серверам. Поэтому взаимодействие устройств ввода-вывода с гостевыми системами происходит через монитор виртуальных машин или гипервизор (рис. 11). Чтобы виртуальные машины могли работать непосредственно с устройствами ввода-вывода, минуя гипервизор, необходима специальная поддержка со стороны системы DMA. Примеры такой поддержки можно найти у ведущих производителей процессоров и наборов системной логики, в частности: Intel (технология VT-d) и AMD (технология IOMMU).

На рис. 12, заимствованном из статьи [Yaozu Dong et al., 2009], приведены данные по пропускной способности дисковых операций и загрузке процессора в зависимости от длины очереди запросов. Результаты даны для трех случаев:

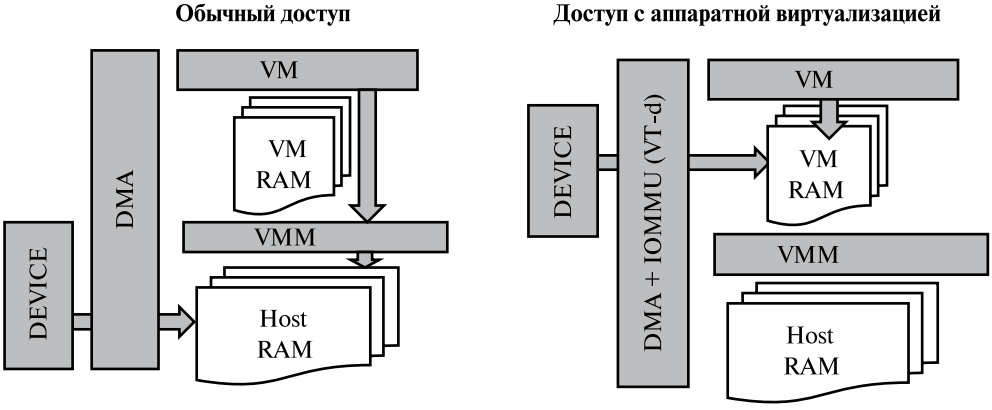


Рис. 11. Аппаратная поддержка прямого доступа к памяти для гостевых систем

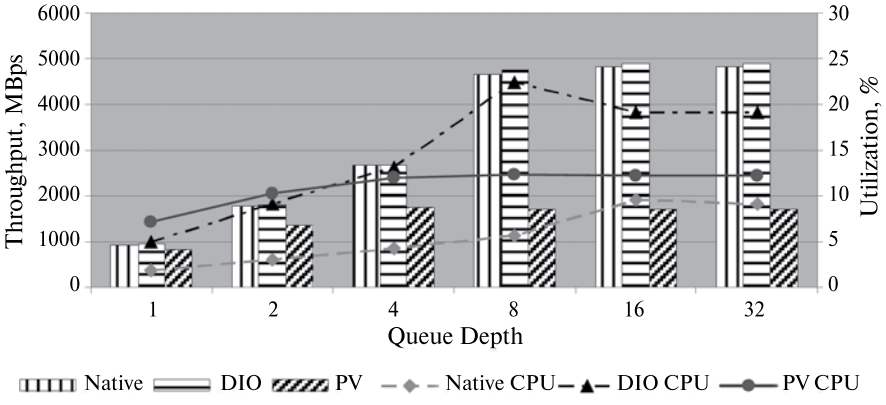


Рис. 12. Пропускная способность дисковых операций (заимствовано из работы [Yaozu Dong et al., 2009])

базовая система (Native), паравиртуализация (PV) и виртуализация с поддержкой прямого ввода-вывода (DIO). Как видно, поддержка прямого ввода-вывода виртуальных машин позволила в приведенном примере получить в два раза бóльшую пропускную способность, при этом загрузка центрального процессора оказалась в два раза ниже.

ЗАКЛЮЧЕНИЕ

В данной статье сформулирован ряд ключевых ориентиров, на которые следует обратить внимание при выборе технологии виртуализации для развертывания инфраструктуры облачных вычислений в центрах обработки данных. Приведенный обзор дает определенное представление о возможностях и ограничениях существующих технологий виртуализации, а также о возможных потерях производительности, с которыми может столкнуться конечный пользователь. Авторы надеются, что данный обзор будет полезен при выборе адекватной технологии.

Работа выполнена при поддержке Президиума РАН в рамках программы фундаментальных исследований «Проблемы создания национальной научной распределенной информационно-вычислительной среды на основе развития GRID технологий и современных телекоммуникационных сетей».

ЛИТЕРАТУРА

[Pradeep Padala et al., 2008] *Pradeep Padala*, *Xiaoyun Zhu*, *Zhikui Wang*, *Sharad Singhal*, *Kang G. Shin*. Performance Evaluation of Virtualization Technologies for Server Consolidation: Technical Rep. HP Laboratories. HPL-2007-59R1. Hewlett-Packard Development Company, L. P., 2008.

[Waldspurger, 2002] *Waldspurger C.A.* Memory Resource Management in VMware ESX Server // Proc. 5th Symp. Operating Systems Design and Implementation (OSDI'02). Dec. 2002.

[Yaozu Dong et al., 2009] *Yaozu Dong*, *Jinquan Dai*, *Zhiteng Huang*, *Haibing Guan*, *Kevin Tian*, *Yunhong Jiang*. Towards High-Quality I/O Virtualization // The Israeli Experimental Systems Conf. SYSTOR'09. May 4–6, 2009. Haifa, Israel. N.Y.: ACM Press, 2009. doi: 10.1145/1534530.1534547.

CURRENT AND PERSPECTIVE VIRTUALIZATION TECHNOLOGIES AS A PLATFORM FOR CLOUD COMPUTING: ANALYSIS AND COMPARISON

V. V. Konoplev

Space Research Institute (*IKI RAN*), *Moscow*

The paper presents classification and review for current virtualization technologies; we argue benefits of virtualization as a mean for computing power consolidation; some reasons for performance overhead concerning virtualization technologies are discussed.

Keywords: virtualization, cloud computing, Xen, OpenVZ.

Konoplev Veniamin Viktorovitch — chief specialist, e-mail: v.konoplev@rssi.ru.

УДК 519.7

ХАРАКТЕРИСТИКИ ПЕРСПЕКТИВНЫХ ПРИНЦИПИАЛЬНО НОВЫХ КОМПЬЮТЕРНЫХ УСТРОЙСТВ И СИСТЕМ

О. Н. Граничин

Санкт-Петербургский государственный университет (*СПбГУ*)

Обсуждаются возможные характеристики перспективных принципиально новых компьютерных устройств и систем. Новый компьютер представлен как устройство, состоящее из набора асинхронных динамических моделей (функциональных элементов). Основные черты: стохастичность, гибридность, асинхронность, отсутствие жесткой централизации, динамическая кластеризация на классы связанных моделей. Особое внимание уделено «гипотетической» модели искусственного интеллекта и реализации «за такт» оценки вектора-градиента многопараметрической функции на вычислительном устройстве типа квантового компьютера.

Ключевые слова: гибридные вычисления, стохастичность, оптимизация функционалов, искусственный интеллект.

ВВЕДЕНИЕ

Взглянем на историю развития средств вычислительной техники. За шесть десятилетий пройден путь от ламп, через транзисторы, интегральные микросхемы к сверхбольшим интегральным микросхемам. Что будет дальше? Основной вопрос — как обрабатывать данные? Для людей старшего поколения знакомство с компьютерами начиналось с фантастических романов. Еще школьниками, не видя никогда настоящих компьютеров, многие из них были уверены, что скоро появится «искусственный разум», освободив нас от многих рутинных забот. Наверное поэтому люди старшего поколения, занятые развитием информационных технологий (ИТ), острее чувствуют тенденции кардинальных преобразований. Конечно, кто-то возразит, что многие, не дождавшись за 60 лет «искусственного разума», разочаровались в перспективах, тем более что по мере развития ИТ, кроме такой фантастической цели, появилось огромное количество простых и сложных конкретных задач.

Новые потребности, глобализация задач, экспоненциальное возрастание сложности вычислительных систем и наметившаяся в последнее время тенденция по преодолению отставания отечественной ИТ-отрасли в развитии суперкомпьютерных вычислений («Т-Платформы», «СКИФ-Аврора» и другие проекты) заставляют уже в практическом плане задуматься о перспективах и возможной смене парадигмы: «Что такое процесс вычислений?» Объективные сегодняшние тенденции — миниатюризация и повышение производительности процессоров, как это и было предсказано законом Мура, — приводят технологии к порогу развития традиционных вычислительных устройств. От приоритетов бесконечного наращивания тактовой частоты и мощности одного процессора производители переходят к многоядерности, параллелизму и т. п.

Граничин Олег Николаевич — заведующий лабораторией, профессор, доктор физико-математических наук, e-mail: Oleg_granichin@mail.ru.

Сейчас несколько ядер в процессоре переносного компьютера уже норма, в процессорах суперкомпьютеров ядер уже намного больше. «Джин уже выпущен из бутылки», пройдет совсем немного времени и ядер станет несколько десятков, а потом и тысяч. Появятся совершенно другие архитектуры, ядра будут объединяться в сложные блоки, к данным можно будет получать параллельный одновременный доступ разным вычислительным блокам, «общение» вычислительных блоков между собой будет происходить через общую память. В действительности, изменятся многие аспекты парадигмы: что такое вычислительное устройство и что такое вычислительный процесс. Изменятся традиционные представления о том, как устроен компьютер, что такое вычислительная система. Эти процессы принесут изменения и в стиль программирования, и в то, как будут использоваться вычислительные устройства (ВУ).

1. НОВАЯ ПАРАДИГМА ВЫЧИСЛЕНИЙ

На прошедшей в сентябре 2010 г. в Абрау-Дюрсо Всероссийской научной конференции «Научный сервис в сети Интернет: суперкомпьютерные центры и задачи» (20–25 сентября 2010 г., г. Новороссийск) во многих докладах ставился вопрос: «Что будет при переходе от сегодняшних производительностей суперкомпьютеров в TeraFlops к следующему масштабу ExaFlops?» Вл. В. Воеводин (2010) писал: «Переход к ExaScale, естественно, должен будет затронуть такие важнейшие аспекты вычислительных процессов, как: модели программирования, степень и уровни параллельности, неоднородность программных и аппаратных систем, сложность иерархии памяти и трудности одновременного доступа к ней в распределенных вычислениях, стек системного и прикладного ПО, надежность, энергопотребление, сверхпараллельный ввод/вывод...» Все это неизбежно вызовет смену парадигмы высокопроизводительных вычислений.

Переход к новой парадигме вычислений приведет, наверное, к тому, что архитектура вычислительных устройств «сдвинется» в сторону «набора одновременно работающих асинхронных моделей взаимодействующих динамических систем (функциональных элементов)». Среди новых характерных черт будущей парадигмы все более отчетливо проступают следующие: стохастичность, гибридность, асинхронность, кластерность (отсутствие жесткой централизации и динамическая кластеризация на классы связанных моделей).

Стохастичность. С одной стороны, хорошо известно, что компьютеры становятся все миниатюрнее и миниатюрнее, размер элементарного вычислительного элемента (вентиля) приближается к размеру молекулы или даже атома. На таком уровне законы классической физики перестают работать и начинают действовать квантовые законы, которые в силу принципа неопределенности Гейзенберга не дают точных ответов о состоянии. С другой стороны, стохастичность — это известное свойство сложных динамических систем, состоящих из огромного числа компонент.

Под *гибридностью* будущих процессов вычислений понимается необходимость рассмотрения комбинации непрерывных и дискретных процессов, т.е. учет непрерывной эволюции протекания физических процессов при работе той

или иной модели и скачкообразное переключение с одной модели на другую. Увеличение быстродействия вычислительных устройств и уменьшение их размеров с неизбежностью приводит к необходимости операций с «переходными» процессами. Серьезным ограничением классической модели вычислений является разбиение памяти на изолированные биты, так как сокращение длины такта и расстояний между битами с определенного уровня делает невозможным рассматривать их изолированно в силу законов квантовой механики. Вместо примитивных операций с классическими битами в будущем было бы естественно перейти к операциям, задаваемыми теми или иными динамическими моделями микромира, оперирующими наборами взаимосвязанных «битов». При этом могут остаться классические операции с битами, для которых простейшими «моделями» являются процессы перехода физической системы (триггера) из состояния «1» в «0» и наоборот. Обоснованием целесообразности рассмотрения более широкого класса моделей являются успехи в разработках для традиционных сложных многомерных задач новых алгоритмов, работающих «за такт», в которых результат получается как итог физического адиабатического процесса. Например, П. Шор (1997) предложил алгоритм квантового преобразования Фурье, которое может выполняться за время, пропорциональное $(\log N)^2$, а не за $N\log N$ как классическое быстрое преобразование Фурье. В работе Д. Тиена (2003) обсуждается опирающийся на квантовую адиабатическую теорему гипотетически возможный «физический» способ решения за конечное время 10-й проблемы Гильберта, в работе [Вахитов и др., 2006] предложен эффективный квантовый алгоритм вычисления «за такт» оценки вектора-градиента многопараметрической функции, задаваемой с большой степенью неопределенностей (см. разд. 4). Типичные для математических алгоритмов операции типа «свертки» функций вполне могут обнаружиться «в природе». Последние исследования похожих моделей показывают, что их выполнение из-за присущей природе способности к самоорганизации не обязательно «раскладывается» на более простые кирпичики, т. е. не всегда может быть записано в виде классического алгоритма. Один из возможных примеров «аналоговой» реализации свертки функций на большом регулярном массиве квантовых точек с характерными размерами до 2 нм представлен автором совместно с С. Молодцовым (2006).

Асинхронность. Отказ от унифицированных простых вычислительных элементов неизбежно приводит к отказу от синхронизации работы различных компонент, имеющих существенно отличающиеся физические характеристики и свои длительности «тактов». Согласно классической теории множеств противоречивый смысл понятия единого «такта» выражается в рамках неразрешимости проблемы континуума в аксиоматике Френкеля – Цермело.

Кластерность. Одним из неожиданных результатов многочисленных попыток разработок (создания адекватного описания поведения и управления) сложных стохастических систем оказалась перспективность модели мультиагентных систем, в которой топология связей агентов между собой меняется со временем. При этом понятию агента может соответствовать как некоторая динамическая модель (компонент системы), так и определенный набор моделей. При отсутствии жесткой централизации такие системы способны эффективно решать достаточно сложные задачи, разбивая их на части и автономно перераспределяя

ресурсы на «нижнем» уровне, эффективность часто повышается за счет самоорганизации агентов и динамической кластеризация на классы связанных моделей.

В некотором смысле переход к разработке и созданию моделей сложных вычислений — закономерный этап развития микропрограммирования. Понимание возможных преимуществ работы устройств с эффективным микрокодом отмечалось еще в 50-е гг. прошлого века. Во втором ряду серии ЕС ЭВМ в конструкцию была заложена возможность динамического микропрограммирования. Хорошо известен целый ряд ЭВМ — от «Мир-1» до «Эльбрус», — использующих языки высокого уровня (HLL) как машинные. В настоящее время технологии микропрограммирования естественным образом сдвигаются в сторону физических процессов.

2. ОБОБЩЕНИЕ КЛАССИЧЕСКОЙ СХЕМЫ МАШИНЫ ТЬЮРИНГА

Автором [Граничин, Молодцов, 2006; Граничин, Васильев, 2010; Granichin, Vasil'ev, 2010] предложена и обоснована новая абстракция вычислительного устройства, обобщающая схему классической машины Тьюринга. В рамках новой модели переосмысливаются ставшие традиционными понятия «такта», «памяти», «ленты», «программы» и «состояния». Если в классическом подходе ячейки памяти используются для хранения дискретной информации и их изменение возможно только в тех случаях, когда указатель ленты показывает на нее, то в новой концепции ячейка памяти представляет собой постоянно функционирующую модель какой-то динамической системы (возможно и достаточно сложной), а лента — подмножество в общем пространстве состояний, при достижении которого заканчивается очередной такт и происходит включение программы — скачкообразное переключение с одних моделей на другие. Указатель «считывающей/записывающей головки» — бинарная строка, задающая список включенных моделей, вместе с матрицей видимости. Естественно, что классическая ячейка памяти для хранения бита является частным случаем такого обобщения.

В любых современных вычислительных устройствах хранение информации основано на тех или иных физических принципах, только эволюция состояния ячейки во время хранения более простая — сохраняется постоянной какая-то физическая характеристика. Другими словами, традиционную информатику можно сравнить с арифметикой, она оперирует цифрами — значениями ячеек памяти (или, более точно, — арифметикой конечных двоичных дробей). Предлагаемая новая модель вычислений позволит перейти в информатике от «арифметики» к «функциональному анализу», исследующему процессы эволюции информации внутри новых «ячеек» (операции с функциями).

