ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

10(146) 2008

ТЕОРЕТИЧЕСКИЙ И ПРИКЛАДНОЙ НАУЧНО-ТЕХНИЧЕСКИЙ ЖУРНАЛ

Издается с ноября 1995 г.

УЧРЕДИТЕЛЬ
Издательство "Новые технологии"

СОДЕРЖАНИЕ

Аннотации статей размещены на сайте журнала по адресу http://www.**informika.ru/text/magaz/it/** или **http://novtex.ru/IT.**

Журнал входит в Перечень научных журналов, в которых по рекомендации ВАК РФ должны быть опубликованы научные результаты диссертаций на соискание ученой степени доктора наук.

УДК 004.27

А. Л. Стемпковский, акад. РАН, директор,
Н. Н. Левченко, канд. техн. наук, ст. науч. сотр.,
А. С. Окунев, канд. техн. наук, ст. науч. сотр.,
В. В. Цветков, канд. техн. наук, ст. науч. сотр.,
Институт проблем проектирования
в микроэлектронике РАН, г. Москва
E-mail: stal108@ippm.ru, nick@ippm.ru

Параллельная потоковая вычислительная система — дальнейшее развитие архитектуры и структурной организации вычислительной системы с автоматическим распределением ресурсов

Рассматривается базовая архитектура и структурная организация параллельной потоковой вычислительной системы (ППВС). Оцениваются достоинства и преимущества использования нетрадиционной модели вычислений с управлением потоком данных, аппаратной ее реализации, описываются общие принципы функционирования ППВС. Предлагаются варианты аппаратной реализации системы в виде многоядерного кристалла, а также пути развития и масштабирования данной системы.

Ключевые слова: *архитектура параллельной потоковой вычислительной системы, параллельные вычисления, динамически формируемый контекст, нетрадиционная модель вычислений с управлением потоком данных.*

Введение

Отставание в оснащенности высокопроизводительными вычислительными системами научно-исследовательских институтов, университетов и вузов страны от мирового уровня очевидно, что, в свою очередь, грозит в ближайшем будущем утратой Россией ведущих позиций в создании наукоемких технологий и проектировании сложных объектов и процессов.

На сегодняшний день актуальны следующие задачи, имеющие непосредственное практическое значение, которые требуют высокой производительности вычислительных средств:

- моделирование различных систем и процессов, что фактически означает переход от физического моделирования к математическому;
- структурное исследование генов, молекулярное конструирование лекарств, прогноз экологических изменений и природных геофизических явлений, таких как климат, загрязнение среды, сейсмическая обстановка;
- проектирование сверхсложных радиоэлектронных комплексов, в том числе микропроцессоров и др.;
- многообразные задачи обработки изображения — распознавание образов, компрессия изображений, улучшение качества изображений, наложение эффектов на изображения и другие.

Как известно, основным методом увеличения производительности вычислительных средств является метод распараллеливания вычислительных процессов. На аппаратном уровне основным методом повышения производительности вычислительных систем в настоящее время является совершенствование технологии, а поскольку производительность одного процессора на используемой элементной базе близка к пределу, то увеличивают число процессоров в системе. Однако увеличение числа процессоров в подобных системах не всегда эффективно, ввиду того что падает коэффициент загрузки каждого процессора в системе и соответственно реальная производительность всей системы в целом, это подтверждается результатами, полученными на тестовых пакетах измерения производительности. К основным причинам падения реальной производительности многопроцессорных вычислительных систем можно отнести следующие:

- высокие накладные расходы на организацию параллельных вычислений, пропорциональные числу процессоров в системе;
- низкую пропускную способность подсистем оперативной памяти на реальных задачах (в случае нелинейной организации данных);
- необходимость синхронизации вычислительных процессов по данным.

Таким образом, в настоящее время актуальна задача создания вычислительной системы, которая обеспечивала бы следующее:

- снижение накладных расходов на распараллеливание вычислительных процессов;
- синхронизацию по данным в динамике вычислений, обеспечивающую высокую производи-

тельность при работе с разреженными и сложноорганизованными данными.

Вычислительная система с автоматическим распределением ресурсов

Ранее коллективом под руководством академика В. С. Бурцева велась работа по созданию высокопроизводительной вычислительной системы универсального назначения, реализующей гибридную потоковую модель вычислений с динамически формируемым контекстом. Были разработаны система команд, ассемблер, параллельный машинно-ориентированный язык высокого уровня, программная модель и макет системы. Данная система носила название "Вычислительная система с автоматическим распределением ресурсов" (ВСАРР).

Единицами информации, обрабатываемыми в вычислительной системе, являются токен и пакет (пара) (рис. 1, см. четвертую сторону обложки). Токены представляют собой структуру, которая содержит передаваемое данное, ключ, а также служебные признаки и атрибуты. Пакет представляет собой структуру данных, готовых к обработке на исполнительных устройствах системы. В поле ключа содержится информация, позволяющая однозначно идентифицировать токен или пакет в виртуальном адресном пространстве задачи.

Структура ВСАРР приведена на рис. 2 (см. четвертую сторону обложки). В общих чертах работа данной системы заключается в следующем:

- выполнение задачи начинается с поступления токенов на вход ассоциативной памяти;
- ассоциативная память (АП) сопоставляет данные с одинаковым контекстом, относящиеся к одной и той же программе узла[1] (то есть данные имеют одинаковый ключ), если нужное для сопоставления данное на момент прихода токена отсутствует, то он будет ожидать в АП поступления парного ему токена, в противном случае происходит формирование (по определенным правилам) пакета, который выдается на любое свободное ИУ;
- выходящие из исполнительных устройств данные (токены) содержат адрес узла, в который они должны поступить и где над ними должна выполниться соответствующая операция или программа. Токены через коммутационную среду поступают в АП для сопоставления.

Исполнительное устройство (ИУ) данной системы представляет собой функциональное устройство, предназначенное для выполнения программы узла над данными, поступающими на его вход.

Коммутатор токенов распределяет токены по модулям ассоциативной памяти с помощью хеш-функции. Коммутатор пар необходим для распределения готовых пар между свободными ИУ.

Более подробное описание системы приведено в работах [1, 2].

Параллельная потоковая вычислительная система

В настоящее время в Секторе архитектур высокопроизводительных вычислительных систем Института проблем проектирования в микроэлектронике Российской академии наук (ИППМ РАН) продолжается работа над новыми принципами организации вычислительного процесса и на их основе разрабатывается вычислительная система (ВС), использующая модель вычислений, управляемых потоком данных. Данная ВС — параллельная потоковая вычислительная система (ППВС), по архитектуре и структурной организации является дальнейшим развитием ВСАРР.

В архитектуру и модель вычислений параллельной потоковой вычислительной системы внесены значительные изменения (рис. 3, см. четвертую сторону обложки).

Целью перехода к ППВС является использование всех полезных наработок и устранение имеющихся недостатков ВСАРР. В частности, к недостаткам ВСАРР можно отнести: сложности, связанные с построением иерархии памяти для потоковых ВС; отсутствие возможности работы с различными хеш-функциями; большие коммутационные задержки при передаче пакетов "на значительные расстояния" и ряд других. Для ППВС сохраняется: аппаратная поддержка параллельного выполнения вычислительных процессов решаемой задачи; аппаратная поддержка управления распределением ресурсов вычислительных средств; возможность масштабирования системы на реальных задачах; реализация конвейеризации вычислительного процесса; аппаратно поддержанная синхронизация вычислительных процессов по данным; аппаратное (автоматическое) выявление неявного параллелизма программы, а также высокая производительность при работе с разреженными данными и данными сложной логической организации.

К основным особенностям архитектуры и структурной организации ППВС относятся:

- динамически формируемый контекст (ДФК); в отличие от предыдущих вычислительных систем, основанных на потоке данных, возможности, предоставляемые программисту ДФК, существенно расширены, поскольку ключ полностью доступен для программиста. Использование ДФК позволяет фактически в динамике "настраивать" токен по месту, т. е. программа

[1] Описание параллельного алгоритма состоит из описания узлов. Узлы соответствуют вершинам потокового графа программы. Каждый узел — это небольшая подпрограмма.

узла получает всю необходимую информацию из контекста о местоположении токена, и соответственно, знает, куда и какое число результирующих токенов послать в зависимости от выполняемых условий;

- маскирование; любой токен помимо ключа содержит в своей структуре и поле маски. Маска позволяет программисту задавать область видимости ключа путем его маскирования, причем разряды, попавшие под маску, не участвуют в обнаружении парного токена;
- кратность; задает число сопоставлений токена, при которых формируются пакеты; применение кратности позволяет снизить заполнение АП и нагрузку на коммутационную сеть;
- копии программ узлов содержатся в каждом из исполнительных устройств, что позволяет выполнять обработку программы узла в любом исполнительном устройстве;
- аппаратная реализация ассоциативной памяти.

Опишем теперь основные отличия ППВС от ВСАРР, которые позволили говорить о переходе к новой вычислительной системе, сохранив все достоинства архитектуры ВСАРР.

Отказ от коммутатора пакетов позволил сократить число ступеней конвейера, которые проходят данные и перейти к объединению ИУ и МАП в единый элемент — "ядро" (см. рис. 3). Это объединение позволило перейти от четырех уникальных элементов ВС (МАП, коммутатора пакетов, ИУ, коммутатора токенов) к двум элементам — ядру и коммутатору токенов, упростилась коммутационная среда — между ядрами циркулируют только токены.

Введение механизмов локализации вычислений дает возможность минимизировать число обменов между группами ядер, сгруппировав основную их часть внутри ядра или группы близко расположенных ядер, что повышает общую производительность системы, поскольку на нижнем уровне коммутационной среды (внутри ядра) пропускная способность ее максимальна. В этом случае существенно меньше сказываются задержки коммутационной среды на производительность системы в целом, выстраивается оптимальный конвейер системы. Механизм локализации имеет аппаратную поддержку в виде блоков хеширования для локализации (БХЛ) и позволяет программисту работать с некоторым набором аппаратно-реализованных хеш-функций, самостоятельно выбирая наилучшую под каждую конкретную задачу. В принципе, возможно использование уникальной хеш-функции для каждого узла задачи.

Помимо блока БХЛ, который расположен на выходе из ИУ, в структуру вычислительного ядра системы введен блок хеширования подмножества (БХП). Данный блок позволяет автоматически разбивать задачу на части — подмножества данных, что, в свою очередь, позволяет эффективно осуществлять регулирование заполнения ассоциативной памяти.

В архитектуру ВС введены многовходовые узлы. Реализация многовходовых узлов ранее предполагалась на программном уровне, т. е. на уровне транслятора многовходовые узлы разбиваются на двухвходовые [3]. В работах [4, 5] были сделаны шаги к внедрению аппаратной поддержки многовходовых узлов. Однако концепции многовходовых токенов и пакетов, по которой двухвходовые токен и пакет являются всего лишь частным случаем многовходовых, разработано не было. Сейчас ведется ее проработка.

Создание иерархии памяти позволяет уменьшить размер АП, снизить ее энергопотребление, использовать стандартные механизмы работы с памятью [6, 7].

Ведется дальнейшее развитие средств регулирования параллелизма, часть из них описана в работах [8, 9].

Рассматривается возможность введения векторных операций, которые позволят оптимально реализовывать имеющийся параллелизм задач.

Изменена топология коммутационной среды.

Результаты моделирования

На поведенческой блочно-регистровой модели ППВС было проведено моделирование работы системы и сбор статистики с использованием пакета тестовых задач. Исследовались варианты алгоритмов задач обработки изображения, в частности, задачи "Пространственный фильтр" и "Эрозия" для различных конфигураций системы и изображений различных размеров.

В результате проведенных экспериментов, связанных с изменением конфигурации системы, было выявлено, что с увеличением числа ядер время решения задачи сокращается, т. е. реализуется параллелизм алгоритма. Загрузка исполнительных устройств (ИУ) не падает до тех пор, пока достаточно параллелизма задачи для данного размера изображения.

Ниже приведены графики зависимости времени прохождения задачи "Эрозия" (в тактах) от числа ядер для изображения размером 200 × 200 пикселей (рис. 4). Приведены две кривые, отражающие конфигурации, связанные с различным способом ввода данных (через один канал, либо параллельный ввод через четыре канала). При увеличении числа ядер сокращается время выполнения задачи.