Опишем математическую модель, соответствующую новой парадигме вычислительного устройства, рассматривая процесс вычисления как совокупность работающих параллельно динамических систем с пересекающимися частями фазовых пространств (общей памятью). Будем считать, что вычислительная система состоит из счетного набора аналоговых компонент $\{M_i\}_{i=1}^{\infty}$, таких что каждая компонента M_i определяет динамику некоторых компонент x_i общей памяти X

(имеющей некоторую гетерогенную природу) по правилу $\dot{x}_i(t) = H_i(x_i, t)$, где $x_i(t) \in X_i$, $X_i \subset X$.

Пусть в каждый момент времени t лишь небольшая часть моделей (компонент вычислительного устройства) оказывает влияние на изменение состояния памяти всей системы. Другими словами, в каждый момент времени t можно определить бесконечную строку битов $S(t) = \{s_i(t)\}_{i=1}^{\infty}$, состоящую из конечного числа единиц: $s_i(t) \in \{0, 1\}$ для $i \leqslant i_{\max}(t)$ и $s_i(t) = 0$ для $i < i_{\max}(t)$. Тогда общая динамика системы описывается как $\dot{x}(t) = \sum_{i=1}^{\infty} s_i(t) H(x_i, t)_i$, где $x(t) \in X$ и $x_i(t) \in X_i$.

Множество всевозможных строк $S(t)$ обозначим Σ.

Не умаляя общности, можно считать, что $S(t)$ — кусочно-постоянная функция времени, т.е. на определенных интервалах те или иные модели (компоненты ВУ) «включены». Таким образом, получаем гибридную динамическую систему в традиционном понимании этого термина. Общая динамика системы на k-м интервале постоянства $S(t)$ от t_k до t_{k+1} (со значением S_k) описывается как:

$$\dot{x}_{S_k}(t) = \sum_{i=1}^{i_{\max}(t_k)} s_i(t_k) H_i(x_i, t) = \sum_{i \in \sup p(S_k)} H_i(x_i, t), t \in [t_k, t_{k+1}),$$

где $x_{S_k}(t) \in \overline{X}_{S_k} = \bigcup_{i \in \sup p(S_k)} X_i \subseteq X$.

Для полного описания динамики всей системы необходимо определить правило изменения $S(t)$. Для описания процессов решения (или моделирования) большого числа практических задач достаточно ограничиться заданием направленного графа переходов, связывающего возможные переходы $S_k \xrightarrow{J_{k,j}} S_j$; $S_k, S_j \subset \Sigma$ с указанием условий $J_{k,j} \in J$, при выполнении которых происходит переключение изменения динамики всей системы с набора моделей S_k на S_j. Условием останова системы является $S(t) = 0$, которое эквивалентно тому, что ни один из компонентов памяти не является активным.

Таким образом, можно определить, что в новом смысле программа P — это правила изменения $S(t)$. Если правила хранить в памяти ВУ и дать возможность их корректировки со временем, то с точки зрения программирования это в некотором смысле будет соответствовать архитектуре фон Неймана, когда и данные, и программа хранятся в одной памяти и программа меняется в процессе работы системы.

Более общая схема получается при задании стохастических правил переходов. При этом детерминированный случай хорошо обобщается, задавая достаточно высокие вероятности для определенных в нем переходов и оставляя равновероятный, почти нулевой, уровень вероятности для всех остальных. Но даже такая схема может оказаться более эффективной (по аналогии с известным методом отжига для поиска глобального минимума функции). Случайно установленные «новые связи» могут оказаться полезными для системы и можно допустить корректировку вероятностей переходов в определенных ситуациях с целью

повышения их значимости. Такой механизм будет приводить к элементам самоорганизации моделей в процессе работы.

Вероятностное задание отображений эволюции и программ позволит реализовывать с помощью новой модели динамические, стохастические гибридные системы, вероятностные автоматы, системы со стохастическим управлением и т. п., не описываемые детерминированными законами. Для организации работы такой системы, наверное, целесообразно будет использовать рандомизированные алгоритмы, в которых один или несколько шагов базируются на случайном выборе одного из многих детермированных правил. Типичными трудностями при обработке информации в распределенных сетях и реальном времени являются ограничения ресурсов и недостаточная вариативность данных наблюдения. Рандомизация как раз и может быть использована для обогащения данных наблюдений, а рандомизированные сценарные подходы позволяют решать задачи «эффективно с высокой вероятностью» для почти всех ограничений, которых часто может быть очень много. Детальное описание истории развития и перспектив рандомизированных алгоритмов сделано автором совместно с Б. Т. Поляком в монографии «Рандомизированные алгоритмы оценивания и оптимизации при почти произвольных помехах» (2003), в которой подчеркиваются их лучшие и интересные черты:

- значительное сокращение числа операций (минимаксная оптимальность среди других алгоритмов);
- устранение влияния на работу системы систематических погрешностей, которые практически неизбежны при изменяющейся со временем модели динамической системы;
- независимость сложности алгоритма от размерности данных.

3. ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ

Вернемся к первому абзацу статьи. За более чем пятидесятилетнюю историю практического использования компьютеров одним из основных разочарований для человечества стала невозможность создания искусственного интеллекта. Несмотря на внушительный прогресс в сфере создания мощных вычислительных устройств, пока еще нельзя говорить о детальном понимании этой задачи или о направлениях исследований, которые могли бы гарантированно дать результат. Большая часть работ по искусственному интеллекту, известных автору, посвящена решению конкретных задач, или реализациям каких-либо эвристических поведенческих алгоритмов, эффективно справляющихся с тестовыми заданиями и терпящих неудачу на задачах реального мира. Многие исследователи согласны с тем, что пока еще не разработана методология создания алгоритмов и устройств для решения задач искусственного интеллекта. Разработка подобной методологии, по всей видимости, связана с анализом различных задач искусственного интеллекта, выявлением аналогичных частей и созданием новой формальной логики, адаптированной под эти части.

Человечество накопило большой массив задач, которые компьютеры умеют эффективно решать: можно послать ракету в определенную точку, быстро

вычислить преобразование Фурье или найти решение какого-нибудь важного уравнения (рис. 1). Но как поступить, если необходимо собрать универсальный вычислитель, который способен один выполнить их все? Предположим, что создаем не вычислитель, быстро решающий уравнения, а что-то похожее на людей (искусственное мыслящее существо), которое в условиях неопределенностей способно распознать реальную ситуацию, выбрать адекватную ей задачу и решить ее: например, среди всего реализованного набора задач выбирают блок, ответственный за решение определенной задачи, который говорит: «Да, эту ситуацию я контролирую, это моя цель, пора стрелять или вижу угрозу, пора убегать и т. п.»

Такие системы плохо вписываются в традиционную концепцию архитектуры компьютера, в которой операции обычно выполняются последовательно, данные загружаются последовательно, для выполнении того или иного действия надо последовательно пройти некоторые шаги А, Б, В и т. д., как-то их перебрать. Но пока мы их перебираем, зачастую решаемая задача перестает быть актуальной.

Как будет в перспективе? Простое решение собрать все вычислительные блоки вместе в сегодняшних условиях наткнется на проблемы отвода теплоты, одновременной доставки информации, выбора ведущего блока и многие другие. Наверное, когда-то станет возможным собрать блоки (микросхемы), решающие выбранные задачи (желательно все), в клубок закрученной спирали, как в молекулах ДНК в клетках (рис. 2, см. с. 154), который решит огромное число задач (функций).

Как могут решаться важные для универсального вычислителя задачи с доступом к памяти, параллелизмом, данными? Гипотетически можно представить, что химическое или электромагнитное воздействие (например, луч света) на такой «клубок» может одновременно воздействовать сразу на все блоки и каждый из них одновременно с другими примеряет поступающую информацию на себя. Традиционная альтернатива параллелизму — перебирать по очереди и смотреть, кому лучше подходит. В том блоке, который распознал адекватность текущей информации его задаче, которому информация подошла лучше остальных, можно представить себе возникновение состояния некоторого *информационного резонанса* (см. рис. 2*а*).

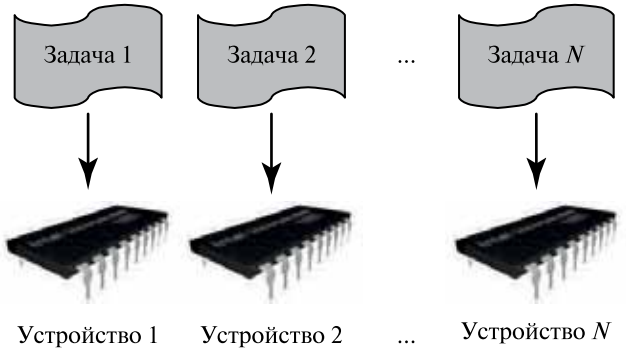


Рис. 1. Набор задач и решений

Рассмотрим пример интеллектуальной системы, работающей на перечисленных выше принципах. Представим себе робота, взаимодействующего с реальным миром через стандартные устройства — датчики, камеры, микрофоны, радиолокаторы, гусеницы, щупальца и т. д. Центральный компьютер робота состоит из тысяч устройств (блоков из моделей), обрабатывающих одни и те же данные, поступающие из внешнего мира, и создающих внутреннее представление мира в «мозгу» робота. Это внутреннее представление по очевидным причинам будет создаваться с помехами и постоянно корректироваться.

Центральный компьютер использует созданное представление мира для расчета оптимальной траектории, позволяющей роботу избежать опасностей и выполнить поставленное задание. Функционирование робота в подобных условиях можно представить как одновременное выполнение нескольких параллельных задач, контролируемых разными устройствами в центральном компьютере.

Следует отметить, что если резонансные устройства не конкурируют за общий ресурс (камеру, руку и т. д.), то нет необходимости выбирать из них главное.

Работа всех устройств робота (в том числе и вычислительных) регулируется вектором параметров, размерность которого может быть очень большой. Поиск оптимальных значений этих параметров возможен только после постановки задачи, включающей в себя определение функционала качества. В данном случае функционал качества будет оцениваться центральным компьютером. Основные возможные критерии — соответствие внутренней картины мира реальным условиям (число коррекций), число и качество информационных резонансов и т. д. Настройку параметра с высокой размерностью удобнее всего проводить рандомизированным алгоритмом стохастической оптимизации, описанным далее, который хорошо ложится на логику квантовых вычислительных устройств и по многим параметрам предпочтительнее других подобных алгоритмов для оптимизации систем в реальном времени.

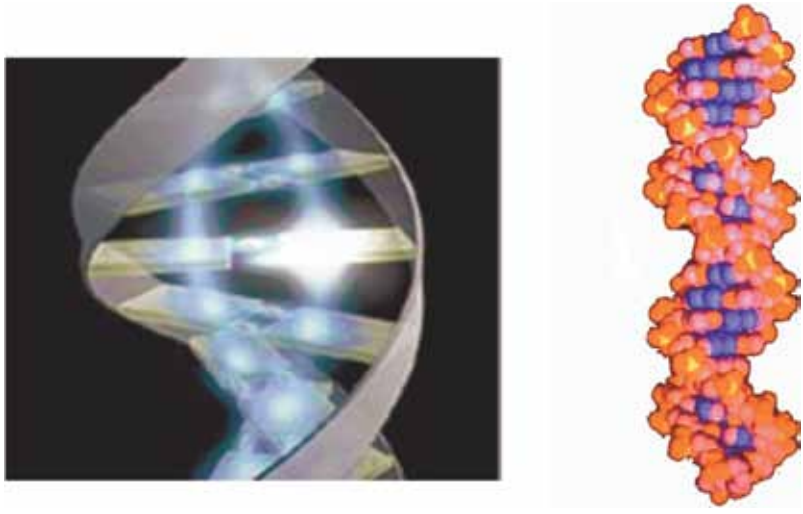


Рис. 2. Информационный резонанс (слева). Фрагмент молекулы ДНК (справа)

Более конкретный пример. Предположим, что есть набор устройств, способных распознавать определенные виды изображений (каждое из устройств настроено на какой-либо конкретный объект). Все эти устройства получают в реальном времени изображение с некоторой цифровой камеры. В каждый момент времени только одно из устройств, объявившее себя главным, может управлять камерой. Существует также одно устройство, управляющее камерой в случаях, когда никто не объявил себя главным. Оно может действовать по какому-нибудь простому алгоритму — например, осматриваться или сканировать некоторый важный участок, в котором ожидается появление объектов. Картинка с камеры постоянно подается на входы всех распознавателей, и те в свою очередь пытаются распознать в ней свой объект. Результатом их работы является степень совпадения в процентах (например, объект является кошкой с вероятностью 75 %). Под информационным резонансом естественно понимать превышение у какого-то из устройств «степени достоверности» заранее установленного порога (например, 90 %), в результате оно объявляет себя главным и берет на себя управление камерой для более детального изучения объекта или слежения за ним.

4. ВЫЧИСЛЕНИЕ «ЗА ТАКТ» ГРАДИЕНТА МНОГОПАРАМЕТРИЧЕСКОЙ ФУНКЦИИ

Рассмотрим задачу о выборе оптимального значения многомерного параметра сложной системы в условиях реального времени и оценке ее работы при неконтролируемых возмущениях.

Математически при достаточно общих предположениях эту проблему можно сформулировать как задачу о минимизации функции

$$f(x) = \mathrm{E}\{F(w, x)\big|\mathbf{w}\}$$

(типа функционала среднего риска), зависящей от контролируемого векторного q-мерного аргумента x и неконтролируемого случайного вектора $\mathbf{w}$, вообще-то говоря, с неизвестным распределением. Здесь $\mathrm{E}\{\cdot|\mathbf{w}\}$ — операция усреднения по распределению $\mathbf{w}$.

Если $f(\cdot)$ — непрерывно дифференцируемая функция, то необходимым условием того, что θ — точка минимума функции $f(\cdot)$, является равенство нулю в этой точке ее вектор-градиента $\nabla f(\theta) = 0$. Предположив, что известны значения вектор-градиента функции $f(\cdot)$ и матрицы ее вторых производных (гессиана), для нахождения точки минимума можно воспользоваться классической схемой вычислений по методу Ньютона:

$$\hat{\theta}_k = \hat{\theta}_{k-1} - \left[\nabla^2 f(\hat{\theta}_{k-1})\right]^{-1} \nabla f(\hat{\theta}_{k-1}),$$

где $n = 1, 2, \ldots$. Если матрица-гессиан $\nabla^2 f(\hat{\theta}_{k-1})$ в некоторой окрестности точки θ задает положительный ограниченный оператор и начальное значение $\hat{\theta}_0$ выбрано достаточно близко к точке локального минимума θ, то последовательность оценок $\{\hat{\theta}_k\}$ сходится к θ. Недостатком этого алгоритма является необходимость

обращать матрицу-гессиан на каждом шаге, что может представлять собой определенную трудность при большой размерности. В некоторых случаях удается выбрать рекуррентный способ для пересчета матриц, обратных гессиану. Для упрощения алгоритма их иногда обоснованно заменяют на положительные числа α_k, получая в результате алгоритм типа стохастической аппроксимации.

Если значения градиента функции $f(\cdot)$ не известны, то стандартным подходом к решению задачи является использование конечных разностей для аппроксимации градиента. Обозначим через $\mathbf{e}_i$ стандартный единичный вектор в направлении i-й координаты. Пусть $\{\beta_k\}$ — некоторая последовательность положительных чисел. В качестве аппроксимации i-й компоненты вектор-градиента можно использовать фурмулы

$$\nabla f(\hat{\theta}_{k-1})^{(i)} \approx \frac{f(\hat{\theta}_{k-1}+\beta_k \mathbf{e}_i)-f(\hat{\theta}_{k-1}-\beta_k \mathbf{e}_i)}{2\beta_k}.$$

Отметим, что этот стандартный подход к аппроксимации вектор-градиента требует на каждом шаге алгоритма оценивания провести $2q$ измерений значений минимизируемой функции при размерности искомого минимизирующего вектора, равной q.

Как поступить, если нельзя использовать в алгоритме не только градиент функции $f(\cdot)$, но и ее точные значения? Такая проблема возникает, если вид функций $f(\cdot)$и $F(\cdot, \cdot)$ известен не полностью, либо, если на вычисление соответствующих значений затрачивается чрезмерное количество усилий при дороговизне экспериментов или большой размерности вектора неизвестных параметров. Более того, в задачах оптимизации достаточно часто можно воспользоваться только зашумленной информацией о значениях функции $F(w, x)$ в выбираемых точках x с неконтролируемыми при этом значениями случайной величины w.

Существенно упростить процедуру оценки градиента и улучшить характеристики ее оценок позволяет включение одновременно в канал наблюдения, через выбираемый параметр, и в направление вектора изменения очередной оценки так называемого *пробного одновременного возмущения*. В отличие от классических конечно-разностных процедур при выборе очередной точки измерения функции случайному возмущению подвергаются одновременно все координаты.

Пусть $\{\mathbf{\Delta}_k\}$ — последовательность наблюдаемых, независимых бернуллиевских случайных векторов (координаты вектора $\mathbf{\Delta}_k$ не зависят друг от друга и принимают с равной вероятностью значения плюс/минус единица). Оказывается, что при зашумленных наблюдениях без существенных потерь в скорости сходимости для построения состоятельной последовательности оценок можно применить алгоритм, использующий всего два зашумленных измерения функции $F(\cdot, \cdot)$ на каждой итерации:

$$\hat{\theta}_k = \hat{\theta}_{k-1} - \alpha_k \Delta_k \frac{y_k^+ - y_k^-}{2\beta_k}, \quad y_k^{\pm} = F(w_k^{\pm}, \hat{\theta}_{k-1} \pm \beta_k \Delta_k) + v_k^{\pm}.$$

Более того, аналогичными свойствами обладает алгоритм с одним зашумленным наблюдением на каждой итерации:

$$\hat{\theta}_k = \hat{\theta}_{k-1} - \frac{\alpha_k}{\beta_k}\Delta_k y_k, \quad y_k = F(w_k, \hat{\theta}_{k-1} + \beta_k \Delta_k) + v_k.$$

В дальнейшем удобнее будет использовать другой вид алгоритма с одним измерением:

$$\hat{\theta}_k = \hat{\theta}_{k-1} - \gamma_k (x_k - \hat{\theta}_{k-1}) y_k, \quad x_k = \hat{\theta}_{k-1} + \beta_k \Delta_k, \quad y_k = F(w_k, x_k) + v_k, \quad \gamma_k = \frac{\alpha_k}{\beta_k^2}. \quad (1)$$

Эти рекуррентные процедуры называются *рандомизированными алгоритмами стохастической аппроксимации*, так как в их структуру неотъемлемой частью входит случайное пробное одновременное по всем координатам возмущение, которое также одновременно используется и в задании направления очередного изменения оценки и при выборе новой точки измерения. Иногда встречаются названия *стохастическая аппроксимация со случайными направлениями*, *поисковый алгоритм стохастической аппроксимации* или *стохастическая аппроксимация с возмущением на входе*. В англоязычной литературе широко используется название *одновременно возмущаемая стохастическая аппроксимация* (“simultaneous perturbation stochastic approximation”, SPSA).

Преимущества SPSA:

- асимптотически оптимальная скорость сходимости [Поляк, Цыбаков, 1990];
- минимальность числа измерений на итерации [Spall, 1992];
- состоятельность при почти произвольных помехах [Граничин, 1989];
- работоспособность в нестационарных задачах [Вахитов и др., 2009];
- легко может быть реализована на квантовом компьютере [Вахитов и др., 2006].

В работе [Граничин, Поляк, 2003] приведены точные условия, обеспечивающие состоятельность оценок рандомизированных алгоритмов стохастической аппроксимации, из которых наиболее существенным является условие о слабой коррелированности пробного возмущения $\{\Delta_k\}$ и последовательностей неопределенностей $\{w_k\}$ и $\{v_k\}$. Естественно, что среднеквадратичная скорость сходимости первого рандомизированного алгоритма с двумя измерениями обычно выше, чем у второго. Но стоит заметить, что в целом ряде практических задач оптимизации систем реального времени, обнаружения сигналов и адаптивного управления важно иметь возможность использовать алгоритм только с одним наблюдением на каждом шаге, так как в этих задачах трудно сделать не только $2q$ наблюдений, как в классических конечно-разностных процедурах, но недоступны даже два наблюдения с независимыми от Δ_k помехами. В отличие от оценивания по классическим конечно-разностным процедурам применение рандомизированных алгоритмов эффективно и при почти произвольных аддитивных помехах в наблюдении $\{v_k\}$. В подтверждение этого факта в случае неизвестной, но ограниченной детерминированной последовательности помех $\{v_k\}$ остановимся здесь только на неформальном объяснении, близком к доказательству в работе [Граничин, 1989].

Пусть дважды непрерывно дифференцируемая вещественная функция $f(X)$ вещественного аргумента X имеет в $\mathbf{R}^q$ единственный минимум в некоторой точке θ:

$$(X-\theta)^T \nabla f(X) \geqslant \mu \left\| X-\theta \right\|^2, \quad \forall X \in \mathbf{R}^q$$

с некоторой постоянной $\mu > 0$, и для градиента функции выполнено условие Липшица:

$$\left\| \nabla f(X) - \nabla f(\theta) \right\| \leqslant \left\| X-\theta \right\|, \ \forall X, \theta \in \mathbf{R}^q$$

с некоторой постоянной A. Выберем пробное одновременное возмущение Δ_k принимающим с равной вероятностью значения плюс/минус единица независимо от v_k.

Рассмотрим последовательность оценок $\{\hat{\theta}_k\}$, формируемых по рандомизированному алгоритму стохастической аппроксимации с одним измерением. Обозначим $D_k = \left\| \hat{\theta}_k - \theta \right\|^2$ и, учитывая центрированности Δ_k и независимость Δ_k от v_k, оценим условное математическое ожидание:

$$\mathrm{E}\left\{ D_k \mid \hat{\theta}_i, i<k \right\} \leqslant D_{k-1} - \frac{\alpha_k}{\beta_k}(\hat{\theta}_{k-1}-\theta)^T \mathrm{E}\left\{ \Delta_k y_k \mid \hat{\theta}_i, i<k \right\} + \frac{\alpha_k^2}{\beta_k^2}\mathrm{E}\left\{ \left\| \Delta_k \right\|^2 y_k^2 \mid \hat{\theta}_i, i<k \right\} =$$

$$= D_{k-1} - \frac{\alpha_k}{\beta_k}(\hat{\theta}_{k-1}-\theta)^T \mathrm{E}\left\{ \Delta_k f(\theta_{k-1}+\beta_k\Delta_k) \mid \hat{\theta}_i, i<k \right\} + \frac{\alpha_k^2}{q\beta_k^2}\mathrm{E}\left\{ y_k^2 \mid \hat{\theta}_i, i<k \right\}.$$

Разложив значение функции $f(\hat{\theta}_{k-1}+\beta_k\Delta_k)$ по формуле Тейлора, получим

$$f(\hat{\theta}_{k-1}+\beta_k\Delta_k) = f(\hat{\theta}_{k-1}) + \beta_k \Delta_k^T \ \nabla f(\hat{\theta}_{k-1}) + \beta_k^2 \zeta_k,$$

где ζ_k — некоторое число между $\hat{\theta}_{k-1}$ и $\hat{\theta}_{k-1}+\beta_k\Delta_k$ (вообще говоря, ζ_k — случайная величина). С учетом последней формулы, принимая во внимание центрированность Δ_k, выводим:

$$\mathrm{E}\left\{ D_k \mid \hat{\theta}_i, i<k \right\} \leqslant (1+\frac{1}{2}\alpha_k\beta_k)D_{k-1} - \alpha_k(\hat{\theta}_{k-1}-\theta)^T \nabla f(\hat{\theta}_{k-1}) + \xi_k,$$

где $\xi_k = \mathrm{E}\left\{ \dfrac{\alpha_k\beta_k\zeta_k^2}{2} + \alpha_k^2\beta_k^{-2}y_k^2 \mid \hat{\theta}_i, i<k \right\}$. Отсюда, из условия сильной выпуклости функции $f(\cdot)$, видно, что последовательность $\{D_k\}$ — почти супермартингал:

$$\mathrm{E}\left\{ D_k \mid \hat{\theta}_i, i<k \right\} \leqslant (1-\gamma_k)D_{k-1} + \xi_k,$$

где $\gamma_k = \mu\alpha_k - \dfrac{\alpha_k\beta_k}{2}$. Если предположить, что для числовых последовательностей $\{\alpha_k\}$ и $\{\beta_k\}$ выполняются условия

$$\sum_k \alpha_k = \infty, \quad \beta_k \to 0, \quad \sum_k \frac{\alpha_k^2}{\beta_k^2} < \infty, \quad \sum_k \alpha_k\beta_k < \infty,$$

то выполнены все условия леммы Роббинса – Сигмунда (1971) о сходимости почти супермартингалов. Таким образом, последовательность оценок $\{\hat{\theta}_n\}$ сходится к θ с вероятностью единица. При этом дисперсия ошибки оценивания прямо пропорциональна α_n^2/β_n^2.

Рассмотрим проблему выбора вычислительного устройства, наилучшим образом подходящего для выполнения рандомизированного алгоритма стохастической оптимизации с одним измерением функции штрафа на итерации. Предположим, что разрядность используемого квантового вычислителя равна n. Пусть на самом деле задан квантовый «черный ящик», который вычисляет $f(x)$. Более точно, пусть определен унитарный оператор, реализующий на квантовом компьютере функцию $f(x)$, который задан на всех определяющих аргумент функции классических двоичных цепочках x длины qn и «управляющей» произвольной двоичной цепочке z длины n, так:

$$U_f : |x\rangle|z\rangle \to |x\rangle|z \oplus f(x)\rangle,$$

где $\oplus$ — побитовая операция «логического или». Это способ задания оператора на базисных векторах. На все остальные векторы оператор продолжается линейно. Легко видеть, что построенный оператор обратим и действует в комплексном пространстве размерности 2^{qn+n}.

Рассмотрим квантовую цепь, изображенную на рис. 3, которая позволяет получить следующую оценку точки минимума функции по рандомизированному алгоритму стохастической аппроксимации с одним измерением (1) за «один такт».

Для подачи на вход вычислителя первоначально подготавливается суперпозиция 2^q возмущенных значений текущего вектора оценки:

$$x_k = H_\beta |\hat{\theta}_{k-1}\rangle = \frac{1}{2^{q/2}} \sum_{\Delta_i \in \{-1,+1\}^q} |\hat{\theta}_{k-1} + \beta_k \Delta_i\rangle,$$

где ± 1 рассматриваются как n-разрядные числа. После применения унитарного оператора U_f к $|x_k\rangle|0\rangle$ получаем

$$U_f |x_k\rangle|0\rangle = \frac{1}{2^q} \sum_{\Delta_i \in \{-1,+1\}^q} |\hat{\theta}_{k-1} + \beta_k \Delta_i\rangle |f(\hat{\theta}_{k-1} + \beta_k \Delta_i)\rangle.$$

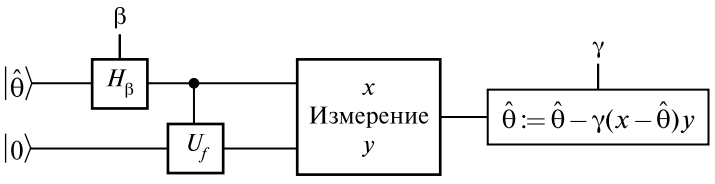


Рис. 3. Вычисление «за такт» градиента многопараметрической функции

Из общих свойств модели квантовых вычислений следует, что после измерения полученного состояния, с вероятностью $1/2^q$ будет определен один из векторов вида

$$\left|\hat{\theta}_{k-1}+\beta_k\Delta_i\right\rangle\left|f(\hat{\theta}_{k-1}+\beta_k\Delta_i)\right\rangle,\quad \Delta_i\in\{-1,+1\}^q.$$

Из первых qn разрядов этого вектора легко получить реализацию вектора случайного возмущения Δ_i. По рандомизированному алгоритму оценивания координаты вектора Δ_i надо умножить на соответствующее значение функции потерь в возмущенной точке, которое записано в последних n разрядах полученного после измерения результата

$$\left|\hat{\theta}_{k-1}+\beta_k\Delta_i\right\rangle\left|f(\hat{\theta}_{k-1}+\beta_k\Delta_i)\right\rangle,\quad \Delta_i\in\{-1,+1\}^q.$$

ЗАКЛЮЧЕНИЕ

Описанная новая модель вычислений позволяет описывать если не все, то подавляющее большинство процессов реального мира, а также работу всевозможных существующих и будущих вычислительных устройств, включая аналоговые и био-, нейро- и квантовые компьютеры и т. д. Особенностью предлагаемого подхода является отказ от редукции сложности в процессе вычисления. Сложность вычислимого объекта должна быть эквивалентна сложности вычисляемого. Другими словами, понятие вычислительной сложности правильнее рассматривать относительно выбранной системы базисных эволюционных примитивов, а не относительно традиционно рассматриваемых битовых преобразований {0; 1}. Квантовые и нейрокомпьютеры обещают сильно изменить представления о вычислительной мощности современных вычислительных устройств. Увеличение вычислительной мощности, возможное при использовании новых моделей вычислений, основывающихся на физических явлениях, позволяет предположить, что в будущем новые компьютеры смогут решать задачи, не выполнимые для обычных компьютеров.

ЛИТЕРАТУРА

[Вахитов и др., 2006] *Вахитов А. Т.*, *Граничин О. Н.*, *Сысоев С. С.* Точность оценивания рандомизированного алгоритма стохастической оптимизации // Автоматика и телемеханика. 2006. № 4. С. 86–96.

[Вахитов и др., 2009] *Вахитов А. Т.*, *Граничин О. Н.*, *Гуревич Л. С.* Алгоритм стохастической аппроксимации с пробным возмущением на входе в нестационарной задаче оптимизации // Автоматика и телемеханика. 2009. № 11. С. 70–79.

[Воеводин, 2010] *Воеводин В. В.* Современные суперкомпьютерные технологии: лекция // Всерос. науч. конф. «Научный сервис в сети Интернет: суперкомпьютерные центры и задачи». 20–25 сент. 2010, Новороссийск.

[Граничин, 1989] *Граничин О. Н.* Об одной стохастической рекуррентной процедуре при зависимых помехах в наблюдении, использующей на входе пробные возмущения // Вестн. Ленингр. ун-та. 1989. Сер. 1. Вып. 1. С. 19–21.

[Граничин, Васильев, 2010] *Граничин О. Н.*, *Васильев В. И.* Гибридная модель процесса вычислений: обобщение концепции машины Тьюринга // Нейрокомпьютеры: разработка, применение. 2010. № 6. С. 51–58.

[Граничин, Молодцов, 2006] *Граничин О. Н.*, *Молодцов С. Л.* Создание гибридных сверхбыстрых компьютеров и системное программирование. СПб., 2006. 108 с.

[Граничин, Поляк, 2003] *Граничин О. Н.*, *Поляк Б. Т.* Рандомизированные алгоритмы оценивания и оптимизации при почти произвольных помехах. М.: Наука, 2003. 291 с.

[Поляк, Цыбаков, 1990] *Поляк Б. Т.*, *Цыбаков А. Б.* Оптимальные порядки точности поисковых алгоритмов стохастической аппроксимации // Проблемы передачи информации. 1990. № 2. С. 45–53.

[Granichin, Vasil'ev, 2010] *Granichin O. N.*, *Vasil'ev V. I.* (2010) Computational model based on evolutionary primitives. Turing machine generalization // Intern. J. Nanotechnology and Molecular Computation. 2010. V. 2. N. 2. P. 30–43.

[Robbins, Siegmuud, 1971] *Robbins H.*, *Siegmuud D.* A convergence theorem for nonnegative almost super-martingales and some applications // Optimizing Methods in Statistics / Ed. Rustagi J. S. N. Y.: Academic Press, 1971. P. 233–257.

[Shor, 1997] *Shor P.* Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer // SIAM J. Computing. 1997. V. 26. P. 1484–1509.

[Spall, 1992] *Spall J. C.* Multivariate stochastic approximation using a simultaneous perturbation gradient approximation // IEEE Trans. Automatic Control. 1992. V. 37. P. 332–341.

[Tien, 2003] *Tien D. K.* Computing the non-computable // Contemporary Physics. V. 44. P. 51–71.

PRINCIPLE FEATURES OF PERSPECTIVE NEW COMPUTER SYSTEMS AND DEVICES

O. N. Granichin

Saint Petersburg State University

We discuss the possible characteristics of future fundamentally new computing devices and systems. New computer appears as a device consisting of a set of asynchronous dynamic models (functional elements). Key features: stochastic, hybrid, asynchronous, absence of rigid centralization, dynamic clustering for classes of related models. Particular attention is paid "Hypothetical" model of artificial intelligence and the implementation "Per cycle" estimates of the vector gradient of multiparameter functions on the computing device such as a quantum computer.

Keywords: hybrid computation, stochastics, functional optimization, artificial intelligence.

Granichin Oleg Nikolaevich — head of the laboratory, professor, doctor of physical and mathematical sciences, e-mail: Oleg_granichin@mail.ru.