Была проведена серия экспериментов, связанных с введением в архитектуру ППВС многовходовых узлов. На рис. 5 приведены данные по результатам экспериментов на задаче "Пространст-

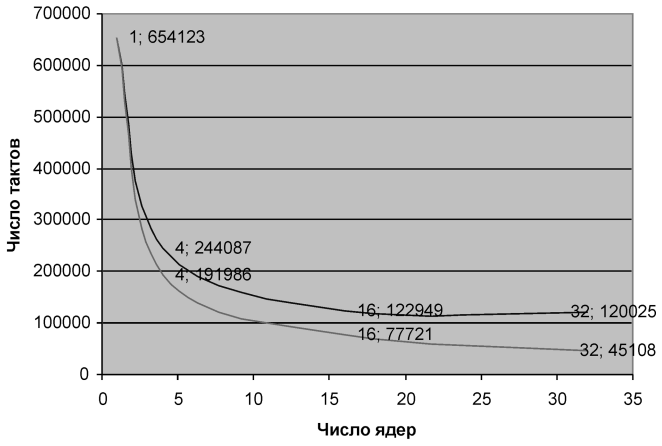

Рис. 4. Зависимость времени выполнения алгоритма "Эрозия_МР" от числа ядер и способа ввода данных для изображения размером 200 × 200 (32 ядра)

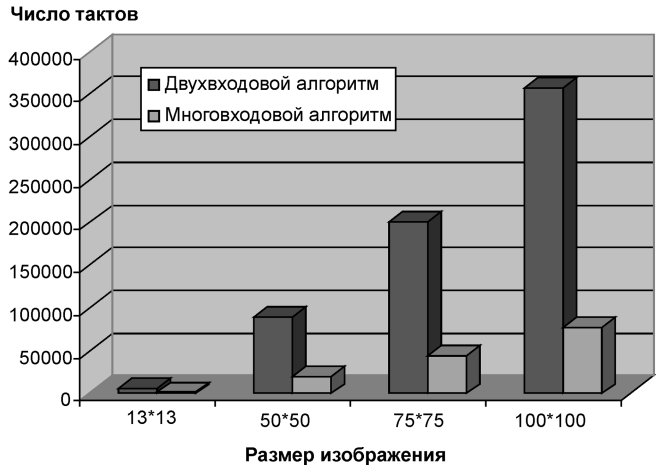

Рис. 5. Диаграмма сравнения алгоритмов "фильтр_Д" и "фильтр_М"

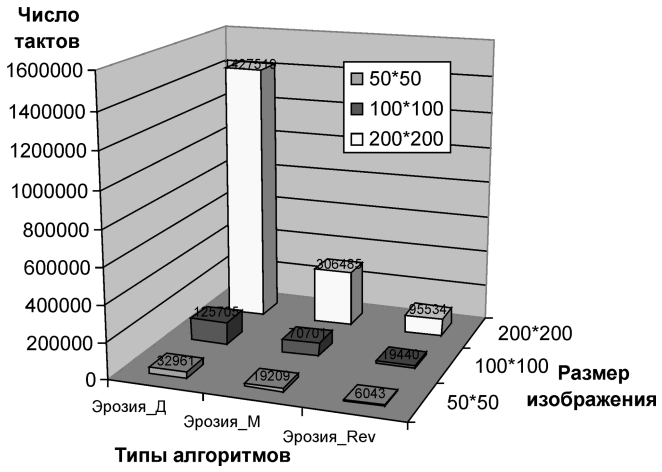

Рис. 6. Диаграмма сравнения времени выполнения алгоритмов "Эрозия"

венный фильтр", показывающие преимущества применения нового алгоритма, использующего многовходовые узлы ("фильтр_М") по сравнению со старым, в котором применяются только двухвходовые узлы ("фильтр_Д"). Число ядер в данном эксперименте — 16.

На рис. 6 представлены результаты обработки статистики по сравнению алгоритмов "эрозия_Д", "эрозия_М" и "эрозия_МР". При обработке изображения размером 200 × 200 пикселей выигрыш усовершенствованного алгоритма "эрозия_МР" над алгоритмом "эрозия_Д", который не использует проведенную адаптацию ППВС, составляет практически 15-кратное значение. С увеличением размерности изображения это значение будет только расти. Алгоритмы "эрозия_М" и "эрозия_МР" используют многовходовые узлы.

В качестве тестовой задачи для анализа эффекта локализации (использовалась локализующая хеш-функция, которая позволяет локализовать вычисления в пределах группы ядер, что уменьшает пересылки между ядрами) в ППВС был взят алгоритм перемножения матриц. Результаты прохождения этой задачи с размером матриц 128 × 128 на конфигурации 64 ядра с параллельным вводом по восьми каналам следующие:

- с использованием стандартной хеш-функции задача решается за 14 526 549 тактов (загрузка исполнительных устройств 7,7 %);
- с использованием локализующей хеш-функции задача решается за 1 555 716 тактов (загрузка исполнительных устройств 99 %).

Применение локализующей хеш-функции увеличивает реальную загрузку ИУ и, соответственно, сокращает время выполнения программы.

Приведенные данные свидетельствуют о переносе значительной части посылок токенов внутрь ядра (где параметры задержки коммутатора минимальны), что приводит к значительному снижению нагрузки на коммутационную сеть системы и возможности построения масштабируемой вычислительной системы без существенных потерь на коммутацию данных.

Возможности масштабирования ППВС

Базовым элементом системы является ядро. Вычислительное ядро, состоит из:

- модуля ассоциативной памяти (МАП);
- исполнительного устройства (ИУ);
- блока регулирования параллелизма (БРП);
- внутреннего коммутатора (ВКМ), связывающего ядро с другими ядрами и обеспечивающего обратную связь для токена, который, согласно функции распределения, направляется в МАП того же ядра;

- блока хеширования для локализации (БХЛ) вычислений по группам ядер;
- блок хеширования подмножества (БХП).

Группа ядер образует чип (кристалл). На рис. 7 приведен пример масштабирования параллельной потоковой вычислительной системы.

Рассматривается несколько вариантов структуры чипа ППВС.

Первый вариант представляет собой гомогенную структуру (рис. 8), в которой вычислительные ядра (ВЯ) идентичные друг другу объединены между собой в группы из четырех ядер посредством коммутатора каждый с каждым. В свою очередь, группы ядер объединяются между собой по аналогичному принципу.

Преимуществом данного подхода является более простое программирование, поскольку программа узла может выполняться на любом ядре, что позволяет значительно проще распределять граф вычислительного процесса по физическому пространству ядер.

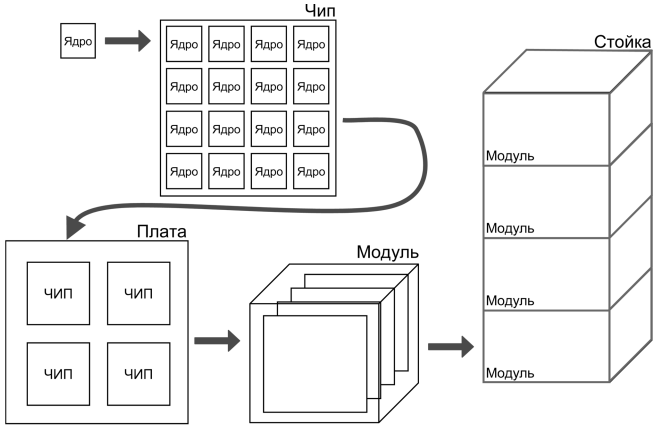

Рис. 7. Пример масштабирования ППВС

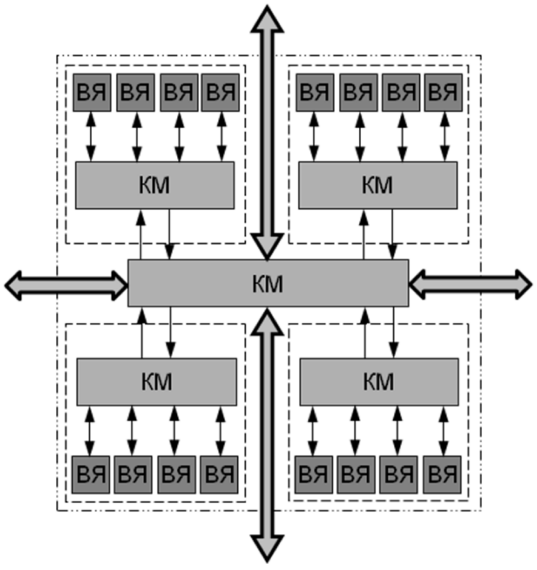

Рис. 8. Структура многоядерного кристалла без коммутатора пакетов

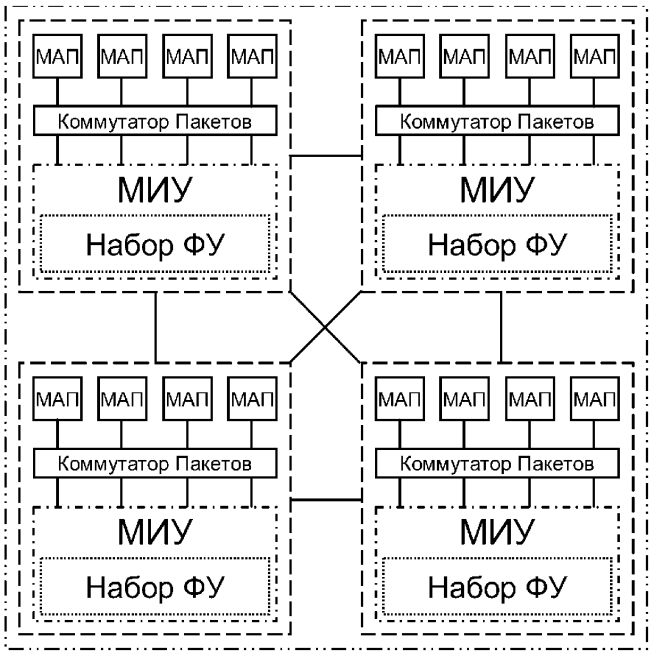

Рис. 9. Структура многоядерного кристалла с коммутатором пакетов

Отличие второго варианта от первого заключается в том, что на уровне группы ядер существует коммутатор пакетов (рис. 9). Это не будет мешать локализации данных, но вместе с тем, позволит более полно использовать имеющийся набор функциональных устройств (ФУ).

Группа из четырех ядер объединяется на внутреннем уровне, т. е. четыре МАП через локальный коммутатор пакетов обращаются к мультитредовому исполнительному устройству (МИУ) с единым набором функциональных устройств (ФУ). Программирование данной структуры многоядерного кристалла не отличается от предыдущего варианта, поскольку каждая из групп ядер может выполнить любую программу узла. Однако набор функциональных устройств МИУ может изменяться относительно базового набора ФУ, обеспечивающего аппаратное выполнение всей системы команд МИУ.

Выводы

Рассмотрена базовая архитектура и структурная организация параллельной потоковой вычислительной системы (ППВС). Модель вычислений, реализованная в ППВС, продолжает совершенствоваться в области локализации вычислений, построения иерархии памяти, планирования вычислений, разработки эффективных средств регулирования параллелизма и возможности масштабирования системы. Проведена оценка достоинств и преимуществ ППВС. Результаты моделирования задач на поведенческой модели системы

свидетельствуют о правильно выбранном направлении работ по введению новых архитектурных решений и изменениях в структурной организации ППВС.

Список литературы

1. **Бурцев В. С.** Новые принципы организации вычислительных процессов высокого параллелизма // Материалы Международной научно-технической конференции "Интеллектуальные и многопроцессорные системы — 2003". Т. 1. Таганрог: Изд-во ТРТУ, 2003.
2. **Бурцев В. С.** Выбор новой системы организации выполнения высокопараллельных вычислительных процессов, примеры возможных архитектурных решений построения суперЭВМ // Параллелизм вычислительных процессов и развитие архитектуры суперЭВМ. М.: Изд-во "НЕФТЬ И ГАЗ", 1997. 152 с.
3. **Левченко Н. Н.** Реализация трансляции многовходовых узлов языка высокого уровня ВМ нетрадиционной архитектуры потокового типа // С. А. Лебедев и развитие отечественной вычислительной техники. М.: МАИК, 2002. С. 172—175.
4. **Левченко Н. Н., Окунев А. С.** Методы аппаратной реализации многовходовых узлов вычислительной системы с автоматическим распределением ресурсов // Научно-теоретический журнал "Искусственный интеллект". ІПШІ "Наука і освіта". 2004. № 3. С. 149—156.
5. **Левченко Н. Н., Окунев А. С., Чумаченко Г. О., Хайлов И. К.** Варианты аппаратной поддержки многовходовых узлов в вычислительной системе с автоматическим распределением ресурсов//Проблемы и методы информатики. II научная сессия Института проблем информатики Российской академии наук, Москва, ИПИ РАН, 2005. С. 206—208.
6. **Климов А. В., Окунев А. С., Степанов А. М.** Распределение вычислений по модулям в вычислительной системе массового параллелизма // Приложение к журналу "Открытое образование". Материалы XXXIV международной конференции "Информационные технологии в науке, социологии, экономике и бизнесе". IT + SE'07. Майская сессия. Украина, Крым, Ялта-Гурзуф, 20—30 мая, 2007. С. 44—46.
7. **Хайлов И. К.** Варианты использования оперативной прямоадресуемой памяти и кэш-памяти в потоковой ЭВМ // Вычислительные машины с нетрадиционной архитектурой. СуперЭВМ. Сб. науч. тр. ИВВС РАН. Вып. 7. М.: Изд. ИВВС РАН, 1998. С. 31—39.
8. **Левченко Н. Н., Окунев А. С.** Аппаратно-программные методы ограничения "взрывного" параллелизма в вычислительной системе с автоматическим распределением ресурсов // Приложение к журналу "Открытое образование". Материалы XXXIII международной конференции "Информационные технологии в науке, образовании, телекоммуникации и бизнесе" (IT + S&E'06), Гурзуф, 2006. С. 42—43.
9. **Окунев А. С., Левченко Н. Н.** Методы преодоления блокировки вычислительного процесса в вычислительной системе с автоматическим распределением ресурсов // Приложение к журналу "Открытое образование". Материалы XXXIII международной конференции "Информационные технологии в науке, образовании, телекоммуникации и бизнесе" (IT + S&E'06), Гурзуф, 2006. С. 53—54.

УДК 519.873:004.732

А. И. Перегуда, д-р техн. наук, проф.,
Д. А. Тимашов, студент,
Обнинский государственный технический университет атомной энергетики (ИАТЭ)

Математическая модель надежности локальной вычислительной сети

Рассмотрена математическая модель надежности локальной вычислительной сети (ЛВС) произвольной топологии с восстанавливаемыми элементами. Получены соотношения для нахождения асимптотических значений таких показателей надежности, как вероятность потери данных в ЛВС, вероятность исправного функционирования всей сети, вероятность функционирования сети со сниженной эффективностью.

Ключевые слова: *надежность, локальная сеть, математическая модель, вероятность отказа.*

Постановка задачи

Без высокоразвитых систем обработки информации в настоящее время практически невозможна эффективная работа организаций, следовательно, требуется обеспечить надежное функционирование этих систем. Поэтому необходимо оценивать показатели надежности систем, чтобы добиться оптимального сочетания надежности и стоимости системы. Современные информационные системы (ИС) построены на основе компьютерных технологий и состоят из программного обеспечения и аппаратного обеспечения. Эти ИС являются распределенными и строятся на основе локальных вычислительных сетей (ЛВС). Поскольку современное программное обеспечение является достаточно сложным, оно также вносит свой вклад в надежность системы.

Для описания данной системы будем использовать математические модели, построенные на основе абстрактных логических подсистем с общими для любой информационной системы функциями. Такая модель позволяет рассматривать аппаратные и программные причины отказов одновременно, поскольку с точки зрения последствий отказов системы безразлично, из-за какой именно ошибки недоступна информация. Абстрагирование от реального оборудования и рассмотрение логических подсистем позволяет получить методы расчета показателей надежности, которые можно применять к любым ЛВС вне зависимости от состава оборудования, масштабов системы и ее функций.

Под ЛВС будем понимать совокупность аппаратных и программных средств, реализующих

следующие основные функции: хранение, обработку, передачу и защиту данных.

ЛВС содержит:

- *сервер* — элемент вычислительной сети (ВС), включающий в себя систему хранения данных (СХД) со своими системой передачи данных (СПД) и системой безопасности (СБ);
- *клиент* — элемент ВС, включающий в себя систему обработки данных (СОД) со своими СПД и СБ;
- *концентратор* — служит для связи клиентов и сервера и состоит из СПД и СБ.

К системе хранения данных относятся аппаратные и программные средства, обеспечивающие функцию хранения данных. Функциями этой системы являются прием, хранение и выдача информации. Функциями системы обработки данных являются преобразование данных и связь СХД с пользователем. СБ выполняет функции контроля над работой остальных систем и сохранения целостности данных. Она должна либо предотвращать потерю информации, либо сигнализировать о невозможности защиты данных. Аварией ЛВС будем считать потерю данных. Под потерей данных понимаем либо реальное их уничтожение, либо невозможность в течение достаточно длительного интервала времени получить доступ к ним.

Нашей задачей является разработка математической модели надежности ЛВС с восстанавливаемыми элементами, учитывающей последствия отказов подсистем и получение соответствующих показателей надежности. Подобные задачи рассматривались ранее в различных работах. Например, в работах [1, 2] были получены выражения для вычисления вероятности потери данных для сетей с невосстанавливаемыми элементами. В работе [3] рассматривалась оценка надежности и эффективности резервированных сетевых структур с выделением двух групп узлов и коммуникационной подсистемы.

Решение задачи

Введем необходимые обозначения случайных величин:

χ — наработка до отказа СОД;

γ — время восстановления СОД;

φ_1, φ_2 и φ_3 — наработки до отказа СПД клиента, порта концентратора и сервера;

ψ_1, ψ_2 и ψ_3 — время восстановления СПД клиента, порта концентратора и сервера;

ω — случайная наработка до отказа СХД;

ε — время восстановления СХД;

ξ_1, ξ_2 и ξ_3 — наработки до скрытого отказа СБ клиента, порта концентратора и сервера;

η_1, η_2 и η_3 — время восстановления СБ клиента, порта концентратора и сервера.

Так как мы предполагаем, что рассматриваемая система состоит из восстанавливаемых подсистем, то включаем в рассмотрение T_1, T_2 и T_3 — периоды профилактик СБ клиента, порта концентратора и сервера и θ_1, θ_2 и θ_3 — продолжительности профилактик СБ клиента, порта концентратора и сервера. Будем предполагать, что все введенные функции распределения непрерывны, т. е. они обладают плотностями распределения, и кроме того все случайные величины взаимно независимы.

Рассмотрим подробнее процесс функционирования клиента. Очевидно, что СОД и СПД клиента соединены последовательно, так как необнаруженный отказ любой из них приводит к аварии клиента. Но тогда в общем случае процесс функционирования клиента не имеет моментов регенерации. Поэтому ограничимся случаем, когда процесс функционирования клиента имеет моменты регенерации. Предположим, что клиент функционирует так, что в процессе его функционирования возможны два варианта развития событий.

Во-первых, СОД может отказать раньше СПД:

- если СБ в этот момент исправна, то произойдет останов клиента и восстановление СОД, при этом предполагаем, что по окончании восстановления СОД можно считать, что СПД также полностью восстановлена;
- если СБ в этот момент неисправна, то произойдет авария клиента.

Во-вторых, СПД может отказать раньше СОД:

- если СБ в этот момент исправна, то произойдет останов клиента и восстановление СПД, при этом предполагаем, что по окончании восстановления СПД можно считать, что СОД также полностью восстановлена;
- если СБ не сработает, то произойдет авария клиента. Данное предположение выполняется, например, для экспоненциальных наработок до отказа элементов, поскольку экспоненциальное распределение обладает свойством "нестарения".

Будем предполагать, что проводится периодическая процедура контрольных профилактик СБ, которая выполняется не мгновенно. При этом считаем, что во время контрольной профилактики система безопасности не может выполнять свои функции.

Найдем вначале функцию распределения времени до первой аварии клиента. При данных предположениях i-й цикл регенерации процесса функционирования клиента имеет длительность, равную $(\chi_i + \gamma_i)J_{\chi_i \le \varphi_{1i}} + (\varphi_{1i} + \psi_{1i})J_{\varphi_{1i} < \chi_i}$, где

$J_{w \in A} = \begin{cases} 1, w \in A; \\ 0, w \notin A. \end{cases}$. Иными словами, если СОД отказала раньше СПД, то цикл регенерации состоит из наработки СОД до отказа и времени восстановления клиента после отказа СОД, а если СПД отказала раньше СОД, то цикл регенерации состоит из наработки до отказа СПД и времени восстановления клиента после отказа СПД. Тот цикл регенерации, с номером π, во время которого произойдет авария клиента, будет, соответственно, иметь длительность, равную $\chi_\pi J_{\chi_\pi \le \varphi_{1\pi}} + \varphi_{1\pi} J_{\varphi_{1\pi} < \chi_\pi}$. Наработка клиента до аварии будет, таким образом, складываться из $\pi - 1$ полных циклов регенерации и того цикла регенерации, на котором произошла авария. Тогда функцию распределения времени наработки до первой аварии клиента ρ можно записать так:

$$F_\rho(t) = P(\rho \leqslant t) = P\Big(\sum_{i=1}^{\pi-1}((\chi_i + \gamma_i)J_{\chi_i \le \varphi_{1i}} + (\varphi_{1i} + \psi_{1i})J_{\varphi_{1i} < \chi_i}) + \chi_\pi J_{\chi_\pi \le \varphi_{1\pi}} + \varphi_{1\pi} J_{\varphi_{1\pi} < \chi_\pi} \leqslant t\Big),$$

где π — номер цикла, на котором произошла авария клиента. Используя формулу полной вероятности, функцию $F_\rho(t)$ можно вычислить так:

$$F_\rho(t) = \sum_{n=1}^{\infty} P\Big(\sum_{i=1}^{n-1}((\chi_i + \gamma_i)J_{\chi_i \le \varphi_{1i}} + (\varphi_{1i} + \psi_{1i})J_{\varphi_{1i} < \chi_i}) + \chi_n J_{\chi_n \le \varphi_{1n}} + \varphi_{1n} J_{\varphi_{1n} < \chi_n} \leqslant t\Big) q(1-q)^{n-1}, \quad (1)$$

где q — вероятность аварии клиента на цикле регенерации, т. е. вероятность того, что авария произошла именно на n-м цикле, равна $P(\pi = n) = q(1-q)^{n-1}$.

Найдем теперь q. Введем в рассмотрение две вспомогательные случайные величины U_n и V_n, учитывающие периодические проверки оборудования. Эти величины определены следующими соотношениями:

$$\sum_{i=1}^{n} \xi_{1i} + \sum_{i=1}^{n-1}\Big((T+\theta) - \Big\{\frac{\xi_{1i}}{T+\theta}\Big\}(T+\theta)\Big) + \sum_{i=1}^{n-1} \eta_{1i} = U_n;$$

$$\sum_{i=1}^{n} \xi_{1i} + \sum_{i=1}^{n}\Big((T+\theta) - \Big\{\frac{\xi_{1i}}{T+\theta}\Big\}(T+\theta)\Big) + \sum_{i=1}^{n} \eta_{1i} = V_n,$$

где U_n — время функционирования СБ клиента до n-го отказа; V_n — время функционирования СБ клиента до n-го восстановления; $[x]$ — целая часть x, $\{x\}$ — дробная часть x. Очевидно, что если отказ СПД или СОД произойдет во время восстановления СБ, то будет авария клиента. Условие аварии для этого случая записывается в виде $U_n \leqslant \chi \wedge \varphi_1 < V_n$. Но аварии клиента могут произойти также и в случаях, когда отказ СПД или СОД придется на время проведения контрольной профилактики. Условие аварии тогда запишется в виде последовательности событий:

$$V_{n-1} + T_1 \leqslant \chi \wedge \varphi_1 < V_{n-1} + (T_1 + \theta_1);$$

$$V_{n-1} + (T_1 + \theta_1) + T_1 \leqslant \chi \wedge \varphi_1 < V_{n-1} + 2(T_1 + \theta_1); \ldots$$

$$V_{n-1} + ([\xi_{1n}/(T_1 + \theta_1)] - 1)(T_1 + \theta_1) + T_1 \leqslant \chi \wedge \varphi_1 < V_{n-1} + [\xi_{1n}/(T_1 + \theta_1)](T_1 + \theta_1).$$

Опуская несложные преобразования, вероятность аварии за период регенерации запишем в виде

$$q = \sum_{n=1}^{\infty} \int_0^{\infty} (P(U_n \leqslant x) - P(V_n \leqslant x) + M \sum_{i=1}^{[\xi_{1n}/(T_1+\theta_1)]} (P(V_{n-1} + (i-1)(T_1 + \theta_1) + T_1 \leqslant x) - P(V_{n-1} + i(T_1 + \theta_1) \leqslant x)) dF_{\chi \wedge \varphi_1}(x).$$

Теперь следует представить q в виде суммы $q = q' + q''$ и слагаемые раздельно вычислить. Если обозначить $F_1(x) = P(T_1 + \theta_1 - \{\xi_{1i}/(T_1 + \theta_1)\}(T_1 + \theta_1) \leqslant x)$, то нетрудно увидеть, что q' допускает запись

$$q' = \sum_{n=1}^{\infty} \int_0^{\infty} (F_{\xi_1} * (F_{\xi_1} * F_{\eta_1} * F_1)^{*(n-1)}(x) - (F_{\xi_1} * F_{\eta_1} * F_1)^{*(n)}(x)) dF_{\chi \wedge \varphi_1}(x),$$

где $P(\xi_1 + \xi_2 \leqslant t) = \int_0^t P(\xi_1 \leqslant t - z) dF_2(z) = F_1 * F_2(t)$ — свертка функций распределений $F_1(t)$ и $F_2(t)$, а $F^{*(2)}(x) = F * F(x)$ и т. д.

Замечая, что $H_0(x) = \sum_{n=1}^{\infty} F_{\xi_1} * (F_{\xi_1} * F_{\eta_1} * F_1)^{*(n-1)}(x)$ и $H_1(x) = \sum_{n=1}^{\infty} (F_{\xi_1} * F_{\eta_1} * F_1)^{*(n)}(x)$ — это функции 0-восстановления и 1-восстановления альтернирующего процесса восстановления $\{\xi_{1i}, \eta_{1i}\}_{i>1}$, q' перепишем так:

$$q' = \int_0^{\infty} (H_0(x) - H_1(x)) dF_{\chi \wedge \varphi_1}(x) =$$

$$= \int_0^{\infty}\int_0^{x} (1 - F_1 * F_{\eta_1}(x-y)) dH_0(y) dF_{\chi \wedge \varphi_1}(x).$$

Вычислим теперь q'':

$$q'' = \sum_{n=1}^{\infty} \int_0^{\infty} M \sum_{i=1}^{[\xi_{1n}/(T_1+\theta_1)]} (P(V_{n-1} + (i-1) \times (T_1+\theta_1) + T_1 \leqslant x) - P(V_{n-1} + i(T_1+\theta_1) \leqslant x))) dF_{\chi \wedge \varphi_1}(x).$$

Меняя порядок суммирования и интегрирования и вводя обозначение $F_{2i}(x) = P(V_{n-1} + i(T_1 + \theta_1) \leqslant x)$, видим, что подынтегральное соотношение приводится к виду $M \sum_{i=1}^{\infty} \sum_{n=1}^{\infty} (F_{2i}(x + \theta_1 J_{i \leq [\xi_{1n}/(T_1+\theta_1)]}) - F_{2i}(x))$, и заметим, что $\sum_{n=1}^{\infty} F_{2i}(x) = H_{0i}(x)$ — функция 0-восстановления, тогда $q'' = \int_0^{\infty} M \sum_{i=1}^{\infty} (H_{0i}(x + \theta_1 J_{i \leq [\xi_1/(T_1+\theta_1)]}) - H_{0i}(x)) dF_{\chi \wedge \varphi_1}(x)$.

Используя основную теорему восстановления [4], представим q' следующим образом:

$$\lim_{x \to \infty} (H_0(x) - H_1(x)) =$$

$$= \frac{M\eta_1 + M(T_1 + \theta_1 + [\xi_1/(T_1+\theta_1)](T_1+\theta_1)) - M\xi_1}{M\eta_1 + M(T_1 + \theta_1 + [\xi_1/(T_1+\theta_1)](T_1+\theta_1))} = K_1.$$

Отсюда следует, что $q' \approx K_1$. Чтобы вывести асимптотическое соотношение для q'', используем теорему Блэкуэлла [4] и в итоге получаем:

$$\lim_{x \to \infty} M \sum_{i=1}^{\infty} (H_{0i}(x + \theta_1 J_{i \leq [\xi_1/(T_1+\theta_1)]}) - H_{0i}(x)) =$$

$$= \frac{M[\xi_1/(T_1+\theta_1)]\theta_1}{M\eta_1 + M(T_1 + \theta_1 + [\xi_1/(T_1+\theta_1)](T_1+\theta_1))} = K_2.$$

Отсюда следует, что $q'' \approx K_2$. Тогда, сложив q' и q'', представим q в виде

$$q \approx K_1 + K_2 =$$

$$= 1 - \frac{M\xi_1 - M[\xi_1/(T_1+\theta_1)]\theta_1}{M\eta_1 + M(T_1 + \theta_1 + [\xi_1/(T_1+\theta_1)](T_1+\theta_1))} =$$

$$= 1 - K_{\text{г}}^{\text{СБ}} = K_{\text{нг}}^{\text{СБ}},$$

где $K_{\text{г}}^{\text{СБ}}$ и $K_{\text{нг}}^{\text{СБ}}$ — стационарные коэффициенты готовности и неготовности СБ соответственно.