УДК 004.75, 681.518

РАСПРЕДЕЛЕННОЕ ДЕТЕКТИРОВАНИЕ СОБЫТИЙ В МНОГОМЕРНЫХ ПОТОКАХ ДАННЫХ В БЕСПРОВОДНЫХ СЕНСОРНЫХ СЕТЯХ

А. А. Пойда [1,2], М. Н. Жижин [1,2], Д. П. Медведев [1,2], А. Е. Москвитин [3], А. В. Андреев [1,2]

[1] *Учреждение Российской академии наук Геофизический центр РАН (ГЦ РАН), Москва*
[2] *Учреждение Российской академии наук Институт космических исследований РАН (ИКИ РАН), Москва*
[3] *Институт космофизических исследований и распространения радиоволн ДВО РАН (ИКИР ДВО РАН), Камчатский край, пос. Паратунка*

В данной статье разработан и реализован на действующем макете комплекс программного обеспечения для сбора данных и детектирования погодных, сейсмических и геоакустических событий на узлах пространственно-распределенной беспроводной сети.

Результаты получены при финансовой поддержке РФФИ, проекты № 10-07-00682-а и 10-07-90711-моб_ст. Авторы статьи выражают благодарность Александру Годунову за помощь при подготовке материалов статьи.

Ключевые слова: беспроводные сенсорные сети, мониторинг, детектор событий, потоковая обработка данных, геоакустика, вулканология, сейсмология.

ВВЕДЕНИЕ

В данной работе создан и реализован на действующем макете комплекс программного обеспечения для сбора данных и детектирования погодных, сейсмических и геоакустических событий на узлах пространственно-распределенной беспроводной сети.

За рубежом исследования в этой области и с применением беспроводных сенсорных сетей активно ведутся уже десять лет. Разработан и внедрен в мелкосерийное производство целый ряд платформ, использующих разные микропроцессоры, наборы сенсоров и протоколы радиообмена. Наиболее распространенными платформами на сегодня являются семейства mica (micaZ*, mica2**) и Telos (TelosA, TelosB)***.

Пойда Алексей Анатольевич — старший научный сотрудник, кандидат физико-математических наук, e-mail: poyda@wdcb.ru.

Жижин Михаил Николаевич — заведующий лабораторией сетевых информационных технологий, кандидат физико-математических наук, e-mail: jjn@wdcb.ru.

Медведев Дмитрий Петрович — научный сотрудник, e-mail: dmedv@wdcb.ru.

Москвитин Андрей Евгеньевич — младший научный сотрудник, e-mail: moskvitin.andrey@ikir.ru.

Андреев Александр Викторович — инженер, e-mail: and@wdcb.ru.

* http://www.openautomation.net/uploadsproductos/micaz_datasheet.pdf.

** http://arri.uta.edu/acs/ee5369/ee5369%20lectures/Introduction%20to%20Crossbow%20Mica2%20Sensors.pdf.

*** http://www2.ece.ohio-state.edu/~bibyk/ee582/telosMote.pdf.

Различаясь по схемотехнике, они используют общий протокол обмена Zigbee [Shahin Farahani, 2008], который обеспечивает энергетическиэффективный многохоповый радиообмен короткими сообщениями между узлами сенсорной сети с динамической маршрутизацией, выбирающей пути доставки сообщений с учетом работоспособности промежуточных узлов. Дальность связи между узлами сильно зависит от условий их размещения (под землей, на деревьях, в помещении), но в среднем имеет порядок 10 м. Поэтому для мониторинга протяженных объектов требуется либо большое число узлов, либо наличие промежуточных Wi-Fi-мостов.

Для многопотокового программирования задач обработки данных в реальном времени на энергосберегающих микропроцессорах в университете Беркли разработаны специальная операционная система TinyOC [Hill et al., 2000] и расширение языка C nesC [Gay et al., 2003; Levis, Gay, 2009], позволяющее компилировать на Windows- и Linux-платформах многозадачные приложения для TinyOS и размещать их в типичных 8 КБ памяти микропроцессора для обработки 512 КБ буфера данных. Эти задачи включают оцифровку нескольких входных сигналов с сенсоров (например, температура, освещенность и т. п.), циклическую буферизацию и осреднение данных, простейший их анализ, формирование сообщения и передачу данных по сети в центр сбора.

Большинство технических проблем, включая протокол радиообмена данными между узлами сети, операционную систему и программное обеспечение микропроцессоров, базовые сенсоры (освещенности, влажности, давления, температуры), можно считать на сегодня решенными. При этом на рынке имеются несколько программно-аппаратных решений со сравнимыми характеристиками в диапазоне цен 20...200 дол. за узел. Технологически трудными остаются создание всепогодных корпусов, стоимость которых может составлять до 90 % от стоимости узла, оптимизация соотношения скорость и объем обмена данными (десятки герц по нескольким каналам) и продолжительность автономной работы (1...2 года), и создание компактных специализированных датчиков, например для химического анализа воды и атмосферы.

Оставляя в стороне приложения сенсорных сетей для систем безопасности и военных технологий, назовем несколько недавних проектов по мониторингу окружающей среды. С 2005 г. в Университете Джона Хопкинса в Балтиморе совместно с Исследовательским центром по базам данных Майкрософт в Сан-Франциско реализуется проект Life Under Your Feet: Environmental Sensing for Soil Ecology* (руководитель Alex Szalay [szalay@jhu.edu]). В рамках проекта разработаны специальные сенсоры для контроля экологии почв, программное обеспечение для сбора данных, специальная схема базы данных для хранения и индексирования наблюдений, и накоплен большой объем (сотни гигабайт) непрерывных наблюдений с нескольких сенсорных сетей на восточном побережье США. В Европе достаточно известен Swiss Experiment: a collaborative cyberinfrastructure for environmental scientists** (руководитель Nicholas Dawes [dawes@slf.ch]). Швейцарский эксперимент аналогичен по структуре проекту Университета

* http://lifeunderyourfeet.org/en/.

** http://www.openautomation.net/uploadsproductos/micaz_datasheet.pdf.

Джона Хопкинса, но превосходит его по числу предметных областей для приложений, поддерживаемых сенсорных платформ и набору предоставляемых сервисов по доступу и визуализации данных.

В настоящей работе внимание сосредоточено на алгоритмах и программном обеспечении для сбора и анализа данных.

Задачу сбора и анализа данных можно разбить на две части:

1) высокочастотные наблюдения в реальном времени;

2) низкочастотные наблюдения с буферизацией данных.

В первой задаче нельзя передавать все данные, так как это превысит пропускную способность сети. Здесь требуется использовать локальную обработку получаемых данных непосредственно на узлах, и передавать данные либо частично, фильтруя их по важности, либо вообще отказаться от передачи сырых данных, заменив ее передачей результатов обработки. Кроме того, локальная обработка позволяет перенести вычислительную нагрузку начальной стадии анализа данных с центрального сервера на узлы-детекторы, что приводит:

1) к повышению масштабируемости (т.е. можно увеличить число детекторов в сети);

2) увеличению срока работы детекторов без подзарядки и снижению потребляемой мощности;

3) повышению отказоустойчивости вследствие потерь данных.

Во второй задаче пропускная способность сети позволяет передавать весь объем данных в реальном времени. Но в таком варианте требуется постоянная работа радиопередающей аппаратуры узла, а это — самый энергозатратный ресурс. При этом сильно сокращается время автономной работы узлов. Чтобы избежать энергозатрат и снизить стоимость передачи данных, можно использовать локальную обработку данных на узле как при высокочастотных наблюдениях. Но есть и возможность отказаться от передачи данных в реальном времени и использовать их буферизацию непосредственно на узлах с последующей передачей. Отрицательная сторона этого варианта — задержка между получением данных и их передачей в центральную систему. Чтобы обеспечить гибкость при работе с беспроводной сетью, была поставлена цель совместить обе задачи и разработать комплекс программного обеспечения для **сбора и буферизации данных** и **детектирования** погодных, сейсмических и геоакустических событий на узлах высокопроизводительных многосенсорных беспроводных сетей.

Решение поставленной задачи потребовало согласованной работы нескольких системных компонент:

1) макета беспроводной сети сбора и предварительной обработки данных, оснащенной измерительными сенсорами, используемыми в сейсмологии, геоакустике, при анализе погоды;

2) системы долгосрочного хранения и обработки наблюдений;

3) промежуточного программного обеспечения, реализующего распределенные детекторы пространственно-временных событий, а также буферизацию, маршрутизацию и синхронизацию узлов сети;

4) системы для интерактивной визуализации данных наблюдений и событий в сенсорных сетях.

Рис. 1. Внешний вид модуля беспроводной сенсорной сети TelosB

В ходе разработки системы ориентировались на сенсоры TelosB производства американской компании Crossbow (рис. 1). На материнской плате сенсора установлены процессор Texas Instruments MSP430, радиотракт 2,4 МГц, оперативная память 512 КБ, порт USB для связи с компьютером, и базовый набор сенсоров напряжения питания, температуры и относительной влажности воздуха, освещенности в видимом и ближнем инфракрасном спектре. Более подробные технические характеристики узла приведены в спецификации Crossbow TelosB datasheet*, а также в приложении «Спецификация узла беспроводной сенсорной сети TelosB». Время автономной работы сенсора от двух AA-батарей зависит от продолжительности и мощности радиообмена и в среднем составляет несколько недель. Однако в случае использования сенсорной сети для сбора низкочастотных наблюдений с буферизацией данных на узлах удается обеспечить облегченный режим работы, сохраняющий заряд батарей на более продолжительное время.

В ходе работы базовый набор сенсоров был расширен термопарами, инфразвуковыми микрофонами, акселерометрами.

Термопары с измерительным порогом до 400 °C были подключены к платформе TelosB на основе специализированной микросхемы MAX6675 (рис. 2). Внешний АЦП соединен с платой TelosB пятью каналами: заземления, управляющим для передачи команды инициации процесса получения очередного значения, тактовым управляющим для контроля скорости передачи данных, цифровым

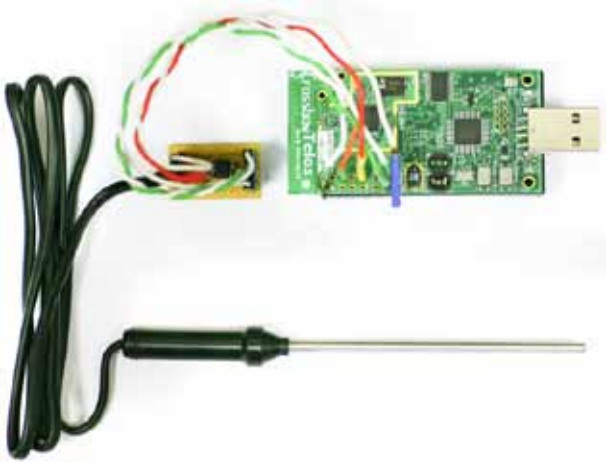


Рис. 2. Платформа TelosB, оснащенная термопарой

* http://www.openautomation.net/uploadsproductos/micaz_datasheet.pdf.

каналом получения данных, запитки устройства. Для работы с термопарами был написан программный модуль на языке NesC.

Инфразвуковой сенсор собран на основе аналогового операционного усилителя и RC-фильтра низких частот с порогом 30 Гц. Сенсор подключен непосредственно к платформе TelosB по двум каналам: заземления и передачи данных (рис. 3). Сенсор вырабатывает аналоговый сигнал, поэтому используется АЦП, встроенный в платформу TelosB. Сенсор работает от источника напряжения 9 В, поэтому требует дополнительного питания (так как TelosB обеспечивает только 6 В), которое обеспечивается отдельной девятивольтовой батареей. Все части системы, включая микрофоны, ориентированные по трем пространственным измерениям, упакованы в пластмассовый контейнер и проложены поролоном. Для работы с сенсором написан драйвер на языке NesC.

Плата с акселерометром MMA7260 подключена непосредственно к платформе TelosB. Подключение идет сразу по четырем каналам. Один канал — заземление, остальные три — для передачи данных (по одному на каждую пространственную ось X, Y, Z). Сенсор кодирует данные аналоговым сигналом, поэтому для оцифровки используется АЦП TelosB. Все части помещены в пластмассовый контейнер и проложены поролоном. Для работы с сенсором был также написан программный модуль на языке NesC.

Узлы беспроводной сети работают под управлением операционной системы TinyOS версии 2.1 [Hill et al., 2000]. На каждом узле установлено специально разработанное приложение, позволяющее обрабатывать потоки данных с локальных сенсоров в реальном времени, сохранять сырые данные или детектировать локальные события.

Рис. 3. Платформа TelosB, оснащенная инфразвуковым сенсором

Для синхронизации бортового времени на узлах в настоящее время используется протокол FTSP [Maroti et al., 2004], но в дальнейшем планируется подключить GPS-приемник.

Сообщения и данные с узлов поступают в параллельный кластер баз данных, организованный по оригинальной масштабируемой технологии ActiveStorage, созданной участниками проекта при поддержке предыдущих грантов РФФИ совместно с Исследовательским центром Майкрософт в Кембридже [Zhizhin et al., 2008].

1. ЗАДАЧА СБОРА И АНАЛИЗА ВЫСОКОЧАСТОТНЫХ НАБЛЮДЕНИЙ В РЕАЛЬНОМ ВРЕМЕНИ

Для решения задачи сбора и анализа высокочастотных наблюдений в реальном времени требуется постоянное прослушивание сенсорными узлами высокочастотных значений параметров. В ходе прослушивания сенсор должен уметь определить наступление некоторого заранее запрограммированного события и в режиме реального времени передавать на базовую станцию оповещение о его начале. Кроме сигнала оповещения, от сенсора требуется передать сами данные, идущие во время события. В зависимости от требований, данные могут либо передаваться в режиме реального времени с минимальной задержкой, либо сохраняться во flash-памяти сенсора и передаваться по возможности. Вне события сохранять данные не требуется.

Еще одним ключевым требованием является точное проставление времени получения данных. Это необходимо, чтобы иметь возможность детектирования пространственных событий и сопоставления событий, зафиксированных разными узлами. Для решения этой задачи в проекте был использован протокол FTSP [Maroti et al., 2004].

1.1. Архитектура и основные характеристики сети

В качестве сенсорных модулей подразумеваются модули фирмы Crossbow на базе платформы TelosB. Для работы сенсоров на языке NesC было написано промежуточное программное обеспечение, устанавливаемое непосредственно на модули.

Для всех модулей используется одинаковое программное обеспечение. Один из мотов подключается к машине (компьютеру, ноутбуку и т. п.) и исполняет роль базовой станции для сбора результатов измерений с остальных сенсоров. При передаче данных все узлы направляют результаты измерений на базовую станцию, которая через USB-порт передает их управляющей программе, работающей под управлением операционной системы.

Программное обеспечение позволяет собирать данные с помощью как встроенных сенсоров (термометр, гигрометр), так и внешних аналоговых (например, микрофона, магнитометра и др.), подключенных ко входу аналого-цифрового преобразователя беспроводного сенсорного модуля. Использование стандартных протоколов LE (Link Estimation) и CTP (Collection Tree Protocol), пакеты с реализацией которых включены в базовую комплектацию TinyOS, позволило

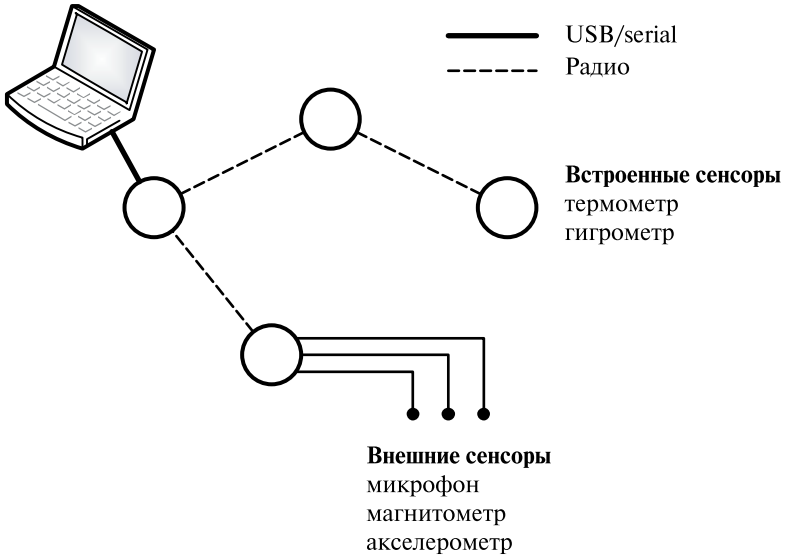


Рис. 4. Схема сети беспроводных сенсорных модулей

организовать передачу данных между сенсорными модулями по цепочке, образуя мультихоповый путь. В том случае, если какой-либо модуль находится вне зоны действия базовой станции, он может передать данные по цепочке через другие модули. Примерная схема работы сети показана на рис. 4.