Используя обозначение операции свертки, запишем выражение (1) так:

$$F_{\rho}(t) = \sum_{n=1}^{\infty} (F_{\sigma_1}^{*(n-1)} * F_{\sigma_2})(t) K_{\text{нг}}^{\text{СБ}} (1 - K_{\text{нг}}^{\text{СБ}})^{n-1},$$

где $F_{\sigma_1}(t) = P((\chi + \gamma) J_{\chi \leq \varphi_1} + (\varphi_1 + \psi_1) J_{\varphi_1 < \chi} \leqslant t)$, $F_{\sigma_2}(t) = P(\chi J_{\chi \leq \varphi_1} + \varphi_1 J_{\varphi_1 < \chi} \leqslant t)$. Функции распределения $F_{\sigma_1}(t)$ и $F_{\sigma_2}(t)$ можно преобразовать следующим образом:

$$F_{\sigma_1}(t) = \int_0^t \Big(\int_0^{(t-y)} F_{\chi}(u) dF_{\varphi_1}(u) \Big) dF_{\gamma}(y) +$$

$$+ \int_0^t F_{\chi}(t-y)(1 - F_{\varphi_1}(t-y)) dF_{\gamma}(y) +$$

$$+ \int_0^t (1 - F_{\chi}(u)) F_{\psi_1}(t-u) dF_{\varphi_1}(u),$$

$$F_{\sigma_2}(t) = F_{\chi}(t) + F_{\varphi_1}(t) - F_{\chi}(t) F_{\varphi_1}(t).$$

Используя преобразование Лапласа—Стилтьеса, запишем:

$$\tilde{F}_{\rho}(s) = \sum_{n=1}^{\infty} (\tilde{F}_{\sigma_1}(s))^{n-1} \tilde{F}_{\sigma_2}(s) K_{\text{нг}}^{\text{СБ}} \times$$

$$\times (1 - K_{\text{нг}}^{\text{СБ}})^{n-1} = \frac{K_{\text{нг}}^{\text{СБ}} \tilde{F}_{\sigma_2}(s)}{1 - (1 - K_{\text{нг}}^{\text{СБ}}) \tilde{F}_{\sigma_1}(s)}, \qquad (2)$$

где $\tilde{F}_{\rho}(s) = \int_0^{\infty} \mathbf{e}^{-st} dF_0(t) = M\mathbf{e}^{-s\rho}$ — преобразование Лапласа—Стилтьеса функции распределения случайной величины ρ. Используя свойство преобразования Лапласа—Стилтьеса, имеем: $M\rho = M\sigma_2 + M\sigma_1(1 - K_{\text{нг}}^{\text{СБ}})/K_{\text{нг}}^{\text{СБ}}$.

Рассмотрим частный случай, когда все случайные величины имеют экспоненциальное распределения с параметром λ. Опуская несложные, но

громоздкие вычисления, запишем соотношения для $F_{\sigma_1}(t)$ и $F_{\sigma_2}(t)$:

$$F_{\sigma_1}(t) = 1 - \frac{\lambda_\chi \mathbf{e}^{-\lambda_\gamma t}}{\lambda_\chi + \lambda_{\varphi_1} - \lambda_\gamma} - \frac{\lambda_{\varphi_1} \mathbf{e}^{-\lambda_{\psi_1} t}}{\lambda_\chi + \lambda_{\varphi_1} - \lambda_{\psi_1}} +$$

$$+ \frac{(\lambda_\chi \lambda_\gamma + \lambda_{\varphi_1}\lambda_{\psi_1} - \lambda_\gamma \lambda_{\psi_1})\mathbf{e}^{-(\lambda_\chi + \lambda_{\varphi_1})t}}{(\lambda_\chi + \lambda_{\varphi_1} - \lambda_\gamma)(\lambda_\chi + \lambda_{\varphi_1} - \lambda_{\psi_1})} = 1 - A\mathbf{e}^{-\lambda_\gamma t} -$$

$$- B\mathbf{e}^{-\lambda_{\psi_1} t} + C\mathbf{e}^{-(\lambda_\chi + \lambda_{\varphi_1})t};\ F_{\sigma_2}(t) = 1 - \mathbf{e}^{-(\lambda_\chi + \lambda_{\varphi_1})t},$$

а их преобразования Лапласа—Стилтьеса имеют вид:

$$\tilde{F}_{\sigma_1}(s) = \frac{A_{\lambda_\gamma}}{\lambda_\gamma + s} + \frac{B\lambda_{\psi_1}}{\lambda_{\psi_i} + s} - \frac{C(\lambda_\chi + \lambda_{\varphi_1})}{\lambda_\chi + \lambda_{\varphi_1} + s};$$

$$\tilde{F}_{\sigma_2}(s) = \frac{\lambda_\chi + \lambda_{\varphi_1}}{s + \lambda_\chi + \lambda_{\varphi_1}}.$$

Используя соотношение (2), $\tilde{F}_\rho(s)$ запишем так:

$$\tilde{F}_\rho(s) = \tilde{D}(s)/\tilde{E}(s),$$

где $\tilde{D}(s) = K_{\text{нг}}^{\text{СБ}}(\lambda_\chi + \lambda_{\varphi_1})(s + \lambda_\gamma)(s + \lambda_{\psi_1})$;

$\tilde{E}(s) = (s+\lambda_\gamma)(s+\lambda_{\psi_1})(s+\lambda_\chi+\lambda_{\varphi_1}) - (1 - K_{\text{нг}}^{\text{СБ}}) \times$

$\times (A\lambda_\gamma(s + \lambda_{\psi_1})(s + \lambda_\chi + \lambda_{\varphi_1}) + B\lambda_{\psi_1}(s + \lambda_\gamma)(s + \lambda_\chi +$

$+ \lambda_{\varphi_1}) - C(\lambda_\chi + \lambda_{\varphi_1})(s + \lambda_\gamma)(s + \lambda_{\psi_1}))$.

Вычисляя обратное преобразование, получаем

$$F_\rho(t) = 1 + \frac{K_{\text{нг}}^{\text{СБ}}(\lambda_\chi + \lambda_{\varphi_1})(s_1 + \lambda_\gamma)(s_1 + \lambda_{\psi_1})}{s_1(s_1 - s_2)(s_1 - s_3)}\mathbf{e}^{s_1 t} +$$

$$+ \frac{K_{\text{нг}}^{\text{СБ}}(\lambda_\chi + \lambda_{\varphi_1})(s_2 + \lambda_\gamma)(s_2 + \lambda_{\psi_1})}{s_2(s_2 - s_1)(s_2 - s_3)}\mathbf{e}^{s_2 t} +$$

$$+ \frac{K_{\text{нг}}^{\text{СБ}}(\lambda_\chi + \lambda_{\varphi_1})(s_3 + \lambda_\gamma)(s_3 + \lambda_{\psi_1})}{s_3(s_3 - s_1)(s_3 - s_2)}\mathbf{e}^{s_3 t},$$

где s_1, s_2 и s_3 — корни уравнения $\tilde{E}(s) = 0$.

Среднее время до первой аварии запишем в виде

$$M\rho = 1/(\lambda_\chi + \lambda_{\varphi_1}) + (A/\lambda_\gamma + B/\lambda_{\psi_1} -$$

$$- C/(\lambda_\chi + \lambda_{\varphi_1}))(1 - K_{\text{нг}}^{\text{СБ}})/K_{\text{нг}}^{\text{СБ}}.$$

Вычислим теперь $K_{\text{нг}}^{\text{СБ}}$. Опуская несложные вычисления, запишем соотношение для коэффициента неготовности СБ:

$$K_{\text{нг}}^{\text{СБ}} = 1 -$$

$$- \frac{1/\lambda_{\xi_1} - \theta_1 \mathbf{e}^{-\lambda_{\xi_1}(T_1+\theta_1)}/(1 - \mathbf{e}^{-\lambda_{\xi_1}(T_1+\theta_1)})}{1/\lambda_{\eta_1} + (T_1 + \theta_1) + (T_1 + \theta_1)\mathbf{e}^{-\lambda_{\xi_1}(T_1+\theta_1)}/(1 - \mathbf{e}^{-\lambda_{\xi_1}(T_1+\theta_1)})}.$$

Показатели надежности порта вычисляются по той же схеме. Опуская некоторые промежуточные вычисления, приведем сразу результат. Функция распределения времени до первой аварии порта $F_{\rho'}(t)$:

$$\tilde{F}_{\rho'}(s) = \frac{q_2 \tilde{F}_{\varphi_2}(s)}{1 - (1 - q_2)\tilde{F}_{\varphi_2}(s)\tilde{F}_{\psi_2}(s)}.$$

Среднее время до первой аварии порта $M\rho'$:

$$M\rho' = M\varphi_2 + (M\varphi_2 + M\psi_2)(1 - q_2)/q_2;$$

$$q_2 \approx K_{\text{нг}}^{\text{СБ}} =$$

$$= 1 - \frac{M\xi_2 - M[\xi_2/(T_2 + \theta_2)]\theta_2}{M\eta_2 + M(T_2 + \theta_2 + [\xi_2/(T_2 + \theta_2)](T_2 + \theta_2))}.$$

В случае экспоненциально распределенных случайных величин соответственно запишем:

$$F_{\rho'}(t) = 1 + \frac{K_{\text{нг}}^{\text{СБ}}\lambda_{\varphi_2}(s_1 + \lambda_{\psi_2})}{s_1(s_1 - s_2)}\mathbf{e}^{s_1 t} +$$

$$+ \frac{K_{\text{нг}}^{\text{СБ}}\lambda_{\varphi_2}(s_2 + \lambda_{\psi_2})}{s_2(s_2 - s_1)}\mathbf{e}^{s_2 t};$$

$$s_{1/2} = (-(\lambda_{\varphi_2} + \lambda_{\psi_2}) \pm$$

$$\pm \sqrt{(\lambda_{\varphi_2} + \lambda_{\psi_2})^2 - 4\lambda_{\varphi_2}\lambda_{\psi_2}K_{\text{нг}}^{\text{СБ}}})/2;$$

$$K_{\text{нг}}^{\text{СБ}} = 1 -$$

$$- \frac{1/\lambda_{\xi_2} - \theta_2 \mathbf{e}^{-\lambda_{\xi_2}(T_2+\theta_2)}/(1 - \mathbf{e}^{-\lambda_{\xi_2}(T_2+\theta_2)})}{1/\lambda_{\eta_2} + (T_2 + \theta_2) + (T_2 + \theta_2)\mathbf{e}^{-\lambda_{\xi_2}(T_2+\theta_2)}/(1 - \mathbf{e}^{-\lambda_{\xi_2}(T_2+\theta_2)})};$$

$$M\rho' = \frac{1}{\lambda_{\varphi_2}} + \frac{1 - K_{\text{нг}}^{\text{СБ}}}{K_{\text{нг}}^{\text{СБ}}}\left(\frac{1}{\lambda_{\varphi_2}} + \frac{1}{\lambda_{\psi_2}}\right).$$

Показатели надежности для сервера вычисляются аналогично. Опять опуская вычисления, приведем сразу результаты.

Функция распределения времени до первой аварии сервера $F_{\rho''}(t)$:

$$\tilde{F}_{\rho''}(s) = \frac{K_{\text{нг}}^{\text{СБ}}\tilde{F}_{\sigma_2'}(s)}{1 - (1 - K_{\text{нг}}^{\text{СБ}})\tilde{F}_{\sigma_1'}(s)};$$

$$F_{\sigma_2'}(t) = F_\omega(t) + F_{\varphi_3}(t) - F_\omega(t)F_{\varphi_3}(t);$$

$$F_{\sigma_1'}(t) = \int_0^t \Big(\int_0^{t-y} F_\omega(u) dF_{\varphi_3}(u) \Big) dF_\varepsilon(y) +$$

$$+ \int_0^t F_\omega(t-y)(1 - F_{\varphi_3}(t-y)) dF_\varepsilon(y) +$$

$$+ \int_0^t (1 - F_\omega(u)) F_{\psi_3}(t-u) dF_{\varphi_3}(u).$$

Среднее время до первой аварии сервера $M\rho''$:

$$M\rho'' = M\sigma_2' + M\sigma_1'(1 - K_{\text{НГ}}^{\text{СБ}})/K_{\text{НГ}}^{\text{СБ}};$$

$$K_{\text{НГ}}^{\text{СБ}} = 1 - \frac{M\xi_3 - M[\xi_3/(T_3+\theta_3)]\theta_3}{M\eta_3 + M(T_3+\theta_3+[\xi_3/(T_3+\theta_3)](T_3+\theta_3))}.$$

Когда все случайные величины экспоненциально распределены, получаем:

$$F_{\sigma_1'}(t) = 1 - \frac{\lambda_\omega \mathbf{e}^{-\lambda_\varepsilon t}}{\lambda_\omega + \lambda_{\varphi_3} - \lambda_\varepsilon} - \frac{\lambda_{\varphi_3} \mathbf{e}^{-\lambda_{\psi_3} t}}{\lambda_\omega + \lambda_{\varphi_3} - \lambda_{\psi_3}} +$$

$$+ \frac{(\lambda_\omega\lambda_\varepsilon + \lambda_{\varphi_3}\lambda_{\psi_3} - \lambda_\varepsilon\lambda_{\psi_3})\mathbf{e}^{-(\lambda_\omega+\lambda_{\varphi_3})t}}{(\lambda_\omega + \lambda_{\varphi_3} - \lambda_\varepsilon)(\lambda_\omega + \lambda_{\varphi_3} - \lambda_{\psi_3})} =$$

$$= 1 - A'\mathbf{e}^{-\lambda_\varepsilon t} - B'\mathbf{e}^{-\lambda_{\psi_3} t} + C'\mathbf{e}^{-(\lambda_\omega+\lambda_{\varphi_3})t};$$

$$F_{\sigma_2'}(t) = 1 - \mathbf{e}^{-(\lambda_\omega+\lambda_{\varphi_3})t};$$

$$\widetilde{F}_{\sigma_1'}(s) = \frac{A'\lambda_\varepsilon}{\lambda_\varepsilon + s} + \frac{B'\lambda_{\psi_3}}{\lambda_{\psi_3} + s} - \frac{C'(\lambda_\omega + \lambda_{\varphi_3})}{\lambda_\omega + \lambda_{\varphi_3} + s};$$

$$\widetilde{F}_{\sigma_2'}(s) = \frac{\lambda_\omega + \lambda_{\varphi_3}}{s + \lambda_\omega + \lambda_{\varphi_3}};$$

$$\widetilde{F}_{\rho''}(s) = \frac{K_{\text{НГ}}^{\text{СБ}}(\lambda_\omega + \lambda_{\varphi_3})(s+\lambda_\varepsilon)(s+\lambda_{\psi_3})}{(s+\lambda_\varepsilon)(s+\lambda_{\psi_3})(s+\lambda_\omega+\lambda_{\varphi_3}) - (1-K_{\text{НГ}}^{\text{СБ}})(A'\lambda_\varepsilon(s+\lambda_{\psi_3})(s+\lambda_\omega+\lambda_{\varphi_3})} \rightarrow$$

$$\rightarrow \frac{}{+ B'\lambda_{\psi_3}(s+\lambda_\varepsilon)(s+\lambda_\omega+\lambda_{\varphi_3}) - C'(\lambda_\omega+\lambda_{\varphi_3})(s+\lambda_\varepsilon)(s+\lambda_{\psi_3}))};$$

s_1', s_2' и s_3' — полюса $\widetilde{F}_{\rho''}(s)$,

$$F_{\rho''}(t) = 1 + \frac{K_{\text{НГ}}^{\text{СБ}}(\lambda_\omega+\lambda_{\varphi_3})(s_1'+\lambda_\varepsilon)(s_1'+\lambda_{\varphi_3})}{s_1'(s_1'-s_2')(s_1'-s_3')}\mathbf{e}^{s_1' t} +$$

$$+ \frac{K_{\text{НГ}}^{\text{СБ}}(\lambda_\omega+\lambda_{\varphi_3})(s_2'+\lambda_\varepsilon)(s_2'+\lambda_{\varphi_3})}{s_2'(s_2'-s_1')(s_2'-s_3')}\mathbf{e}^{s_2' t} +$$

$$+ \frac{K_{\text{НГ}}^{\text{СБ}}(\lambda_\omega+\lambda_{\varphi_3})(s_3'+\lambda_\varepsilon)(s_3'+\lambda_{\psi_3})}{s_3'(s_3'-s_1')(s_3'-s_2')}\mathbf{e}^{s_3' t},$$

$$M\rho'' = \frac{1}{\lambda_\omega + \lambda_{\varphi_3}} + \frac{1 - K_{\text{НГ}}^{\text{СБ}}}{K_{\text{НГ}}^{\text{СБ}}}\Big(\frac{A'}{\lambda_\varepsilon} + \frac{B'}{\lambda_{\psi_3}} - \frac{C'}{\lambda_\omega + \lambda_{\varphi_3}}\Big);$$

$$K_{\text{НГ}}^{\text{СБ}} = 1 -$$

$$- \frac{1/\lambda_{\xi_3} - \theta_3 \mathbf{e}^{-\lambda_{\xi_3}(T_3+\theta_3)}/(1-\mathbf{e}^{-\lambda_{\xi_3}(T_3+\theta_3)})}{1/\lambda_{\eta_3} + (T_3+\theta_3) + (T_3+\theta_3)\mathbf{e}^{-\lambda_{\xi_3}(T_3+\theta_3)}/(1-\mathbf{e}^{-\lambda_{\xi_3}(T_3+\theta_3)})}.$$

Рассмотрим теперь процесс функционирования сети. Пусть сеть состоит из конечного числа элементов N. Пусть $\varphi(x) = \varphi(x_1, x_2, ..., x_N)$ — структурная функция сети, а $h(P_1, P_2, ..., P_N) = M\varphi(x) = P(\varphi(x) = 1)$ — функция надежности сети, где $P_i = P(x_i = 1) = Mx_i$ — вероятность безаварийной работы i-го элемента сети. Наработка до аварии ρ_i i-го элемента есть случайная величина. После аварии элемент восстанавливается в течение времени Δ_i, также случайного. Предполагаем, что восстановление полное. Таким образом, все время функционирования сети распадается на отдельные циклы, в каждом из которых сеть часть времени работает без аварий (множество интервалов времени Q^+), а остальное время затрачивается на устранение последствий аварии (множество интервалов времени Q^-). Множество интервалов времени, в течение которых i-й элемент работает без аварий, обозначим Q_i^+, а множество остальных интервалов обозначим Q_i^-; $P_i(t)$ — вероятность безаварийного функционирования i-го элемента в момент времени t. Введем дуальную к булевой функции $\varphi(x) = \varphi(x_1, x_2, ..., x_N)$ функцию множеств $A_1, A_2, ..., A_N$ $\bar{\varphi}(A) = \bar{\varphi}(A_1, A_2, ..., A_N)$, определенную так, что если A_i — множество тех моментов времени, в которые i-й элемент находится в исправном состоянии, то $\bar{\varphi}(A_1, A_2, ..., A_N)$ есть множество моментов исправной работы всей системы. Вероятность нахождения сети в аварийном состоянии $P_{\text{а}}$, очевидно, запишется в виде

$$P_{\text{а}}(t) = P(t \in Q^-) = 1 - P(t \in Q^+) =$$
$$= 1 - P(t \in \bar{\varphi}(Q_1^+, Q_2^+, ..., Q_N^+)) =$$
$$= 1 - P(\varphi(x_1(t), x_2(t), ..., x_N(t)) = 1) =$$
$$= 1 - h(P_1(t), P_2(t), ..., P_N(t)).$$

Используя формулу полной вероятности, $P_i(t)$ запишем в виде

$$P_i(t) = P(t \in Q_i^+) = \int_0^\infty \int_0^\infty P(t \in Q_i^+ | \rho_{i1} = x, \Delta_{i1} = y) dF_{\rho_i}(x) dG_{\Delta_i}(y),$$

где ρ_{i1} и Δ_{i1} — случайные времена до первой аварии и до первого восстановления соответственно. Процесс функционирования i-го элемента распадается на циклы регенерации длительностью $\rho_i + \Delta_i = \tau_i(\rho_i, \Delta_i)$. Отметим, что возможны два варианта: $\tau_i(\rho_i, \Delta_i) > t$ и $\tau_i(\rho_i, \Delta_i) \leqslant t$, тогда:

$$P_i(t) = \iint\limits_{\tau_i(x,y) \leq t} P(t \in Q_i^+ | \rho_{i1} = x, \Delta_{i1} = y) dF_{\rho_i}(x) dG_{\Delta_i}(y) + \iint\limits_{\tau_i(x,y) > t} P(t \in Q_i^+ | \rho_{i1} = x, \Delta_{i1} = y) dF_{\rho_i}(x) dG_{\Delta_i}(y) = I_1 + I_2.$$

Вычислим сначала I_2. Условие $\tau_i(\rho_i, \Delta_i) > t$ означает, что момент регенерации i-го элемента наступил после момента времени t и, следовательно, условие $t \in Q_i^+$ эквивалентно условию $t \in [0, \rho_{i1}]$. Тогда

$$I_2 = \iint\limits_{x+y>t} P(t \in [0, \rho_{i1}] | \rho_{i1} = x, \Delta_{i1} = y) dF_{\rho_i}(x) dG_{\Delta_i}(y) = \iint\limits_{x+y>t} (J_{t \in [0, x]}) dF_{\rho_i}(x) dG_{\Delta_i}(y) = 1 - F_{\rho_i}(t).$$

При рассмотрении ситуации $\rho_i + \Delta_i \leqslant t$ учтем, что так как в момент времени $\rho_i + \Delta_i$ имеет место момент регенерации, то $P(t \in Q_i^+ | \rho_{i1} = x, \Delta_{i1} = y) = P_i(t - x - y)$.

Тогда I_1 запишем в виде

$$I_1 = \iint\limits_{\tau_i(x,y) \leq t} P_i(t \in Q_i^+ | \rho_{i1} = x, \Delta_{i1} = y) dF_{\rho_i}(x) dG_{\Delta_i}(y) = \iint\limits_{\tau_i(x,y) \leq t} P_i(t - x - y) dF_{\rho_i}(x) dG_{\Delta_i}(y) = \int\limits_0^t P_i(t - z) dF_{\tau_i}(z).$$

Сложим теперь I_1 и I_2 и получим интегральное уравнение

$$P_i(t) = g_i(t) + \int\limits_0^t P_i(t - z) dF_{\tau_i}(z),$$

где $g_i(t) = \overline{F}_{\rho_i}(t)$, $F_{\tau_i}(t) = P(\rho_i + \Delta_i \leqslant t)$. Решение уравнения будем искать, используя преобразование Лапласа—Стилтьеса:

$$\tilde{P}_i(s) = \tilde{g}_i(s) + \tilde{P}_i(s) \tilde{F}_{\tau_i}(s).$$

Тогда

$$\tilde{P}_i(s) = \tilde{g}_i(s)/(1 - \tilde{F}_\tau(s)) = (1 - M\mathbf{e}^{-s\rho_i})/(1 - M\mathbf{e}^{-s(\rho_i + \Delta_i)}).$$

Найти обратное преобразование в общем виде невозможно, поэтому найдем асимптотическое соотношение для $P_i(t)$, применяя Тауберову теорему $\lim\limits_{t \to \infty} P_i(t) = \lim\limits_{s \to 0} \tilde{P}_i(s)$, тогда $P_i = \lim\limits_{t \to \infty} P_i(t) = M\rho_i/(M\rho_i + M\Delta_i) = K_{\text{г}i}$. Таким образом, предел $P_i(t)$ при $t \to \infty$ совпадает с коэффициентом готовности i-го элемента $K_{\text{г}i}$. Асимптотическое соотношение для вероятности нахождения сети в состоянии аварии $P_\text{а}$ тогда запишется в виде $P_\text{а} \approx 1 - h(K_{\text{г}1}, K_{\text{г}2}, ..., K_{\text{г}N})$.

Найдем теперь такие показатели надежности сети, как вероятность того, что сеть полностью исправна, и вероятность того, что сеть работает со сниженной эффективностью. Интервал времени, в течение которого элемент функционирует без аварий, разделим на два подмножества моментов времени:

- подмножество R_i^+, когда элемент исправно функционирует без остановов;
- подмножество R_i^-, когда элемент остановлен после срабатывания СБ и его восстанавливают.

Тогда найдем выражение для $P_i^+(t)$ — вероятности того, что элемент исправно функционирует без аварий и остановов. Рассмотрим вычисления на примере клиента. По формуле полной вероятности запишем:

$$P_{Ki}^+(t) = \int\limits_0^\infty \int\limits_0^\infty P(t \in R_i^+ | \rho_{i1} = x, \Delta_{i1} = y) dF_{\rho_i}(x) dG_{\Delta_i}(y).$$

Процесс функционирования i-го клиента распадается на циклы регенерации длительностью $\rho_i + \Delta_i = \tau_i(\rho_i, \Delta_i)$.