1.2. Синхронизация по времени

Для проставления времени замера можно воспользоваться локальными часами, встроенными в каждый мот. Однако здесь возникают две проблемы:

- синхронизация мотов связана с расхождением локальных часов на разных часах, в результате это приводит к невозможности идентификации одного и того же события, зафиксированного разными узлами, что в свою очередь ведет к невозможности сбора направленных данных;
- синхронизация локального времени сенсора и времени UTC.

Для синхронизации мотов между собой в данной работе использован протокол FTSP (Flooding Time Synchronization Protocol) — распространенный протокол синхронизации времени для беспроводных сетей, обеспечивающий высокую точность синхронизации с использованием специальных меток на двух уровнях стека протоколов передачи сообщений по радио. Этот протокол также учитывает дрейф тактовых сигналов сенсорных модулей и вводит соответствующую поправку с использованием линейной регрессии.

Учитывая решение первой проблемы, для решения второй достаточно, чтобы только один сенсор синхронизировался со временем UTC (остальные сенсоры сети будут синхронизироваться по UTC на основании локальных часов сети). В настоящее время сопоставление времени UTC и локального времени

сенсорной сети проводится на стороне принимающей программы TinyOS при получении пакетов с данными на основании времени операционной системы. Этот метод имеет погрешность, и в дальнейшем будет осуществлена попытка использовать GPS-приемник.

1.3. Детектирование событий

В настоящее время получил широкое распространение так называемый STA/LTA (Short Time Averaging / Long Time Averaging) детектор. Принцип его действия (рис. 5) заключается в анализе отношения мощности сигнала, усредненного на коротком интервале времени (STA), к мощности, усредненной на продолжительном интервале (LTA). Решение о присутствии сигнала в шуме принимается, если статистика STA/LTA превышает установленный порог. Этот порог обычно находится в диапазоне значений 4–5 и зависит от свойств шумов в конкретной ситуации. Стоит отметить, что статистика отношения STA/LTA является оптимальной для обнаружения сигналов в гауссовом шуме.

После включения узел начинает собирать данные с сенсора и буферизовать их в кольцевом массиве, размер которого выбирается равным сумме размеров длинного и короткого окон. Процесс буферизации продолжается до тех пор, пока массив полностью не заполнится. После этого рассчитываются суммы значений для окон LTA и STA, после чего находится отношение STA/LTA. При следующей итерации, которая начинается при получении нового измерения с сенсора, сумма значений для LTA и STA не рассчитывается заново. Из суммы вычитается первое значение, окно сдвигается на один отсчет вперед, и к сумме прибавляется последнее значение в окне. Благодаря этому, снижается нагрузка на процессор узла и увеличивается скорость вычисления. Окна движутся по массиву по кольцевому принципу.

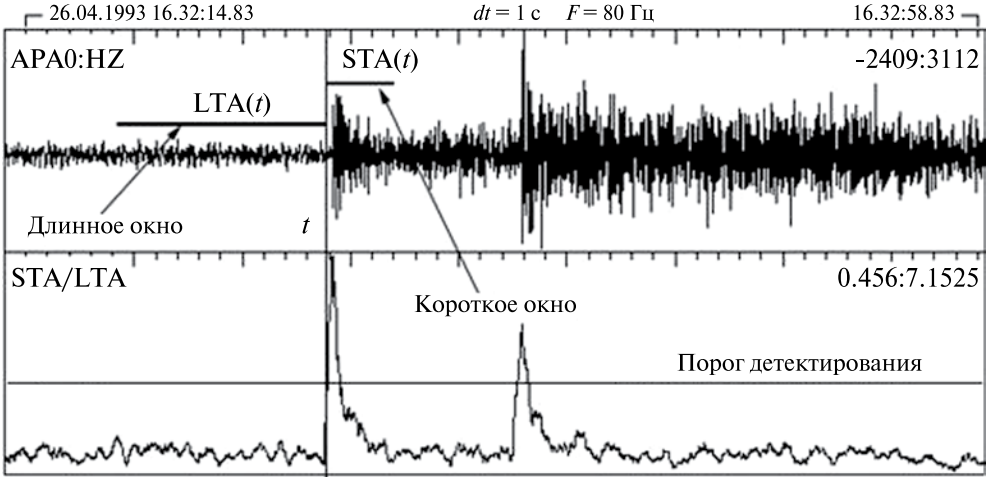


Рис. 5. Схема детектора STA/LTA

При первом превышении заданного порога отношения STA/LTA на главный сенсор отправляется информационное сообщение, содержащее номер сенсора, на котором зафиксировано возникновение события, и значение отношения STA/LTA. Данные при этом сохраняются в flash-памяти узла. Следующее информационное сообщение с узла будет отправлено, когда значение отношения STA/LTA станет ниже заданного порога.

1.4. Сохранение данных

Первая возможность сохранения данных — пересылка их в базу данных (рис. 6). Все собранные данные передаются по USB-порту на машину, где управляющая программа пересылает их в базу данных. В качестве базы данных используется хранилище ActiveStorage.

ActiveStorage — разработанное в Геофизическом центре РАН универсальное хранилище численных данных, предназначенное для хранения временных рядов, спутниковых изображений, результатов численного моделирования, а также любой другой информации, которая может быть представлена в виде многомерных численных массивов. Особенности хранилища таковы:

- универсальная архитектура, позволяющая держать разнородные данные в единой системе хранения;
- эффективное индексирование больших объемов данных (десятки и сотни терабайт);
- возможность базовой обработки данных непосредственно на узлах хранилища (арифметические операции, статистическая обработка, линейная свертка).
- интегрированные метаданные; описания данных не отделимо от самих данных;
- возможность автоматического распределения данных (а также распараллеливания обработки) по нескольким узлам;
- возможность использования в инфраструктуре грид через сервисы OGSA-DAI.

Другой возможностью сохранения данных является отправка их в систему Earthworm [Childs, Começ, 2003]. Earthworm — это распределенная система для обработки данных в области сейсмологии, обладающая следующими характеристиками:

1) модульность – каждая функция обработки инкапсулируется в отдельном модуле;

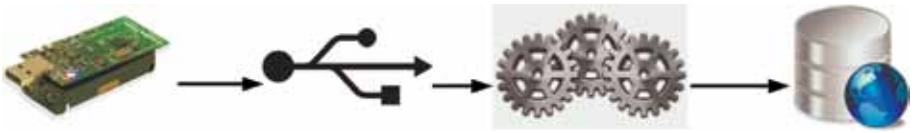


Рис. 6. Сценарий сохранения данных в активное хранилище ActiveStorage

2) независимость от системы выполнения; так как Earthworm является распределенной системой, при ее работе компьютеры с различными характеристиками объединяются для достижения общей цели; принцип независимости позволяет динамически менять конфигурацию Earthworm, перекладывая разную функциональность с одного участника на другого без привязки к характеристикам машин;

3) масштабируемость – система эффективно работает в сетях различного размера;

4) способность к взаимодействию – не является изолированной, позволяя другим автоматическим системам подключаться в режиме реального времени;

5) надежность – система является устойчивой к возникновению и изолированию ошибок.

Очень упрощенная конфигурация системы Earthworm приведена на рис. 7. Незаменимыми компонентами являются управляющий модуль startstop и буфер сообщений (message ring). Все остальные модули рабочие (каждый выполняет свою отдельную функцию) и могут быть подключены по желанию для проведения требуемого анализа. Все модули могут находиться на различных машинах. Сценарий взаимодействия следующий.

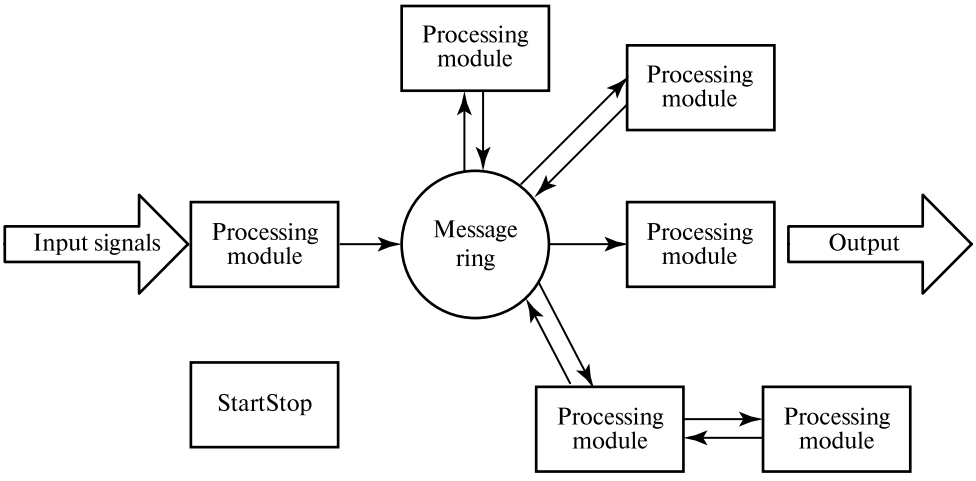


Рис. 7. Упрощенная конфигурация системы Earthworm

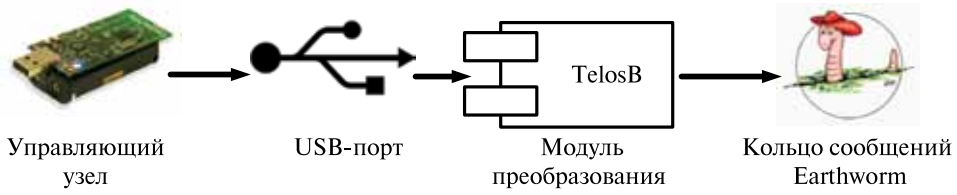


Рис. 8. Сценарий перенаправления данных в кольцо сообщений системы Earthworm

При передаче сообщения модуль-отправитель оформляет его в виде стандартного пакета и отправляет в кольцо сообщений. Все остальные модули, подключенные к кольцу, могут его «прослушивать» на предмет искомого пакета (например, от определенного автора или пакет определенного типа). Если «слушающий» модуль нашел требуемое сообщение, он копирует его и может обрабатывать. Если результат требуется передать далее по цепочке, то оформляется пакет, который отправляется в кольцо.

Для реализации взаимодействия сенсорной сети и системы Earthworm был написан модуль TelosB, относящийся к классу источников данных. Модуль принимает сообщения от управляющего узла, отправленные по USB, разархивирует их и отправляет в кольцо сообщений (рис. 8).

1.5. Визуализационный апплет

Для визуализации полученных данных было разработано простейшее Windows-приложение, позволяющее читать одномерные массивы (временные ряды) из ActiveStorage и отображать их в виде графиков. Приложение позволяет также следить за обновлением данных в хранилище в интерактивном режиме, считывая хвостовую часть массива заданной длины через определенные интервалы времени (например, каждую секунду). Рабочее окно приложения для визуализации полученных данных представлено на рис. 9.

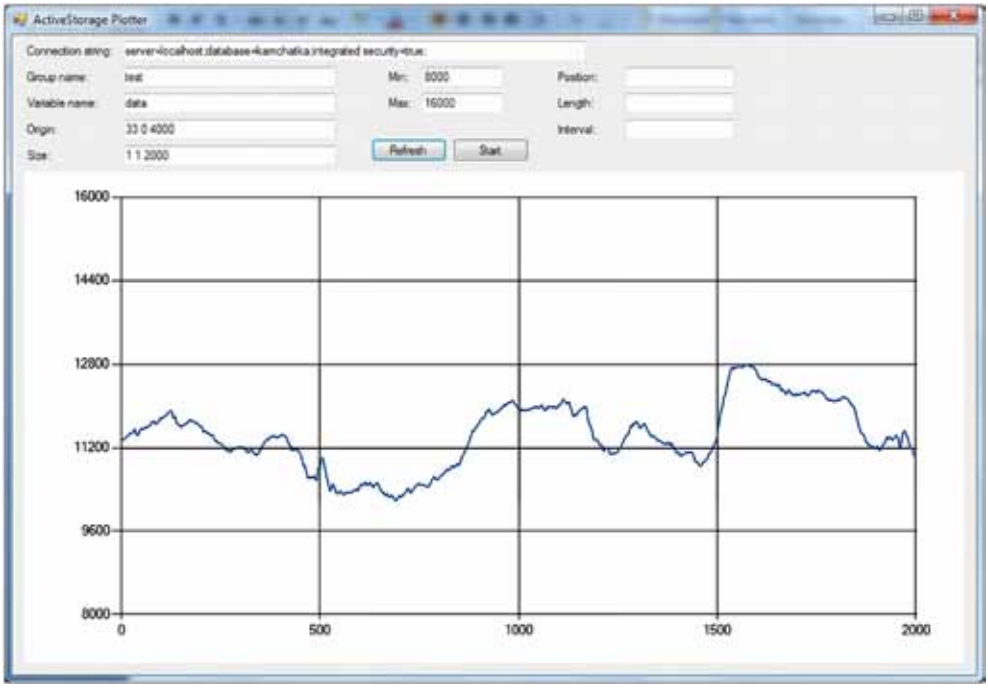


Рис. 9. Рабочее окно приложения для визуализации полученных данных

1.6. Тестирование

Было проведено несколько экспериментов с небольшими сетями, состоящими из 3–5 беспроводных сенсорных модулей, расположенных друг от друга на расстоянии порядка 10 м, в прямой видимости. Данные собирались с помощью встроенных и внешних аналоговых сенсоров (микрофон, магнитометр, акселерометр), подключенных ко входу АЦП беспроводного сенсорного модуля.

Тестирование показало, что наиболее узким местом является USB-соединение между управляющим узлом и управляющей программой. Надежная передача данных поддерживалась при частоте передачи порядка 270 пакетов в секунду. В зависимости от требуемой скорости реакции системы данные можно буферизовать, накапливая в одном пакете данные нескольких измерений.

2. ЗАДАЧА СБОРА РЕЗУЛЬТАТОВ НИЗКОЧАСТОТНЫХ НАБЛЮДЕНИЙ С БУФЕРИЗАЦИЕЙ ДАННЫХ НА УЗЛАХ

В данной задаче требуется собирать данные с большими интервалами между замерами (от десятков секунд до часов), буферизовать их в памяти узла и передавать на управляющий узел по запросу. Такой вариант актуален для долгосрочных наблюдений окружающей среды. Сценарий работы следующий.

1. Узлы с сенсорами устанавливаются на территории эксперимента и начинают брать пробы с последующим их сохранением во flash-памяти. В то же время узлы могут обмениваться техническими данными между собой (например, для синхронизации часов).

2. Один из узлов сети (управляющий) подключен к вычислительной машине по USB-порту. Его основной задачей является обеспечение процесса скачивания данных с локальных узлов. Управляющий узел может не находиться в сети все время, он даже может быть выключен большую часть времени.

3. По требованию оператора или таймера управляющий узел инициирует процесс скачивания данных. Для этого он рассылает каждому узлу в сети команду передачи данных и управляющую информацию (например, список узлов, через которые следует передавать данные), после чего открывает виртуальный канал, принимает данные и передает их по USB-порту на машину, к которой подключен.

Основной проблемой, возникающей при решении данной задачи, является увеличение работоспособности мотов без подзарядки батареек. Больше всего энергии затрачивается на радиообмен, поэтому требуется его максимальное сокращение. Возможны следующие варианты:

1) выключать радиоаппаратуру, если не идет передача данных;
2) сократить время передачи данных.

Первый метод в чистом виде нереализуем, так как, если узел будет все время выключен из радиообмена, он не сможет получить команду передачи данных. Поэтому он должен периодически прослушивать радиотракт с целью обнаружения управляющих сообщений. Поскольку получения данных в реальном времени

не требуется, интервалы прослушивания сети могут быть достаточно большими. Кроме того, требуется легкий протокол «прослушивания и оповещения», чтобы максимально сократить время и объем передачи технических данных.

Для реализации второго метода можно:

1) сжимать данные перед передачей;
2) выбирать наиболее оптимальный маршрут передачи данных;
3) оптимизировать передачу данных по установленному маршруту.

Для достижения требуемых результатов был адаптирован проект Koala [Musaloiu-Elefteri et al., 2008].

2.1. Архитектура и основные характеристики сети. Проект Koala

Проект Koala — это система надежного сбора данных для долгосрочных наблюдений окружающей среды с облегченным режимом работы. Система разрабатывается в Университете Джона Хопкинса (США).

Система Koala достигает облегченного режима работы отключением на узлах радиопередающей аппаратуры в течение большей части времени. Это достигается, во-первых, эффективным способом включения узлов в радиообмен в любое время, когда бы управляющий узел ни пожелал скачать данные (для этого используется техника LPP). Во-вторых, в отличие от других систем, которые расходуют энергию на поддержание сетевых таблиц (маршрутизации, расписания сна и т. п.), Koala не поддерживает таких состояний. Вместо этого управляющий узел вычисляет сетевые пути непосредственно перед скачиванием данных на основании информации о доступности узлов. Затем протокол FCP (Flexible Control Protocol), специально разработанный создателями системы Koala, доставляет информацию о сетевых путях на узлы, которые используют ее для передачи данных.

Техника LPP (Low Power Probing) позволяет быстро «разбудить» узлы в сети. Узлы, находящиеся в режиме с отключенной передающей аппаратурой, периодически «просыпаются» и рассылают короткий пакет, требующий подтверждения. Если такое подтверждение приходит, узел включается в обмен сообщениями и начинает «будить» соседей. Таким образом, когда управляющему узлу требуется скачать данные, он начинает «прослушивать» радиоканал в ожидании LPP-пробы. После получения такого пакета узел отсылает отправителю подтверждение. По заявлению авторов, данная технология может «разбудить» сеть из 24 мотов менее чем за 30 с.

Протокол FCP используется для фиксации в сети информации о текущих маршрутах передачи данных и распространения этой информации на конечные узлы. Сценарий работы представлен на рис. 10 (см. с. 175). Управляющий узел перед скачиванием данных рассылает команду конечным узлам собирать информацию о своих соседях: какой узел в зоне видимости, какая задержка при обмене сообщениями (ping) и т. п. Управляющий узел собирает эту информацию и вычисляет стоимость передачи до каждого узла на основании минимизации общей задержки. Может быть установлено несколько альтернативных маршрутов к одному узлу для балансировки трафика. В этот момент информация о маршрутизации

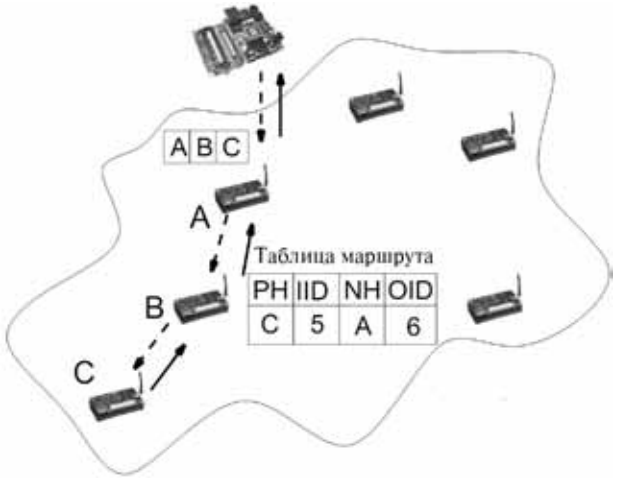


Рис. 10. Сценарий работы системы Koala при сборе данных на управляющий узел

фиксируется (стадия замораживания). При помощи протокола FCP каждому узлу доставляется информация о проходящем через него маршруте. Информация включает идентификаторы: предыдущего узла (PH), входящих пакетов (IID), следующего узла (NH), исходящих пакетов (OID). Полученные установки настраивают узел на прием пакетов с данными, помеченных идентификатором OID, от узла с идентификатором PH и перенаправление этих пакетов на узел с идентификатором NH, проставляя для них идентификатор IID.

Протокол FCP поддерживает два вида передачи данных: по выделенному каналу и пакетную. В первом случае перед передачей данных устанавливается маршрут на узлах-посредниках, после чего начинается обмен между узлами-агентами. Во втором случае данные передаются без предварительного уведомления: промежуточные узлы пересылают пакет в соответствии с ранее построенными маршрутными таблицами. Оба сервиса обеспечивают надежную доставку данных. Первый вариант удобен при передаче серии пакетов, второй — при передаче коротких сообщений.

Еще одна оптимизация системы Koala для экономии заряда батареек — переключение радиоканала для узлов, включенных в маршрут передачи. Это позволяет незадействованным мотам не принимать участие в обмене управляющими LPP-пакетами и, таким образом, «заснуть».

2.2. Макет беспроводной сенсорной сети на полигоне вычислительного центра ИКИ

Эксперимент проводился в вычислительном центре Института космических исследований РАН. Четыре мота были установлены на территории центра (рис. 11, см. с. 176). На рис. 12 (см. с. 176) представлен внешний вид оболочки, в которую были помещены устройства. В коробке были проделаны отверстия для сенсоров освещенности.

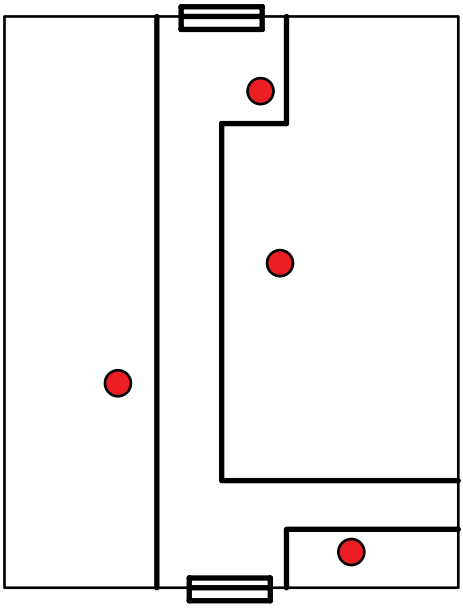


Рис. 11. Схема вычислительного центра ИКИ РАН. Красными кружками отмечено расположение узлов сети

Моты снабжены стандартным набором сенсоров и программным обеспечением Koala. Сенсоры сконфигурированы на получение замеров с частотой один замер в минуту. Управляющий узел подключен к нетбуку, на котором установлена операционная система (ОС) Windows и виртуальная машина с ОС TInyOS 2.1.

Работа проходила по следующему сценарию (рис. 13, см. с. 177).

1. Данные скачивались нерегулярно, с интервалами от 2 до 7 дней. Точка размещения управляющего узла при скачивании данных варьировалась от непосредственной близости к одному из мотов до 10...15 м к ближайшему моту с препятствиями в виде стен, мебели, радиопомех.

2. Результаты сохранялись в сыром специфическом виде в локальном файле в виртуальной машине.

3. Сырые данные нормализовывались в стандартные единицы измерения и переформатировались в числовой формат.

4. Данные в числовом формате загружались в удаленную базу данных MySQL, откуда они могут быть запрошены системой визуализации. Кроме того данные из числового формата могут быть преобразованы в CSV формат и отображены, например, в приложении MS Excel.

Рис. 12. Внешний вид защитной оболочки мота с проделанными отверстиями для сенсоров измерения освещенности

Эксперимент показал следующие результаты:

- начало эксперимента: 03.11.2010 г.;
- окончание эксперимента: 07.12.2010 г.;
- падение заряда батареек: в среднем упал с 3,2 до 3,0 В (здесь стоит отметить агрессивную среду, периодическое прерывание каналов и неоптимальный аппарат системы скачива-

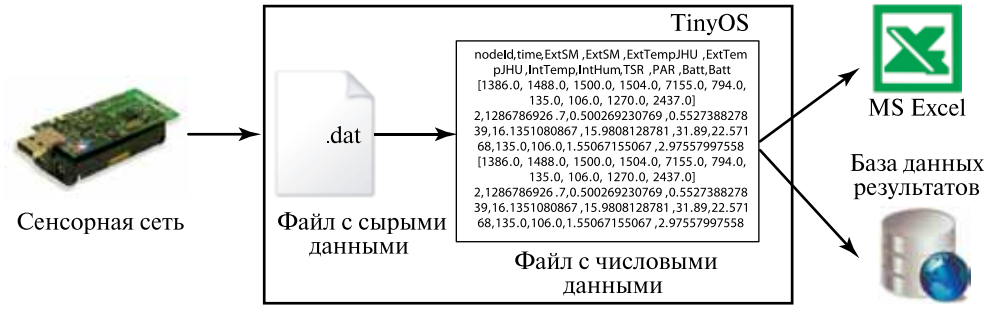


Рис. 13. Сценарий работы с полученными данными

ния данных Koala, требующий больших накладных расходов при потере связи, что будет учтено в дальнейшей работе);

• скорость опроса сети: зависит от удаления сенсора и числа промежуточных звеньев в маршруте загрузки данных, но в среднем время, затраченное на скачивание данных с одного мота, полученных за сутки (6 сенсоров, замер один раз в минуту), равнялось одной минуте;

• скорость LPP протокола: по умолчанию, на то, чтобы «разбудить» все моты, управляющему узлу давалось полторы минуты, но реально приведение сети в готовность происходила за 10...20 с.

Эксперимент показал положительные результаты с точки зрения экономии батарей и надежности сохранения данных (нет ни одного потерянного замера). Положительным моментом также можно считать поддержку связи на расстоянии порядка 10...20 м через препятствия (стены) и в условии постоянных помех от аппаратуры. За более чем месяц эксперимента не потребовалось техническое обслуживание ни одному моту.

К негативным сторонам следует отнести высокие накладные расходы, связанные с разрывом виртуального канала передачи данных, а также довольно низкую скорость передачи данных.

3. СИСТЕМА ВИЗУАЛИЗАЦИИ ДАННЫХ ДЛЯ ВЕБ-ПРИЛОЖЕНИЙ

Для визуализации временных рядов, полученных с сенсоров, была адаптирована система Auroral Resource Toolkit, разрабатываемая в NASA. Система написана на JavaScript с использованием двух пакетов разработки: Qooxdoo [Surhone et al., 2010] и Dygraphs*.

Пакет Qooxdoo — это инструмент разработки веб-интерфейсов. В основе лежат классы, написанные на JavaScript и расширяющие его функциональность. Приложения, созданные с использованием Qooxdoo, поддерживаются современными веб-браузерами (Internet Explorer, Opera, Safari). Пакет Qooxdoo включает

* Программная библиотека на языке JavaScript Dygraphs. http://dygraphs.com/.

платформенно-независимые инструменты разработки приложений, внедренный пакет разработки пользовательских интерфейсов, а также пакеты клиент-серверной архитектуры; распространяется под LGPL/EPL-лицензиями.

Пакет Dygraphs — это свободно распространяемая JavaScript библиотека для создания интерактивных графиков временных рядов с возможностью зумирования. Библиотека предназначена для визуализации плотных наборов данных и снабжает пользователя возможностью исследовать и интерпретировать их.

Общая схема работы представлена на рис. 14. При обращении пользователя к визуализационному окну пакета Qooxdoo отправляется запрос к сервлету получения данных. Сервлету передаются параметры выборки: идентификатор эксперимента, номер сенсора, время начала и конца выборки, название параметра. Сервлет запрашивает базу данных и формирует файл в формате CSV, который

Рис. 14. Общая схема работы системы визуализации временных рядов

```
Date,Temperature ,PAR
Wed Nov  24 00 :00:50 2010 ,23.12,2.0
Wed Nov  24 00 :01:50 2010 ,23.1,2.0
Wed Nov  24 00 :02:50 2010 ,23.08,2.0
Wed Nov  24 00 :03:50 2010 ,23.08,2.0
Wed Nov  24 00 :04:50 2010 ,23.11,2.0
Wed Nov  24 00 :05:50 2010 ,23.18,2.0
Wed Nov  24 00 :06:50 2010 ,23.2,2.0
Wed Nov  24 00 :07:50 2010 ,23.22,2.0
Wed Nov  24 00 :08:50 2010 ,23.2,2.0
Wed Nov  24 00 :09:50 2010 ,23.14,2.0
Wed Nov  24 00 :10:50 2010 ,23.08,2.0
Wed Nov  24 00 :11:50 2010 ,23.06,2.0
Wed Nov  24 00 :12:50 2010 ,23.08,2.0
Wed Nov  24 00 :13:50 2010 ,23.08,2.0
Wed Nov  24 00 :14:50 2010 ,23.08,2.0
Wed Nov  24 00 :15:50 2010 ,23.09,2.0
```

Рис. 15. Данные в CSV формате для пакета Dygraphs

понимает Dygraphs (рис. 15, см. с. 178). Сформированный файл отправляется обратно в JavaScript, где и отображается (рис. 16).

Система визуализации состоит из боковой панели выбора и главного окна. Боковая панель разделена на меню выбора данных и компоненту выбора времени. В компоненте выбора времени отдельно выбираются даты начала обозреваемого интервала и его конца. Кроме того, на ползунке, который располагается между ними, можно выбрать дату и время, которые будут отображаться на графике временного ряда красной линией.

Меню выбора данных поддерживает иерархическое именование. В настоящее время иерархия выглядит следующим образом: SWN (sensor wireless network) → название эксперимента (например, «ИКИ РАН») → название параметра (например, «Температура», «Влажность» и т. д.) → номер сенсора.

При перетаскивании иконок атомарных пунктов (т. е. тех пунктов, которые не содержат подпунктов) из левого меню на рабочий стол вызывается окно, в котором при помощи пакета Dygraphs визуализируется график временного ряда. Границами этого ряда являются либо выбранные даты, либо максимально- (минимально-) возможные даты, для которых в базе данных есть измерения. На главное окно можно вытянуть несколько наборов данных. В каждом отдельном окне можно проводить зумирование мышью. Для всех окон можно менять в компоненте выбора времени даты начала и конца, а также подсвечиваемое время.

В настоящее время система визуализации опубликована в Интернете по адресу http://aeolus.wdcb.ru:8080/sensors/.

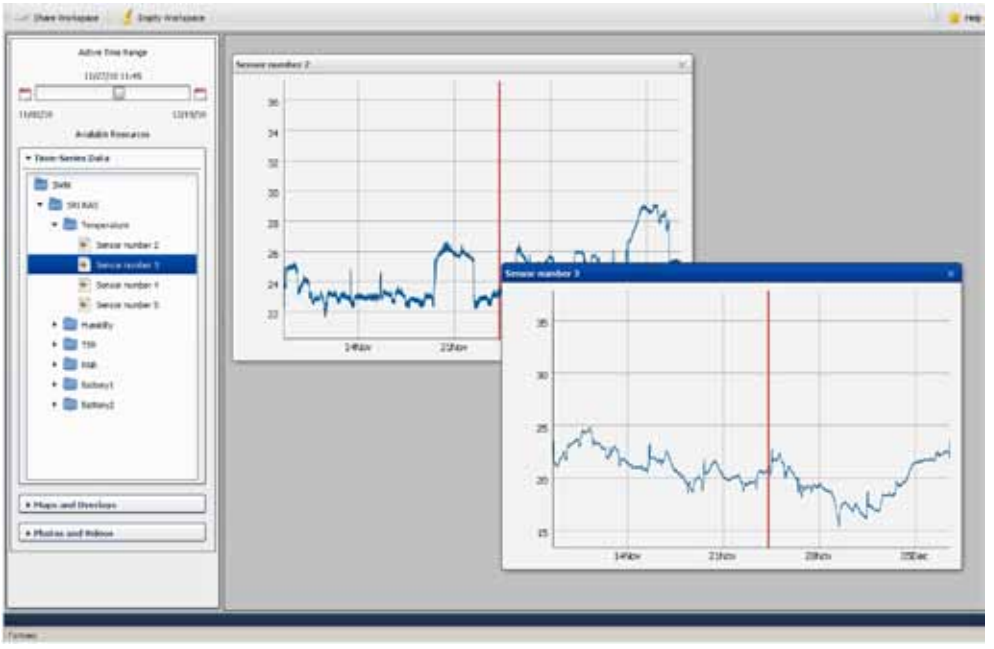


Рис. 16. Визуализация временных рядов данных, полученных с сенсоров

ЗАКЛЮЧЕНИЕ

В ходе работы получены следующие результаты.

1. Создан действующий макет двумерной сенсорной сети.
2. Базовый набор сенсоров расширен дополнительными сенсорами.
3. Разработаны элементы алгоритмов и промежуточного программного обеспечения, реализующие временной детектор событий, а также буферизацию, маршрутизацию и синхронизацию узлов в беспроводных сенсорных сетях.
4. Разработаны сервисы сбора и обработки данных с сенсорных сетей и загрузки данных в активное хранилище и проект Earthworm.
5. Разработаны веб-сервисы интерактивной визуализации и веб-приложение исторической визуализации временных рядов данных сенсорных сетей.
6. Проведены полевые испытания системы сбора данных.

Для дальнейших исследований актуальными выглядят следующие направления:

- добавление программных компонент, анализирующих пространственно-временную связь событий, происходящих на сенсорах; в реализованных на данном этапе макетах взаимосвязь событий не учитывается;
- усовершенствование метода сопоставления времени UTC и локального времени сенсоров;
- расширение созданных алгоритмов и промежуточного программного обеспечения, реализующего временной детектор событий в беспроводных сетях возможностью распределенного детектирования пространственно-временных событий, добавление использования нечеткой логики;
- привязка созданных сервисов визуализации к электронным картам.