Отметим, что возможны два варианта: $\tau_i(\rho_i, \Delta_i) > t$ и $\tau_i(\rho_i, \Delta_i) \leqslant t$, тогда:

$$P_{Ki}^+(t) = \iint\limits_{\tau_i(x,y) \leq t} P(t \in R_i^+ | \rho_{i1} = x, \Delta_{i1} = y) dF_{\rho_i}(x) dFG_{\Delta_i}(y) + \iint\limits_{\tau_i(x,y) > t} P(t \in R_i^+ | \rho_{i1} = x, \Delta_{i1} = y) dF_{\rho_i}(x) dG_{\Delta_i}(y) = I_1 + I_2.$$

Вычислим сначала I_2. Условие $\tau_i(\rho_i, \Delta_i) > t$ означает, что момент регенерации i-го клиента наступил после момента времени t. Условие $t \in R_i^+$ означает, что в момент времени t клиент не находится в состоянии аварии и в то же время и не находится в состоянии останова. Тогда

$$I_2 = \iint\limits_{x+y>t} P(t \in R_i^+ | \rho_{i1} = x, \Delta_{i1} = y) dF_{\rho_i}(x) dG_{\Delta_i}(y) = \int\limits_t^\infty P(t \in R_i^+(x)) dF_{\rho_i}(x).$$

По формуле полной вероятности для $P(t \in R_i^+(x))$ при $x \geqslant t$ запишем:

$$P_{Ki}^{R^+}(t) = P(t \in R_i^+(x)) =$$
$$= \iiiint\limits_{\sigma_1(u,v,a,b) \leq t} P(t \in R_i^+(x) | \chi = u, \gamma = v,$$
$$\varphi_1 = a, \psi_1 = b) dF_{\chi,\gamma,\varphi_1,\psi_1}(u, v, a, b) +$$
$$+ \iiiint\limits_{\sigma_1(u,v,a,b) > t} P(t \in R_i^+(x) | \chi = u, \gamma = v, \varphi_1 = a,$$
$$\psi_1 = b) dF_{\chi,\gamma,\varphi_1,\psi_1}(u, v, a, b) = I_1' + I_2',$$

где $\sigma_1(u, v, a, b) = u \wedge a + vJ_{u<a} + bJ_{a \leqslant u}$. Вычислим сначала I_2':

$$I_2' = \iiiint\limits_{\sigma_1(u,v,a,b) > t} P(t \in R_i^+(x) | \chi = u, \gamma = v, \varphi_1 = a,$$
$$\psi_1 = b) dF_{\chi,\gamma,\varphi_1,\psi_1}(u, v, a, b) =$$
$$= \iint\limits_{u \wedge a > t} dF_{\chi,\varphi_1}(u, a) = 1 - F_{\chi \wedge \varphi}(t) = l(t).$$

Перепишем I_1' в виде

$$I_1' = \iiiint\limits_{\sigma_1(u,v,a,b) \leq t} P(t \in R_i^+(x) | \chi = u, \gamma = v,$$
$$\varphi_1 = a, \psi_1 = b) dF_{\chi,\gamma,\varphi_1,\psi_1}(u, v, a, b) =$$
$$= \int\limits_0^t P_{Ki}^{R^+}(t - z) dF_{\sigma_1}(z).$$

Тогда получаем интегральное уравнение

$$P_{Ki}^{R^+}(t) = l(t) + \int\limits_0^t P_{Ki}^{R^+}(t - z) dF_{\sigma_1}(z).$$

Его решение будем искать, используя преобразование Лапласа—Стилтьеса:

$$\tilde{P}_{Ki}^{R^+}(s) = \tilde{l}(s) + \tilde{P}_{Ki}^{R^+}(s) \tilde{F}_{\sigma_1}(s).$$

Тогда $\tilde{P}_{Ki}^{R^+}(s) = \dfrac{\tilde{l}(s)}{1 - \tilde{F}_{\sigma_1}(s)} = \dfrac{1 - M\mathbf{e}^{-s(\chi \wedge \varphi_1)}}{1 - M\mathbf{e}^{-s\sigma_1}}$.

Найти обратное преобразование в общем виде невозможно, поэтому найдем асимптотическое соотношение для $P_{Ki}^{R^+}(t)$, применяя Тауберову теорему $\lim\limits_{t \to \infty} P_{Ki}^{R^+}(t) = \lim\limits_{s \to 0} \tilde{P}_{Ki}^{R^+}(s)$. Тогда

$$P_{Ki}^{R^+} = \lim_{t \to \infty} P_{Ki}^{R^+}(t) = M(\chi \wedge \varphi_1)/M\sigma_1 = K_{\Gamma Ki}^{R^+}.$$

Здесь $M(\chi \wedge \varphi_1) = M\chi + M\varphi - \int\limits_0^\infty (1 - F_\chi(t) F_\varphi(t)) dt$,

$$M\sigma_1 = M((\chi + \gamma) J_{\chi \leq \varphi_1} + (\varphi_1 + \psi_1) J_{\varphi_1 < \chi}).$$

Тогда $I_2 = \int\limits_t^\infty K_{\Gamma Ki}^{R^+} dF_{\rho_i}(x) = K_{\Gamma Ki}^{R^+}(1 - F_\rho(t)) =$ $= g_i(t)$. При рассмотрении ситуации $\rho_i + \Delta_i^1 \leqslant t$ учтем, что поскольку в момент времени $\rho_i + \Delta_i^1$ имеет место момент регенерации, то

$$P(t \in R_i^+ | \rho_{i1} = x, \Delta_{i1} = y) = P_i^+(t - x - y).$$

Тогда I_1 запишем в виде

$$I_1 = \int\limits_0^t P_{Ki}^+(t - z) dF_{\tau_i}(z).$$

Сложим теперь I_1 и I_2 и получим интегральное уравнение

$$P_{Ki}^+(t) = g_i(t) + \int\limits_0^t P_{Ki}^+(t - z) dF_{\tau_i}(z).$$

Решение уравнения будем искать, используя преобразование Лапласа—Стилтьеса:

$$\tilde{P}_{Ki}^+(s) = \tilde{g}_i(s) + \tilde{P}_{Ki}^+(s) \tilde{F}_{\tau_i}(s).$$

Тогда $\tilde{P}_{Ki}^+(s) = \dfrac{\tilde{g}_i(s)}{1 - \tilde{F}_{\tau_i}(s)} = \dfrac{K_{\Gamma Ki}^{R^+}(1 - M\mathbf{e}^{-s\rho_i})}{1 - M\mathbf{e}^{-s(\rho_i - \Delta_i)}}$.

Найти обратное преобразование в общем виде невозможно, поэтому найдем асимптотическое соотношение для $P_{Ki}^+(t)$, применяя Тауберову теорему $\lim\limits_{t \to \infty} P_{Ki}^+(t) = \lim\limits_{s \to 0} \tilde{P}_{Ki}^+(s)$.

Тогда $P_{Ki}^+ = \lim\limits_{t \to \infty} P_{Ki}^+(t) = K_{\Gamma Ki}^{R^+} M\rho_i/(M\rho_i +$ $+ M\Delta_i) = K_{\Gamma Ki}^+$. Аналогично вычисляется вероятность исправного функционирования и для сервера, но для него $K_{\Gamma Si}^{R^+} = M(\omega \wedge \varphi_3)/M\sigma_1'$, $P_{Si}^+ =$ $= K_{\Gamma Si}^{R^+} M\rho_i/(M\rho_i + M\Delta_i) = K_{\Gamma Si}^+$. Так же можно вычислить и вероятность исправного функционирования порта. Опуская вычисления, сразу запишем $K_{\Gamma Pi}^{R^+} = M\varphi_2/M(\varphi_2 + \psi_2)$, $P_{Pi}^+ = K_{\Gamma Pi}^{R^+} M\rho_i/(M\rho_i +$ $+ M\Delta_i) = K_{\Gamma Pi}^+$. Кроме того, можно рассматривать вероятность того, что элемент сети находится в состоянии останова $P_i^- = K_{\Gamma i}^- = P_i - P_i^+$.

Рассмотрим, к примеру, переход простейшей сети с топологией "звезда" в состояние аварии. Поскольку сеть переходит в состояние аварии, если произошла потеря данных хотя бы на одном из

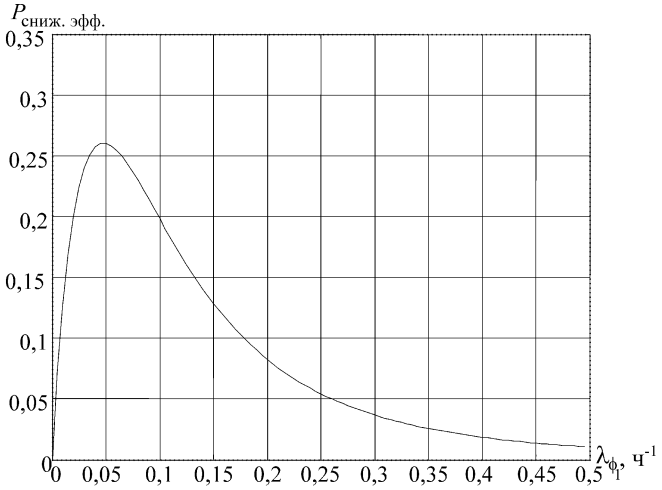

Рис. 1. Зависимость $P_{\text{сниж. эфф}}$ от интенсивности отказов СПД клиента

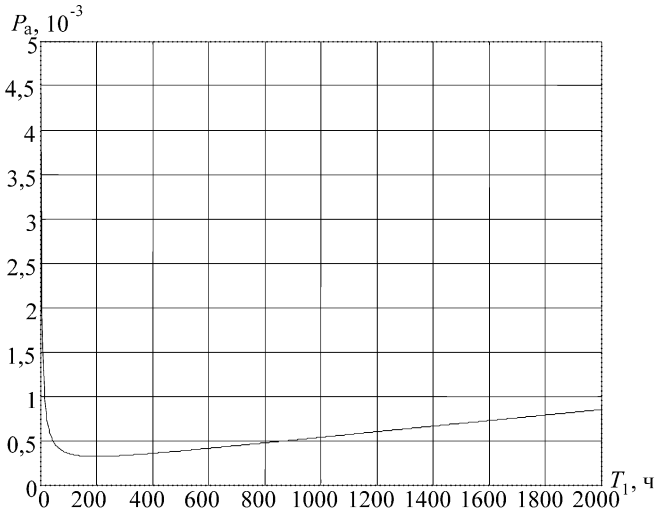

Рис. 2. Зависимость P_a от периода профилактики СБ клиента

клиентов или портов концентратора или на сервере, то вероятность аварии будет равна

$$P_{\text{а}} = 1 - K_{\text{г}}^{S} \prod_{i=1}^{N+1} K_{\text{г}}^{P_i} \prod_{i=1}^{N} K_{\text{г}}^{K_i},$$

где N — число клиентов в сети; $K_{\text{г}}^{K_i}$ — коэффициент готовности i-го клиента; $K_{\text{г}}^{P_i}$ — коэффициент готовности i-го порта концентратора; $K_{\text{г}}^{S}$ — коэффициент готовности сервера. Очевидно, вероятность исправного функционирования всей сети тогда можно записать в виде $P_{\text{ВБР}} = K_{\text{г}S}^{+} \prod_{i=1}^{N+1} K_{\text{г}P_i}^{+} \prod_{i=1}^{N} K_{\text{г}Ki}^{+}$. Тогда вероятность того, что сеть находится в некотором промежуточном состоянии, т. е. работает со сниженной эффективностью, равна $1 - P_{\text{а}} - P_{\text{ВБР}}$. Это промежуточное состояние включает в себя большое число разнообразных вариантов снижения эффективности работы сети, вероятности которых также можно найти, например, вероятность полного останова $P_{\text{останов}} = K_{\text{г}S}^{-} \prod_{i=1}^{N+1} K_{\text{г}P_i}^{-} \prod_{i=1}^{N} K_{\text{г}Ki}^{-}$.

Рассмотрим теперь численный пример. Возьмем топологию "звезда" с 16 клиентами и одним сервером. Пусть $\lambda_\chi = 10^{-6}$ ч$^{-1}$, $\lambda_\gamma = 10^{-1}$ ч$^{-1}$, $\lambda_{\varphi_1} = 10^{-5}$ ч$^{-1}$, $\lambda_{\varphi_2} = 10^{-5}$ ч$^{-1}$, $\lambda_{\varphi_3} = 10^{-5}$ ч$^{-1}$, $\lambda_{\psi_1} = 1$ ч$^{-1}$, $\lambda_{\psi_2} = 1$ ч$^{-1}$, $\lambda_{\psi_3} = 1$ ч$^{-1}$, $\lambda_\omega = 5 \cdot 10^{-6}$ ч$^{-1}$, $\lambda_\varepsilon = 5 \cdot 10^{-1}$ ч$^{-1}$, $\lambda_{\xi_1} = 10^{-4}$ ч$^{-1}$, $\lambda_{\xi_2} = 10^{-4}$ ч$^{-1}$, $\lambda_{\xi_3} = 10^{-4}$ ч$^{-1}$, $\lambda_{\eta_1} = 1$ ч$^{-1}$, $\lambda_{\eta_2} = 1$ ч$^{-1}$, $\lambda_{\eta_3} = 1$ ч$^{-1}$, $T_1 = 100$ ч, $T_2 = 100$ ч, $T_3 = 100$ ч, $\theta_1 = 2$ ч, $\theta_2 = 2$ ч, $\theta_3 = 2$ ч, $M\Delta^K = 40$ ч, $M\Delta^P = 40$ ч, $M\Delta^S = 40$ ч.

На рис. 1 и 2 приведены некоторые графики зависимостей.

Заключение

Таким образом, разработана математическая модель надежности ЛВС произвольной структуры с восстанавливаемыми элементами. Получены выражения для расчета вероятности различных состояний сети при самых общих предположениях о законах распределения случайных величин, например, для вероятности того, что в сети произошла потеря данных, а также для вероятности того, что вся сеть исправно функционирует и для вероятности того, что часть элементов сети восстанавливается и сеть работает со сниженной эффективностью.

Список литературы

1. **Перегуда А. И.** Надежность и безопасность. Модели, показатели и методы их вычисления. Обнинск: ИАТЭ, 2005. 208 с.

2. **Перегуда А. И., Твердохлебов Р. Е.** Математическая модель надежности локальной вычислительной сети клиент-серверной архитектуры с произвольной структурой // Информационные технологии. 2007. № 1. С. 7—15.

3. **Богатырев В. А.** Надежность и эффективность резервированных компьютерных сетей // Информационные технологии. 2006. № 9. С. 25—30.

4. **Байхельт Ф., Франкен П.** Надежность и техническое обслуживание. Математический подход: Пер. с нем. М.: Радио и связь, 1988. 392 с.

УДК 004.315

А. И. Грушин, канд. техн. наук,
ведущий научн. сотр.,
М. Л. Ремизов, инженер-конструктор,
ИТМ и ВТ РАН им. С. А. Лебедева

Нормализующая часть быстродействующего устройства сложения чисел с плавающей запятой

Предлагается алгоритм работы и структура нормализующей части устройства сложения чисел с плавающей запятой, позволяющие выполнить в аппаратуре требования стандарта ANSI/IEEE Standard No. 754 в части режима постепенного отрицательного переполнения без увеличения времени выполнения команды с небольшими дополнительными затратами оборудования.

Ключевые слова: *быстродействующее устройство сложения чисел с плавающей запятой, предсказание величины сдвига, стандарт на двоичную арифметику с плавающей запятой IEEE 754, быстрая нормализация.*

Современные быстродействующие устройства сложения чисел с плавающей запятой обычно состоят из двух частей, работающих параллельно. Одна часть работает в случае эффективного вычитания при разности порядков $РП = |EA - EB|$, равной 0 или 1, другая часть работает при большой разности порядков ($РП > 1$) при эффективном вычитании и во всех случаях эффективного сложения [1]. *Эффективным вычитанием* называется вычитание чисел одинакового знака или сложение чисел разных знаков. *Эффективное сложение* — это сложение чисел одинакового знака или вычитание чисел разных знаков. Этот алгоритм учитывает, что большой сдвиг при выравнивании порядков и большой сдвиг при нормализации результата взаимно исключают друг друга, определение сдвига для нормализации выполняется одновременно со сложением мантисс, округление проводится одновременно со сложением мантисс.

Такое устройство содержит один полноразрядный сдвигатель и один полноразрядный сумматор на критическом пути как в случае большой, так и в случае малой разности порядков. Это позволяет уменьшить время выполнения операции по сравнению с устройством сложения, имеющим обычную структуру [2].

Известно много вариантов реализации устройства сложения чисел с плавающей запятой с делением на две части. Устройство, описанное в [3 и 4], состоит из двух частей (рис. 1), в каждой из них проводится сложение. Первая (округляющая) часть используется в трех случаях:

- когда в устройстве выполняется эффективное сложение;
- когда выполняется эффективное вычитание, разность порядков $РП = 1$ и не требуется нормализация;
- когда выполняется эффективное вычитание и $РП > 1$.

В остальных случаях (эффективное вычитание и $РП = 0$, либо $РП = 1$ и требуется нормализация) работает вторая (нормализующая) часть устройства. В округляющей части (ОКР) по результату вычитания порядков происходит сдвиг вправо на величину $РП$ мантиссы числа с меньшим порядком (MA или MB) и ее обращение в случае эффективного вычитания. Это обеспечивает получение на сумматоре положительного результата, что делает ненужной схему обращения суммы. Затем мантиссы поступают в сумматор, где одновременно с суммированием проводится округление.

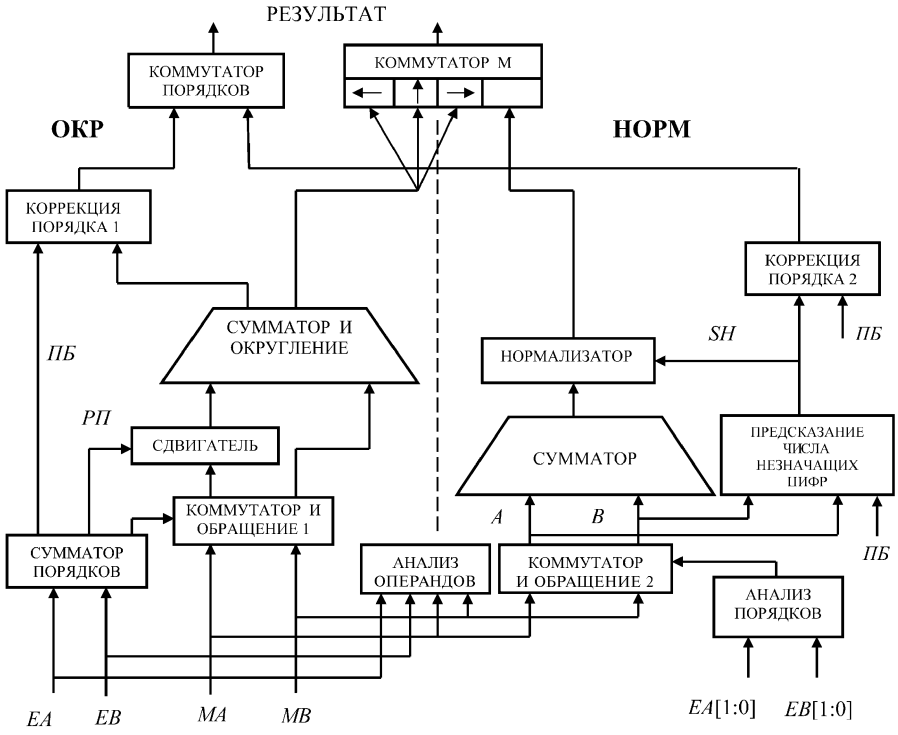

Рис. 1. Структура устройства сложения чисел с плавающей запятой, состоящего из округляющей и нормализующей частей

В нормализующей части (НОРМ) устройства, где $РП \leqslant 1$, по двум младшим разрядам порядков определяется, какой из порядков больше; в случае неравных порядков на один разряд вправо сдвигается мантисса числа с меньшим порядком. Одновременно с суммированием мантисс предсказывается число старших незначащих цифр суммы. В этой части устройства возможна ситуация, когда на выходе сумматора получается число, у которого в старших разрядах большое число незначащих цифр — нулей в случае положительной суммы или единиц в случае отрицательной суммы. В случае отрицательной суммы ее надо обратить. Сумму нужно сдвинуть влево так, чтобы старшая значащая единица попала в позицию скрытого бита (целая часть мантиссы, которая не присутствует в явном виде в представлении числа). При этом предварительный порядок результата, равный порядку большего числа *ПБ*, уменьшается на величину, равную числу старших незначащих разрядов мантиссы *SH*. Согласно стандарту ANSI/IEEE 754 на двоичную арифметику с плавающей запятой [5], порядок результата должен быть в интервале $[E_{\min}, E_{\max}]$, где $E_{\min} = 00...01$, $E_{\max} = 11...10$. Таким образом, мантисса не должна сдвигаться влево на число разрядов, превосходящее максимально допустимую величину *M*, равную *ПБ*-1. Величина *M* называется *кодом ограничения*. Если код ограничения меньше кода сдвига *SH*, то мантисса сдвигается влево только на *M* разрядов, и результат будет денормализованным числом. Тем самым обеспечивается режим постепенного отрицательного переполнения (*gradual underflow*).

На коммутаторе результата на основании анализа разности порядков выбирается результат округляющей или нормализующей части устройства.

Устройство [3, 4] по сравнению с [1] имеет следующие преимущества:

- округление проводится только в округляющей части устройства;
- предсказанный сдвиг для нормализации не требует коррекции;
- осуществлена аппаратная реализация требований стандарта ANSI/IEEE Standard 754 [5] в части обработки денормализованных операндов, бесконечностей и NaN'ов;
- учитывается ограничение на сдвиг для нормализации, что позволяет поддержать в аппаратуре режим постепенного отрицательного переполнения (*gradual underflow*).

Недостатком устройства [4] являются большие аппаратные затраты, необходимые для предсказания кода сдвига, причем это в значительной мере связано с предсказанием младших разрядов кода сдвига.

В [6] рассматриваются несколько вариантов формирования кода сдвига, когда старшие разряды кода сдвига SH_{ms} предсказываются, а младшие разряды SH_{ls} определяются по сумме или по частично нормализованной сумме, т. е. сумме, которая сдвинута влево на величину, определяемую предсказанными старшими разрядами кода сдвига (рис. 2). Во всех случаях код сдвига для нормализации предсказывается и определяется точно, а ограничение учитывается без введения дополнительных логических уровней. При этом уменьшено количество оборудования по сравнению с [4] за счет экономии при определении младших разрядов кода сдвига и получены приемлемые временные характеристики. Анализ экспериментальных результатов показал, что критический путь проходит через цепи предсказания разрядов сдвига, а суммирование и анализ разрядов суммы менее критичны.

Пусть в нормализующей части устройства сложения чисел с плавающей запятой *A* и *B* — это мантиссы, поступающие в сумматор для суммирования и одновременно в схему предсказания кода сдвига для нормализации (мантисса меньшего числа после сдвига и обращения и мантисса большего числа), при этом старший разряд имеет номер 1. В [7] для каждого разряда вводится функция

$$E_i = \overline{A_i \oplus B_i} \bullet (A_{i+1} + B_{i+1}).$$

Ее значение зависит от текущего разряда *i* и от более младшего разряда $i + 1$. Функция E_i определяет положение старшей значащей единицы суммы, а именно, самая старшая $E_i = 1$ находится

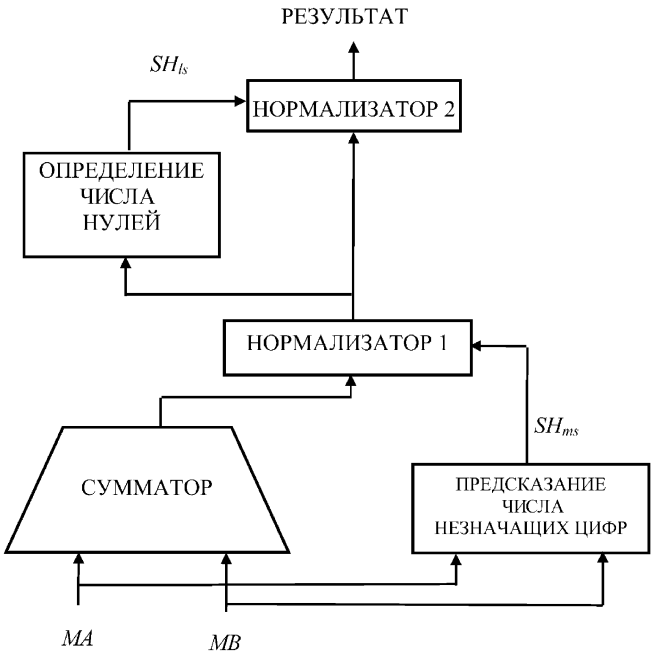

Рис. 2. Предсказание и определение числа старших незначащих цифр по частично нормализованной сумме

в том же разряде (или на один разряд левее), что и первая единица после строки нулей в сумме. Этот способ предсказания применим, если всегда мантисса меньшего числа вычитается из мантиссы большего числа. В этом случае сумма будет положительным числом (если мантиссы равны, то она равна нулю, но тогда нормализации не требуется), следовательно, нужно предсказать только положение старшей единицы, что позволяет упростить формулы для предсказания старшей значащей цифры.

Обозначим

$$N_{j-k} = \overline{E}_j \bullet \overline{E}_{j+1} \bullet \ldots \bullet \overline{E}_{k-1} \bullet \overline{E}_k. \qquad (1)$$

Если $N_{1-k} = 1$, то k или $k + 1$ старших разрядов суммы равны 0. Тогда $N_{1-31} = 1$ означает, что 31 или 32 старших разряда суммы равны 0.

В [4] предложен способ учета кода ограничения $M\{M_5, M_4, M_3, M_2, M_1, M_0\}$ сдвига в случае точного предсказания. Он применим и к формуле (1), которая предсказывает неточно. Для мантиссы формата double, у которой номер старшего разряда равен 1 (это есть так называемый скрытый бит), а номер младшего разряда равен 53, получим следующие выражения для сигналов сдвига на 32, 16, 8 и 4 разряда:

$$SH32 = N_{1-31} \bullet M_5; \qquad (2)$$

$$SH16 = \overline{SH32} \bullet N_{1-15} \bullet (M_5 + M_4) + N_{1-47} \bullet M_4; \qquad (3)$$

$$SH8 = \overline{SH32} \bullet \overline{SH16} \bullet N_{1-7} \bullet (M_5 + M_4 + M_3) + \overline{SH16} \bullet N_{1-39} \bullet (M_4 + M_3) + \overline{SH32} \bullet N_{1-23} \bullet (M_5 + M_3); \qquad (4)$$

$$SH4 = \overline{SH32} \bullet \overline{SH16} \bullet \overline{SH8} \bullet N_{1-3} \bullet (M_5 + M_4 + M_3 + M_2) + \overline{SH32} \bullet \overline{SH16} \bullet N_{1-11} \bullet (M_5 + M_4 + M_2) + \overline{SH32} \bullet \overline{SH8} \bullet N_{1-19} \bullet (M_5 + M_3 + M_2) + \overline{SH32} \bullet N_{1-27} \bullet (M_5 + M_2) + \overline{SH16} \bullet \overline{SH8} \bullet N_{1-35} \bullet (M_4 + M_3 + M_2) + \overline{SH16} \bullet N_{1-43} \bullet (M_4 + M_2) + \overline{SH8} \bullet N_{1-51} \bullet (M_3 + M_2). \qquad (5)$$

Формулы (2)—(5), предсказывающие сигналы сдвига для управления нормализатором, предсказывают сдвиг, определяемый совокупностью четырех сигналов сдвига, с возможной ошибкой, равной 1. Если предсказание неточно, то ошибка заключается в том, что вырабатывается сдвиг, кратный 4, а точный сдвиг должен быть на 1 меньше (например, см. табл. 1).

Таблица 1

Пример неточного предсказания сдвига

Название сигнала	Предсказанное значение сдвига	Точное значение сдвига
*SH*32	0	0
*SH*16	1	0
*SH*8	0	1
*SH*4	0	1

Чтобы получить точные сигналы сдвига, в каждой из функций N_{1-k} нужно добавить еще один сомножитель (например, в N_{1-31} добавить $\overline{E}_{32}$), также необходимо учесть переносы, возникающие в сумматоре, в 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48 и 52-й разряды. Однако это значительно усложняет формулы (2)—(5) и увеличивает критический путь, поэтому принято решение расширить НОРМАЛИЗАТОР 1 слева на один разряд и в случае необходимости корректировать ошибку предсказания сдвига посредством сдвига мантиссы в узле НОРМАЛИЗАТОР 2 вправо на один разряд.