Приложение

Спецификация узла беспроводной сенсорной сети TelosB

Процессор	16-бит RISC
Flash-память	48 КБ
Память для регистрации измерений	1024 КБ
Оперативная память	10 КБ
EEPROM для хранения конфигурации	16 КБ
Последовательный порт	UART
АЦП	12-бит АЦП
ЦАП	12-бит ЦАП
Другие интерфейсы	цифровой I/O, I2C,SPI
Потребляемый модулем (без радиоаппаратуры и сенсоров) ток в активном режиме работы	1,8 мА
Потребляемый модулем (без радиоаппаратуры и сенсоров) ток в режиме «сна»	5,1 мкА
Частота радиоканала	2400...2483,5 МГц
Скорость радиоканала	250 кбит/с
Мощность ВЧ-сигнала	–24...0 дБм

Чувствительность	–90 дБм (мин), –94 дБм (тип.)
Подавление помех соседнего канала сверху (+5 МГц)	47 дБ
Подавление помех соседнего канала снизу (–5 МГц)	38 дБ
Расстояние передачи на открытой местности	75...100 м
Расстояние передачи в помещении	20...30 м
Ток, потребляемый радиоаппаратурой в активном режиме передачи	23 мА
Ток, потребляемый радиоаппаратурой в режиме пониженного энергопотребления	21 мкА
Ток, потребляемый радиоаппаратурой в режиме «сна»	1 мА
Диапазон сенсора видимого спектра (Hamamatsu S1087)	320...730 нм
Диапазон инфракрасного спектра (Hamamatsu S1087-01)	320...1100 нм
Диапазон сенсора относительной влажности (Sensirion SHT11)	0...100 %
Цена деления сенсора относительной влажности	0,03 %
Точность сенсора относительной влажности	±3,5 %
Диапазон температурного сенсора (Sensirion SHT11)	от –40 до 123,8 °C
Цена деления температурного сенсора	0,01 °C
Точность температурного сенсора	±0,5 °C
Источник питания	2X AA
Пользовательский интерфейс	USB
Размер устройства	2,55×1,24×0,24 мм
Масса устройства	23 г

ЛИТЕРАТУРА

[Childs, Komeç, 2003] *Childs D.*, *Komeç A.* The Kandilli Observatory Real-Time Automated Seismic Data Processing System // ORFEUS Newsletter. March, 2003. [Электрон. текст]. Режим доступа: http://www.orfeus-eu.org/newsletter/vol5no1/vol5no1.pdf.

[Gay et al., 2003] *Gay D.*, *Levis Ph.*, *von Behren R.*, *Welsh M.*, *Brewer E.*, *Culler D.* The nesC Language: A Holistic Approach to Networked Embedded Systems // Proc. Programming Language Design and Implementation (PLDI) 2003. June 2003.

[Hill et al., 2000] *Hill J.*, *Szewczyk R.*, *Woo A.*, *Hollar S.*, *Culler D.*, *Pister K.* System Architecture Directions for Networked Sensors // Proc. 9th Intern. Conf. Architectural Support for Programming Languages and Operating Systems (ASPLOS). Nov. 2000.

[Levis, Gay? 2009] *Levis Ph.*, *Gay D.* TinyOS Programming. Cambridge University Press, Apr. 2009. ISBN: 0521896061, ISBN-13: 9780521896061.

[Maroti et al., 2004] *Maroti M.*, *Kusy B.*, *Simon G.*, *Ledeczi A.* The flooding time synchronization protocol // Proc. 2nd ACM Conf. Embedded Networked Sensor Systems (SenSys). Nov. 2004.

[Musaloiu-Elefteri et al., 2008] *Musaloiu- Elefteri R.*, *Liang C.-J.*, *Terzis A.* Koala: Ultra-Low Power Data Retrieval in Wireless Sensor Networks // Proc. IPSN 2008. P. 421–432.

[Shahin Farahani, 2008] *Shahin Farahani.* ZigBee Wireless Networks and Transceivers. Newnes, 2008. ISBN: 978-0-7506-8393-7.

[Surhone et al., 2010] *Surhone L. M.*, *Tennoe M. T.*, *Henssonow S. F.* Qooxdoo, Betascript publishing, 2010, ISBN-10: 6131904790, ISBN-13: 9786131904790.

[Zhizhin et al., 2008] *Zhizhin M.*, *Poyda A.*, *Mishin D.*, *Medvedev D.*, *Kihn E.*, *Lyutsarev V.* Grid Data Mining with Environmental Scenario Search Engine (ESSE) // Data Mining Techniques in Grid Computing Environments / Ed. W. Dubitsky. Wiley. 2008. Ch. 13. P. 281–306.

DISTRIBUTED EVENT DETECTION IN MULTIDIMENTIONAL DATA STREAMS IN WIRELESS SENSOR NETWORKS

A.A. Poyda [1,2], M.N. Zhizhin [1,2], D.P. Medvedev [1,2], A.E. Moskvitin [3], A.V. Andreev [1,2]

[1] I*nstitutuion of the Russian Academy of Sciences Geophysical center of RAS* (*GC RAS*), *Moscow*
[2] *Space Research Institute* (*IKI RAN*), *Moscow*
[3] *Institute of Cosmophysical Researchers and Radio Wave Propagation Far Eastern Branch of the Russian Academy of Sciences* (*IKIR FEB RAS*), *Kamchatka region*, *Paratunka*

In this paper we present software development and hardware implementation for environmental, seismic and geoacoustic data collection and events detection in data streams in a distributed wireless sensor network.

Keywords: wireless sensor networks, environmental monitoring, event detector, stream data processing, geoacoustic, seismology.

Poyda Alexey Anatolyevich — senior scientist, candidate of physical and mathematical sciences, e-mail: poyda@wdcb.ru.
Zhizhin Mikhail Nikolaevich — head of the laboratory, candidate of physical and mathematical sciences, e-mail: jjn@wdcb.ru.
Medvedev Dmitry Petrovich — scientist, e-mail: dmedv@wdcb.ru.
Moskvitin Andrey Evgenevich — junior scientist, e-mail: moskvitin.andrey@ikir.ru.
Andreev Alexandr Viktorovich — engineer, e-mail: and@wdcb.ru.

УДК 004.75

ТЕХНОЛОГИЯ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛЕНИЙ X-COM: ВОЗМОЖНОСТИ, ЗАДАЧИ, НАПРАВЛЕНИЯ РАЗВИТИЯ

Вл. В. Воеводин, С. И. Соболев

Научно-исследовательский вычислительный центр МГУ имени М. В. Ломоносова (*НИВЦ МГУ*)

Распределенные вычисления (метакомпьютинг) — перспективная технология, применяющаяся для объединения множества компьютерных систем с целью решения задач, требующих значительного объема вычислений. В НИВЦ МГУ разрабатывается система метакомпьютинга X-Com — программная платформа для организации и проведения распределенных расчетов. В статье описываются основные возможности системы X-Com, решенные с ее помощью реальные задачи и планы развития системы.

Работа выполнена при содействии гранта Президента Российской Федерации для государственной поддержки молодых российских ученых – кандидатов наук МК-5104.2011.9.

Ключевые слова: большие задачи, распределенные вычисления, распределенные среды, неоднородные среды, метакомпьютинг, X-Com.

ВВЕДЕНИЕ

Технологии распределенных вычислений, активно развивающиеся с середины 90-х гг. прошлого века, подразумевают применение доступных разнородных компьютерных ресурсов, соединенных между собой сетями общего назначения, для решения вычислительно сложных задач. В отличие от традиционного решения таких задач на специализированных высокопроизводительных комплексах, в распределенных вычислениях используются имеющиеся компьютерные ресурсы, от офисных машин до суперкомпьютеров, связанных между собой обычными, неспециализированными каналами связи (например, сетью Интернет). Для объединения компьютеров применяется специальное программное обеспечение, формирующее на основе разнородных компьютерных ресурсов единую распределенную вычислительную среду [Воеводин, Воеводин, 2002].

Можно выделить несколько типичных ситуаций, в которых применение распределенных вычислений целесообразно и оправданно. Например, доступных суперкомпьютерных ресурсов не хватает для решения определенной задачи за необходимое время или с заданной точностью. Объединение несколько суперкомпьютеров в распределенную среду для работы над этой задачей может дать искомый результат. Другой случай — доступа на суперкомпьютеры нет вообще, но под рукой имеются домашние и офисные компьютеры, учебные классы, серверы. Эти ресурсы также могут быть объединены для работы над единой задачей.

Воеводин Владимир Валентинович — заместитель директора, доктор физико-математических наук, член-корреспондент Российской академии наук, e-mail: voevodin@parallel.ru.

Соболев Сергей Игоревич — научный сотрудник, кандидат физико-математических наук, e-mail: sergeys@parallel.ru.

Интересно, что программную платформу для организации распределенных вычислений можно рассматривать как технологию программирования и изначально распараллеливать имеющуюся программу с ее помощью. В ряде случае это оказывается проще, чем переписывать программу с применением технологии MPI и подобных. Если же исходные тексты программы не доступны, то распределенные вычисления — фактически единственный вариант ее запуска в параллельном режиме.

Распределенные среды, полученные в результате объединения компьютерных ресурсов различного плана, обладают целым комплексом свойств, не характерных для традиционных вычислительных систем. Масштабность — потенциально могут быть объединены сотни тысяч компьютеров, доступных через Интернет. Географическая распределенность — компоненты среды могут быть разбросаны по всему миру. Неоднородность — компьютеры, составляющие среду, могут отличаться друг от друга по всем характеристикам: аппаратные платформы, операционные системы, состав установленного программного обеспечения. Динамичность — компьютеры могут подключаться к расчету и отключаться от него в любой момент времени, при этом никакие данные расчета не должны потеряться, а вновь появляющиеся ресурсы должны быть задействованы. Различные политики администрирования — каждый компьютер принадлежит своему владельцу, который регламентирует использование его ресурсов.

Какие задачи могут эффективно решаться в подобной сложной среде? Масштабность среды требует от задачи массивных ресурсов внутреннего параллелизма. Распределенность и динамичность приводят к необходимости минимизации коммуникаций между параллельными процессами и выбору как можно более простой схемы коммуникаций. Класс таких задач достаточно велик — это задачи поиска и оптимизации, матричные и комбинаторные задачи, которые встречаются в самых разных областях. Для организации вычислительного процесса обычно применяется клиент-серверная схема, когда сервер берет на себя разбиение большой задачи на множество независимых подзадач и объединение результатов их работы, а клиенты получают от сервера входные данные, выполняют вычисления и отсылают серверу результат. Именно такая идея заложена в основу практически всех программных платформ для организации распределенных вычислений [Воеводин, 2007].

1. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ДЛЯ ОРГАНИЗАЦИИ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛЕНИЙ

Приведем несколько примеров программных платформ для организации распределенных вычислительных сред. Они схожи архитектурно и выполняют похожие задачи, хотя создавались с различными целями.

Система Condor (1988 г., University of Wisconsin-Madison) — один из первых проектов по созданию платформы распределенных вычислений. Основная цель — эффективное использование ресурсов простаивающих компьютеров в организациях. Известно, что такие компьютеры достаточно мало нагружены даже в рабочее время и полностью простаивают по ночам и в выходные дни. Condor

распределяет потоки относительно небольших независимых задач на доступные ему компьютеры.

Платформа BOINC (2002 г., UC Berkeley) предназначена для организации проектов «добровольных вычислений» в Интернете. Любой человек может поддержать исследовательский проект, подключив свой компьютер к такому расчету. Платформа выросла из известного проекта SETI@home по выявлению закономерностей в сигналах, получаемых от радиотелескопа. В настоящее время более сотни проектов по распределенному решению задач из самых разных сфер науки и технологий базируются на платформе BOINC.

Проект Hadoop (2005 г., Apache Group), поддерживаемый такими крупными компаниями как Yahoo! и IBM, реализует обработку очень больших массивов данных методом MapReduce. В MapReduce выделяются функции map, которые разбивают входные данные на пару «ключ – значение», и reduce, которые группируют полученные пары по ключам. Вычисления ведутся в распределенном режиме, данные доступны через распределенную файловую систему, также реализованную в данном проекте.

Система метакомпьютинга X-Com (2001 г., НИВЦ МГУ имени М.В. Ломоносова) предназначена для быстрого развертывания и проведения распределенных вычислительных экспериментов. Система представляет собой инструментарий для адаптации и поддержки выполнения программ в распределенных неоднородных средах. Разработка системы велась с учетом всех описанных свойств распределенных сред: она поддерживает среды с десятками тысяч вычислительных узлов (процессоров), обеспечивает корректную работу в условиях высокой динамичности состава среды, не требует административного доступа к ресурсам. Система X-Com написана на языке Perl, что обеспечивает ее работоспособность на подавляющем большинстве современных программно-аппаратных платформ [Воеводин и др., 2009].

2. ПРИНЦИПЫ РАБОТЫ СИСТЕМЫ МЕТАКОМПЬЮТИНГА X-COM

Рассмотрим подробнее адаптацию и запуск прикладных задач в распределенной среде X-Com. В основе архитектуры X-Com лежит клиент-серверная схема. Соответственно ей, прикладная задача должна быть логически разделена на две части: серверную и клиентскую. Как уже говорилось, серверная часть отвечает за разбиение задачи на множество независимых порций и объединение результатов. Для реализации серверной части задачи в системе X-Com имеются два прикладных интерфейса (API). В простейшем случае, если необходимо выполнить одну и ту же программу на множестве различных входных файлов, используется API Files, и в этом случае серверная часть задачи описывается в параметрах настройки сервера X-Com (указываются пути к входным и выходным каталогам и список файлов для обработки). В менее тривиальных случаях применяется API Perl. Этот интерфейс предполагает написание модуля на языке Perl, реализующего заданный набор функций: инициализация серверной части задачи, генерация номера первой и последней порции, генерация тела порции по ее номеру, обработка

результатов готовой порции, условия завершения работы серверной части задачи, а также действия, выполняемые перед завершением.

Аналогичный подход применяется и для реализации клиентской части задачи. В элементарном случае в параметрах настройки сервера X-Com указывается формат команды с именами входных и выходных файлов, которая будет запущена клиентом X-Com на вычислительном узле. В более сложном случае клиентская часть задачи может быть описана двумя функциями на языке Perl: инициализация задачи на узле и обработка каждой порции данных.

После реализации клиентской и серверной части прикладной задачи формируется непосредственно вычислительная среда. Этот процесс состоит из двух частей: запуск клиентов X-Com на вычислительных узлах и настройка и запуск сервера X-Com. Клиент X-Com может быть запущен на вычислительном узле постоянно (в монопольном режиме) или запускаться в те моменты времени, когда узел не занят другими процессами (работа по занятости). На высокопроизводительных вычислительных комплексах используются интерфейсы к штатным системам очередей. В текущей версии X-Com поддерживается взаимодействие с Cleo, Torque, LoadLeveler, а также Unicore [Соболев, 2010*а*, *б*].

Сервер X-Com запускается на специальной выделенной машине вручную либо с помощью подсистемы управления заданиями. Последний вариант предоставляет пользователям более удобный способ работы с распределенной средой, позволяя им оперировать привычными понятиями очереди заданий. При этом обеспечивается как последовательное выполнение задач на всех доступных ресурсах, так и параллельное выполнение одновременно нескольких заданий, а также запуск заданий с учетом их требований к ресурсам, на которые они будут распределены.

Существенная особенность системы X-Com — возможность построения иерархических распределенных сред с произвольным числом уровней. Данная функциональность реализована с помощью промежуточных серверов X-Com. Промежуточный сервер получает задания и необходимые данные от вышестоящего сервера X-Com (для этого сервера он представляется клиентом X-Com) и распределяет их внутри пула своих клиентов (для них он представляется единственным сервером X-Com). Введение промежуточных серверов позволяет снизить нагрузку на центральный сервер распределенной среды, оптимизировать потоки данных, а также подключить к расчетам вычислительные ресурсы, находящиеся внутри закрытых сетей [Соболев, 2010*а*].

3. СИСТЕМА X-COM В РЕАЛЬНОЙ ВЫЧИСЛИТЕЛЬНОЙ ПРАКТИКЕ

С помощью системы X-Com был решен целый ряд вычислительно емких задач из различных научных областей, при этом в каждом расчете отрабатывалась та или иная функциональность системы. В ходе одного из самых первых вычислительных экспериментов в 2002 г. совместно с центром «Биоинженерия» РАН решалась задача по определению скрытых периодичностей в генетических последовательностях. Особенностью расчета стал солидный географический

размах: в нем участвовало свыше 400 компьютеров из десяти различных организаций, в числе которых были Уфимский государственный авиационный технический университет (УГАТУ), Уральское отделение Российской академии наук (УрО РАН), Екатеринбург; Научно-исследовательский вычислительный центр МГУ (НИВЦ МГУ), Москва; Объединенный институт ядерных исследований (ОИЯИ), Дубна; Институт программных систем Российской академии наук (ИПС РАН), Переславль-Залесский и другие. Общее время эксперимента составило около 64 часов. Один средний компьютер справился бы с подобной задачей лишь за два года непрерывной работы. Взаимодействие между компьютерами осуществлялось через Интернет, суммарно было передано 9 ГБ данных [Воеводин, Филамофитский, 2003].

Совместно с Пензенским государственным университетом была решена серия задач из области электромагнитной динамики (дифракция электромагнитной волны на телах различной конфигурации). Каждый из этапов решения выполнялся в распределенных средах, построенных на основе нескольких суперкомпьютерных комплексов и обладающих различными свойствами: вычислительные узлы были задействованы монопольно или подключались в моменты простоя [Воеводин и др., 2003; Медведик и др., 2005].

Еще одна серия задач — создание новых лекарственных препаратов методом виртуального докинга и скрининга большого числа соединений — решалась совместно с двумя научными группами: подразделениями РАМН и факультетом биоинженерии и биоинформатики МГУ. Подобные задачи исключительно удачно ложатся на схему распределенных вычислений: один и тот же код выполняется над большим множеством входных файлов. Распределенные среды для решения этих задач строились на основе крупнейших российских суперкомпьютерных центров (МГУ, Томский государственный университет (ТГУ), Южно-Уральский государственный университет (ЮУрГУ), УГАТУ и других), причем ресурсы использовались как монопольно, так и через штатные системы очередей суперкомпьютеров. По результатам одного из проведенных расчетов был открыт и запатентован новый ингибитор тромбина, который в настоящий момент проходит испытания в качестве лекарства от тромбозов [Сулимов и др., 2006*а*, *б*; Соболев, 2007].

Система X-Com может применяться как основа для построения сервисов по распределению заданий на доступные вычислительные ресурсы. Так, совместно с компаниями «Тесис» и «Сигма Технология» в 2009–2010 гг. был реализован программный комплекс для решения оптимизационных гидродинамических задач. Программный комплекс объединял оптимизатор IOSO, решатель FlowVision и систему X-Com. Последняя отвечала за взаимодействие между оптимизатором, установленным на рабочем месте пользователя комплекса и генерирующим пакеты заданий для решателя, и системой очередей, через которую осуществлялся запуск пакета FlowVision на суперкомпьютерах МГУ [Жуматий и др., 2010].

На суперкомпьютере СКИФ МГУ «Чебышев» с помощью системы X-Com был реализован сервис выполнения пакетов однопроцессорных задач на суперкомпьютерах. Основная задача сервиса — дополнение системы управления задачами Cleo функциональностью по группировке нескольких однопроцессорных задач на один узел суперкомпьютера для повышения эффективности

использования ресурсов вычислительной системы [Соболев, Стефанов, 2009]. Кроме того, система X-Com применялась для исследования свойств прикладных задач на процессорном полигоне НИВЦ МГУ. В рамках этой работы проводилось сравнение времени выполнения приложения, откомпилированного с использованием различных компиляторов и запускаемого по одному или более процессов на имеющихся в полигоне программно-аппаратных платформах. Таким образом оценивалась эффективность работы приложения в зависимости от набора факторов (архитектура узла, компилятор и его опции, наличие или отсутствие параллельно работающих процессов).

4. НАПРАВЛЕНИЯ РАЗВИТИЯ ПРОЕКТА X-COM

Именно на вопросах эффективности сосредоточены в настоящее время исследования вокруг системы X-Com. Основная задача — создание комплекса характеристик для оценки качества распределенных компьютерных систем и сред, а также методов и инструментов для определения этих характеристик. Для традиционных сильно связанных вычислительных систем уже имеется устоявшийся набор подобных характеристик: производительность (пиковая и реальная), эффективность, энергоэффективность, свойства коммуникационной среды (пропускная способность и латентность). Эти характеристики понятны и хорошо документированы, их всегда можно найти в описаниях компьютерных систем и их компонентов или же определить по результатам выполнения тестов. На их основе пользователь может ранжировать вычислительные системы и, что самое важное, оценить, насколько быстрым, точным, экономичным окажется решение прикладных задач с известной ему структурой на конкретной высокопроизводительной системе.

Иная ситуация возникает при переходе от сильно связанных компьютерных систем, таких как кластерные, к распределенным вычислительным системам и средам. Распределенная среда динамична, состав ресурсов может меняться в ходе расчета, поэтому понятие пиковой производительности в традиционном смысле здесь оказывается не применимо, так как ее вычисление основывается на точном знании количества и свойств ресурсов. Аналогичная проблема и с реальной производительностью — из-за постоянного изменения состава среды серия одинаковых тестов от запуска к запуску может давать различные результаты. Вследствие неоднородности и географической распределенности сегментов среды невозможно однозначно определить и коммуникационные характеристики среды — для каждого сегмента они могут быть свои и также непостоянны. На практике все это приводит к сложности определения такого, казалось бы, тривиального показателя, как априорная оценка времени выполнения расчета. Хотя именно этот показатель может оказаться критичным для планирования расчета в тех случаях, когда компьютерные ресурсы предоставляются с жестким ограничением по времени.

Различные смыслы могут вкладываться в понятие эффективности. Можно понимать ее как эффективность проведения распределенного расчета и определять, например, как отношение времени, затраченного на организацию и поддержку распределенного расчета, ко времени проведения самого расчета, т. е.

оценивать накладные расходы программно-аппаратной среды. Можно оценивать эффективность расчета по количеству избыточных вычислений и/или потерь, неизбежных при обеспечении надежности методом дублирования расчетов. Можно, наконец, выстраивать теоретический план оптимального (по времени, стоимости и т. д.) распределения задачи на ресурсы и сравнивать их с реальным распределением [Хританков, 2009]. Эти оценки применимы к уже завершившимся расчетам, но и перед началом расчета были бы полезны априорные оценки и методы, позволяющие оптимальным образом сконфигурировать задачу при наличии хотя бы минимальной информации о свойствах среды. Однако на сегодняшний день устоявшихся оценок и методов их определения для распределенных компьютерных сред не существует.

Для описания расчетов, выполненных с помощью системы X-Com, достаточно длительное время использовался следующий набор характеристик: суммарная пиковая производительность среды, серверная и клиентская эффективность. Суммарная пиковая производительность вычислялась как сумма пиковых производительностей всех вычислительных узлов, подключавшихся к расчету хотя бы один раз. Серверная эффективность определялась как отношение времени только вычислений к полному времени обработки всех порций, показывая тем самым уровень накладных расходов на организацию распределенного расчета. В качестве клиентской эффективности, характеризующей динамичность среды, бралось соотношение числа отправленных сервером порций и числа полученных им результатов.

Однако ряд последних вычислительных экспериментов, в которых суперкомпьютерные ресурсы задействовались преимущественно через системы очередей, показал, что при расчетах в подобных режимах приведенные оценки не учитывают существенных особенностей среды. Модули взаимодействия с системами очередей буферизируют как входящие, так и исходящие порции, и не отслеживают конкретное состояние задач в очереди. Кроме того, модули взаимодействия на каждой вычислительной системе настраиваются вручную в соответствии с политиками использования суперкомпьютерных ресурсов. Поэтому крайне затруднено автоматическое определение реального числа одновременно работающих над задачей узлов или клиентских процессов. Очевидно, что добавление к накладным расходам времени ожидания в очереди приводит к снижению показателя серверной эффективности, а буферизация снижает клиентскую эффективность. Но в данном случае снижение чисто формальное, ведь в иных режимах работать с суперкомпьютерными ресурсами практически нереально. Поэтому одно из актуальных направлений работ по проекту X-Com — формирование комплекса новых оценок и характеристик с одновременным повышением уровня детализации информации обо всех этапах расчета.

Естественным развитием этого направления станет внедрение механизмов анализа качества распределенных сред в технологическую платформу X-Com. Это позволит оптимизировать выполнение масштабных вычислительно емких расчетов по заданным критериям. Запуск серии тестов, предшествующий запуску прикладной задачи, автоматически сконфигурирует задачу, платформу и среду в соответствии с заданными условиями (минимальное время решения задачи, приоритетное использование тех или иных ресурсов, минимальный объем пере-

даваемых по сетям данных). Результатом работ по этому направлению, условно названному «распределенный Linpack», станет методика и инструментарий для определения свойств распределенных сред, оценки поведения приложений в таких средах и выдачи рекомендаций по оптимизации распределенного приложения для среды с определенными свойствами.

Возможным продолжением этих работ может быть создание методики и инструментария оценки качества услуг, предоставляемых облачными сервисами. Такие сервисы в настоящее время приобретают все большую популярность, их предоставляют как крупные и известные IT-корпорации (Amazon, IBM, Google), так и небольшие коммерческие компании. Высокий уровень абстракции предоставляемых сервисов и закрытость технологий приводит к тому, что пользователь вынужден полагаться на обещания поставщика по качеству и составу предоставляемых услуг и ресурсов, при этом он не имеет возможности проконтролировать эти параметры самостоятельно. Проектируемый инструментарий должен обеспечить такую возможность.

ЛИТЕРАТУРА

[Воеводин, 2007] *Воеводин Вл. В.* Решение больших задач в распределенных вычислительных средах. // Автоматика и телемеханика. 2007. № 5. С. 32–45.

[Воеводин, Воеводин, 2002] *Воеводин В. В.*, *Воеводин Вл. В.* Параллельные вычисления. СПб.: БХВ-Петербург, 2002. 608 с.

[Воеводин, Филамофитский, 2003] *Воеводин Вл. В.*, *Филамофитский М. П.* Суперкомпьютер на выходные // Открытые системы. 2003. № 5. С. 43–48.

[Воеводин и др., 2003] *Воеводин В. В.*, *Смирнов Ю. Г.*, *Филамофитский М. П.*, *Цупак А. А.* Решение задачи дифракции на диэлектрическом теле методом объемных интегральных уравнений на многопроцессорных системах // Актуальные проблемы науки и образования: Тр. Международ. юбилейного симп. Пенза: Информац.-изд. центр ПГУ, 2003. С. 5–8.

[Воеводин и др., 2009] *Воеводин Вл. В.*, *Жолудев Ю. А.*, *Соболев С. И.*, *Стефанов К. С.* Эволюция системы метакомпьютинга X-Com // Вест. Нижегород. гос. ун-та им. Н. И. Лобачевского. 2009. № 4. С. 157–164.

[Жуматий и др., 2010] *Жуматий С. А.*, *Соболев С. И.*, *Стефанов К. С.* Решение оптимизационных гидродинамических задач в распределенной среде на основе вычислительных ресурсов МГУ // Вычислит. методы и программирование. 2010. Т. 11. № 2. С. 66–69.

[Медведик и др., 2005] *Медведик М. Ю.*, *Смирнов Ю. Г.*, *Соболев С. И.* Параллельный алгоритм расчета поверхностных токов в электромагнитной задаче дифракции на экране // Вычисл. методы и программирование. 2005. Т. 6. № 1. С. 86–95.

[Соболев, 2007] *Соболев С. И.* Использование распределенных компьютерных ресурсов для решения вычислительно сложных задач // Системы управления и информац. технологии. 2007. № 1–3 (27). С. 391–395.

[Соболев, 2010*а*] *Соболев С. И.* Иерархические методы улучшения масштабируемости и эффективности распределенных расчетов в системе метакомпьютинга X-Com // Параллельные вычислит. технологии (ПаВТ'2010): Тр. Международ. научной конф. Уфа, 29 марта – 2 апреля 2010 г. Челябинск: Изд. центр ЮУрГУ, 2010. С. 346–352.

[Соболев, 2010*б*] *Соболев С. И.* Интеграция системы метакомпьютинга X-Com с системами управления прохождением заданий суперкомпьютерных комплексов // 4-я Международ. конф. «Распределенные вычисления и грид-технологии в науке и образовании». Дубна, 28 июня – 3 июля 2010: Сб. тр.

[Соболев, Стефанов, 2009] *Соболев С. И.*, *Стефанов К. С.* Сервис выполнения пакетов однопроцессорных задач на суперкомпьютере СКИФ МГУ «Чебышев» // Материалы Международ. научно-техн. конф. «Многопроцессорные вычислит. системы (МВУС-2009)». Таганрог: Изд-во ТТИ ЮФУ. 2009. Т. 1. С. 225–227.

[Сулимов и др., 2006*а*] *Сулимов В. Б.*, *Романов А. Н.*, *Григорьев Ф. В.*, *Кондакова О. А.*, *Лущекина С. В.*, *Соболев С. И.* Оценка энергии связывания белок-лиганд с учетом растворителя и программа докинга SOL: принцип работы и характеристики // Сб. материалов 8-го Российского национального конгресса «Человек и лекарство». 3 апр. 2006. С. 37.

[Сулимов и др., 2006*б*] *Сулимов В. Б.*, *Романов А. Н.*, *Григорьев Ф. В.*, *Кондакова О. А.*, *Сулимов А. В.*, *Жабин С. Н.*, *Соболев С. И.* Веб-ориентированная система молекулярного моделирования Keenbase для разработки новых лекарств // Тр. Всерос. науч. конф. «Науч. сервис в сети Интернет: технологии параллельного программирования». Новороссийск, 18–23 сент. 2006. С. 170–172.

[Хританков, 2009] *Хританков А. С.* Анализ производительности распределенных вычислительных комплексов на примере системы X-Com // Тр. Всерос. суперкомпьютерной конф. «Науч. сервис в сети Интернет: масштабируемость, параллельность, эффективность». Новороссийск, 21–26 сент. 2009. С. 46–52.

X-COM TECHNOLOGY FOR DISTRIBUTED COMPUTING: ABILITIES, PROBLEMS, PERSPECTIVES

V. V. Voevodin, S. I. Sobolev

Research Computing Center of M. V. Lomonosov Moscow State University

Distributed computing, or metacomputing, is a promising technology for solving large problems by joining resources of heterogeneous computer systems. RCC MSU develops its own metacomputing system, X-Com, as a platform for setting up and performing of distributed computations. The article devoted to X-Com abilities, solved problems and development perspectives.

Keywords: large problems, distributed computing, distributed computational environments, heterogeneous environments, metacomputing, X-Com.

Voevodin Vladimir Valentinovich — vice-director, doctor of physical and mathematical sciences, corr. member of Russian academy of sciences, e-mail: voevodin@parallel.ru.

Sobolev Sergey Igorevich — scientist, candidate of physical and mathematical sciences, e-mail: sergeys@parallel.ru.

Содержание

От организаторов 3

Швецов А. В., *Сатанин А. М.*, *Гельман А. И.*
Управление населенностями кубитов путем воздействия электромагнитных импульсов 9

Денисенко М. В., *А. М. Сатанин А. М.*
Переходы Ландау – Зинера между состояниями взаимодействующих кубитов 24

Аблаев Ф. М.
Квантовые ветвящиеся программы — новая парадигма модели квантовых алгоритмов 36

Крюков А. П., *Демичев А. П.*, *Ильин В. А.*, *Шамардин Л. В.*
Основные подходы к построению грид-инфраструктуры Национальной нанотехнологической сети 51

Васенин В. А., *Шундеев А. С.*
Архитектурно-технологические аспекты эволюции грид 69

Романов Алексей А., *Романов Александр А.*, *Урличич Ю. М.*, *Буравин А. Е.*
Концептуальные подходы к созданию перспективных космических систем . . . 92

Иванников В. П.
Что такое СПО 105

Скобелев П. О., *Иващенко А. В.*, *Андреев А. В.*, *Бабанин И. О.*
Мультиагентные технологии для управления распределением производственных ресурсов в реальном времени 110

Зайцев Ф. С., *Шишкин А. Г.*, *Сычугов Д. Ю.*, *Лукаш В. Э.*, *Митришкин Ю. В.*, *Хайрутдинов Р. Р.*, *Степанов С. В.*, *Сучков Е. П.*
Структура и функциональные возможности комплекса имитационного моделирования «Виртуальный токамак» 123

Коноплев В. В.
Текущие и перспективные технологии виртуализации как платформа для организации облачных вычислений: сравнение и анализ 135

Граничин О. Н.
Характеристики перспективных принципиально новых компьютерных устройств и систем 147

Пойда А. А., *Жижин М. Н.*, *Медведев Д. П.*, *Москвитин А. Е.*, *Андреев А. В.*
Распределенное детектирование событий в многомерных потоках данных в беспроводных сенсорных сетях 162

Воеводин Вл. В., *Соболев С. И.*
Технология распределенных вычислений X-Com: возможности, задачи, направления развития 183

055(02)2

Ротапринт ИКИ РАН
117997, Москва, Профсоюзная, 84/32

Подписано к печати 22.11.2011

Заказ 3265 Формат $70\times100^{1}/_{16}$ Тираж 100 15,6 усл.-печ. л.