На рис. 3 показано формирование *i*-го разряда в узле НОРМАЛИЗАТОР 2. Коммутатор с двух направлений MUX2, которым управляет сигнал сдвига на два разряда *SH2*, выбирает разряды $S4_i$ или $S4_{i+2}$, поступающие с выхода узла НОРМАЛИЗАТОР 1. Далее на элементе 2И-3ИЛИ происходит сдвиг на один разряд влево по сигналу *SH1*, прохождение без сдвига по сигналу *NSH1* или сдвиг вправо на один разряд (коррекция) по сигналу *Kop*. При коррекции проходит разряд $S4_{i-1}$, поступающий с выхода узла НОРМАЛИЗАТОР 1, т. е. в этом случае нормализатор работает быстрее, чем при точном предсказании.

Сигнал коррекции *Kop* и предварительные (без учета ограничения) сигналы сдвига $SH2_p$ и $SH1_p$ вырабатываются по результату анализа четырех старших разрядов, выходящих с узла НОРМАЛИЗАТОР 2, на основании табл. 2, где знаком "x" обозначены значения входных сигналов, которые не влияют на значения функций.

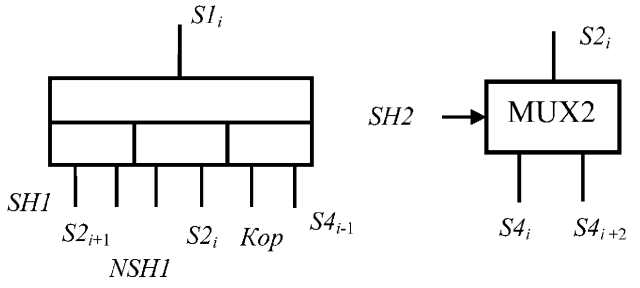

Рис. 3. Формирование *i*-го разряда в узле Нормализатор 2

Таблица 2

Таблица истинности для формирования сигналов $SH2_p$, $SH1_p$, *Кор*

$S4_0$	$S4_1$	$S4_2$	$S4_3$	$S4_4$	$SH2_p$	$SH1_p$	*Кор*
0	1	×	×	×	0	0	0
0	0	1	×	×	0	1	0
0	0	0	1	×	1	0	0
0	0	0	0	1	1	1	0
0	0	0	0	0	0	1	0
1	×	×	×	×	×	0	1

С учетом ограничения кода сдвига формулы для сигналов сдвига имеют следующий вид:

$$SH2 = (M_5 \bullet \overline{SH32} + M_4 \bullet \overline{SH16} + M_3 \bullet \overline{SH8} + M_2 \bullet \overline{SH4} + M_1) \bullet SH2_p;$$

$$SH1 = (M_5 \bullet \overline{SH32} + M_4 \bullet \overline{SH16} + M_3 \bullet \overline{SH8} + M_2 \bullet \overline{SH4} + M_1 \bullet \overline{SH2} + M_0) \bullet SH1_p;$$

$$NSH1 = \overline{SH1} \bullet \overline{Kop}.$$

Предсказание кода сдвига можно ускорить, если анализировать не слагаемые на входе сумматора мантисс, а исходные операнды [8].

В нормализующей части устройства сложения (рис. 4) на элементе XNOR сравниваются младшие разряды порядков EA_0 и EB_0. Если эти разряды совпадают, вырабатывается сигнал "порядки равны" *ПР*, который управляет узлами КОММУТАТОР 1 и КОММУТАТОР 2. Узел КОММУТАТОР 1 пропускает на инверсный выход мантиссу числа *МА* или {0, *МА*[1:52]}, т. е., смещенную на один разряд вправо мантиссу *МА*. КОММУТАТОР 2 делает то же самое с мантиссой *МВ*. Тем самым на выходах этих коммутаторов в случае равных порядков (*ПР* = 1) оказываются инверсии мантисс *МА* и *МВ* соответственно; если *ПР* = 0, через коммутаторы пройдут инверсии сдвинутых вправо на один разряд мантисс.

В схеме СРАВНЕНИЕ определяется, какая из исходных мантисс больше, для этого выполняется вычитание *МА* − *МВ*, определяется перенос из старшего разряда, но сумма не формируется. Сигнал *МАБ* говорит, что *МА* ⩾ *МВ*.

КОММУТАТОР 3 и КОММУТАТОР 4 выбирают слагаемые *А* и *В* для сумматора, ими управляет сигнал *ОбрА*, который равен 1, если надо обратить мантиссу *МА*. Если *ОбрА* = 0, то обращается *МВ*. Сумма *SM* ⩾ 0, поэтому на выходе сумматора не нужна схема обращения.

ОбрА вырабатывается на основании анализа двух младших разрядов порядков, а если порядки равны, то учитывается результат сравнения мантисс *МАБ*.

Сигнал *СдА* = 1 (на рисунке не показан) говорит о том, что надо сдвигать мантиссу *МА*, если *СдА* = 0, то надо сдвигать *МВ*:

$$СдА = \overline{EA_1} \bullet \overline{EA_0} \bullet \overline{EB_1} + \overline{EA_1} \bullet EA_0 \bullet EB_1 + \overline{EA_0} \bullet EA_1 \bullet EB_1 + EA_1 \bullet EA_0 \bullet \overline{EB_1},$$

$$ОбрА = СдА \bullet (-ПР) + ПР \bullet (-МАБ).$$

Функции *NA* вырабатываются по формуле (1) в предположении, что инвертировалась мантисса *МА*, а функции *NB* — в предположении, что инвертировалась *МВ*. Эти функции формируются для групп по четыре разряда. Затем в узле КОММУТАТОР 5 выбирается нужный набор функций, по которому с помощью формул (2)—(5) предсказывается код сдвига на 32, 16, 8 и 4 разряда. Критическим по времени является сигнал *ОбрА*, который уменьшает в 2 раза число функций *N*.

Порядок результата корректируется по алгоритму, описанному в [9].

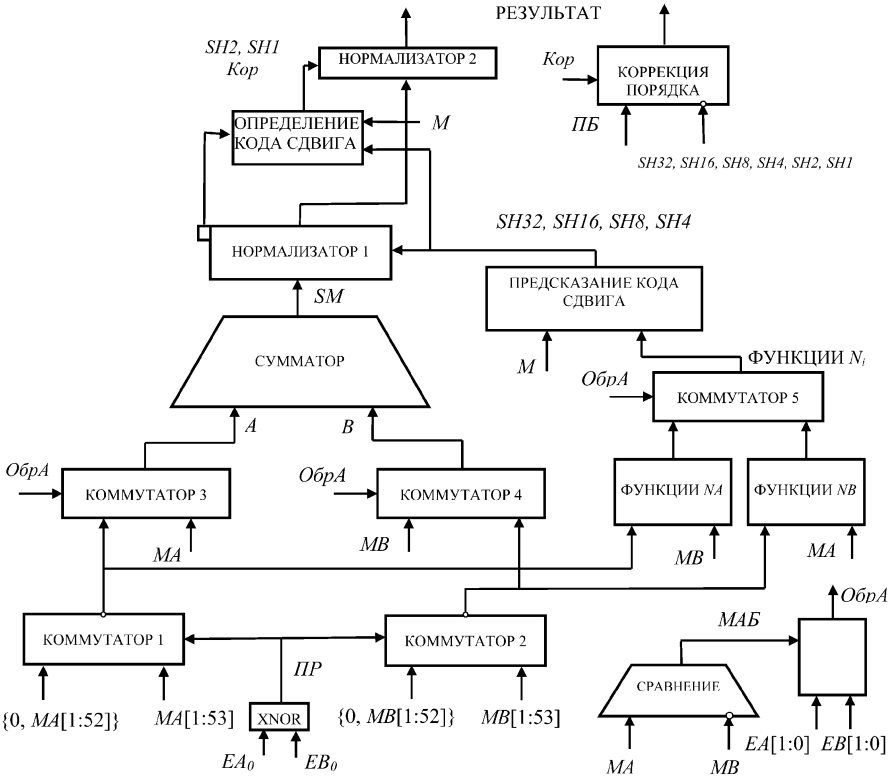

Рис. 4. Предлагаемая структура нормализующей части устройства сложения чисел с плавающей запятой

Заключение

В статье предложен алгоритм предсказания и определения сдвига для нормализации результата сложения чисел с плавающей запятой с учетом ограничения сдвига. Для увеличения скорости и экономии оборудования он упрощен по сравнению с точным алгоритмом, поэтому предсказание может давать ошибку на один разряд. Предложен метод коррекции неточного предсказания, который позволяет получить правильный результат без введения дополнительных логических уровней. Это позволяет реализовать режим постепенного отрицательного переполнения, который предусмотрен стандартом ANSI/IEEE Standard No. 754, в аппаратуре с меньшими временными и аппаратными затратами, чем известные методы.

Описана структура нормализующей части устройства сложения чисел с плавающей запятой, реализующая алгоритм предсказания и определения сдвига с учетом ограничения, позволяющая анализировать исходные операнды, а не слагаемые на входе сумматора, что позволило еще ускорить работу нормализующей части устройства сложения. Экспериментальные оценки показали, что критическая цепь проходит через сумматор, а не через цепи предсказания кода сдвига.

Список литературы

1. **Quach N., Flynn M.** An Improved Algorithm for High-Speed Floating-Point Addition // Tech. Rep. CSL-TR-90-442, Stanford University, August 1990.

2. **Ercegovac M., Lang T.** Digital Arithmetic. San Francisco: Morgan Kaufmann Publishers, 2004. 421 p.

3. **Gorshtein V. Y., Grushin A. I., Shevtsov S. R.** Floating point addition methods and apparatus, U. S. patent 5808926, 9/1998. P. 1—22.

4. **Grushin A. I., Vlasenko E. S.** Computer methods and apparatus for eliminating leading non-significant digits in floating point computations, U. S. patent 5732007, 3/1998. P. 1—34.

5. **IEEE** Standard for Binary Floating-Point Arithmetic, ANSI/IEEE Standard No. 754, American National Standards Institute, Washington, DC, 1985.

6. **Грушин А. И., Ремизов М. Л.** Определение и частичное предсказание кода сдвига для нормализации результата сложения чисел с плавающей запятой // Сб. научн. тр. Института микропроцессорных вычислительных систем РАН. М.: ИМВС РАН, 2003. С. 9—17.

7. **Suzuki H., Morinaka H., Makino H., Nakase Y., Mashiko K., Sumi T.** Leading-Zero Anticipatory Logic for High-Speed Floating Point Addition // IEEE Journal of Solid-State Circuits. August 1996. Vol. 31. N 8.

8. **Wolrich G., Fischer T., Ellis J.** Normalization shift prediction independent of operand subtraction, U. S. patent 5867407, 2/1999.

9. **Грушин А. И., Ремизов М. Л.** Алгоритм быстрой коррекции порядка результата в устройстве сложения // Сб. научн. тр. Института микропроцессорных вычислительных систем РАН. М.: ИМВС РАН, 2002. С. 91—95.

УДК 004.7

С. А. Зинкин, канд. техн. наук, доц., Пензенский государственный университет

Элементы новой объектно-ориентированной технологии для моделирования и реализации систем и сетей хранения и обработки данных

Предлагаются модели и методы, которые могут быть использованы в качестве основы создания новой объектно-ориентированной сетевой технологии моделирования и проектирования распределенных систем хранения и обработки данных на основе согласованных взаимодействий объектов через общее пространство — коммуникационную среду или общее пространство информационных объектов.

Ключевые слова: *системы хранения и обработки данных, объектно-ориентированное моделирование, сетевая информационная технология, согласованные взаимодействия объектов и процессов, коммуникационная среда, пространство информационных объектов.*

Введение

Сетевые представления дискретных динамических систем могут быть положены в основу ряда технологий системного моделирования и распределенного сетевого программирования. Сетевые формализмы также хорошо согласуются с парадигмой согласования объектов и процессов [1].

В работах [2] и [3] были предложены формализмы сетей абстрактных машин (СеАМ) нескольких видов, которые могут быть положены в основу построения новых инструментальных средств для распределенного моделирования и программирования, базирующихся на концепции непосредственно интерпретируемых спецификаций. Последнее означает, что описания моделей "непосредственно" программируются в терминах выражений и операторов некоторого языка или реализуются комплексом специализированных

программно-аппаратных средств. При реализации сетевой технологии, базирующейся на формализме СеАМ, требуется размещение в операционной среде вычислительной сети данных, структурированных как объекты некоторого FS-пространства (от *Function Spaces*).

Концептуально формализм СеАМ берет свое начало от эволюционирующих алгебраических систем [4] и машин Колмогорова [5]. Для построения сетей абстрактных машин в работах [2] и [3] была предложена алгебра СеАМ. Использование основных идей из алгебры алгоритмов Глушкова [6], например суперпозиций темпоральных и дополнительных операторов, позволяет естественным образом проектировать сложные иерархические расширенные сети абстрактных машин (ИРСеАМ). В этой связи выделяются два класса сетей абстрактных машин. В относительно простых сетях первого класса подсети, или модули РСеАМ, начиная с тривиальных подсетей — элементарных обновлений интерпретации текущей сигнатуры, составляющих систему образующих, участвуют в операциях однократно. В сложных сетях второго класса допускается многократное участие указанных подсетей — модулей РСеАМ в произвольных суперпозициях подсетей. В обоих случаях формируются иерархические сети.

В дальнейшем общая аббревиатура СеАМ используется для обозначения технологии или реализации сетей, если это не противоречит контексту; там же, где следует различать конкретные виды выражений, описывающих сети абстрактных машин, используются аббревиатуры СеАМ, РСеАМ и ИРСеАМ.

Реализация сетей абстрактных машин

Основные действия в сети реализуются агентами, которые могут перемещаться, клонироваться, уничтожаться, обмениваться сообщениями, а главное, модифицировать FS-пространство в зависимости от внешних и внутренних условий. Агент — это мобильный или стационарный объект, либо экземпляр некоторого класса, реализующий "методы". Методы формируются на основе спецификаций СеАМ, называемых "выражениями для модулей", или просто модулями. Точнее, модуль — это двойка $<a_i, m_i>$, где a_i — агент, а m_i — выражение на языке СеАМ, интерпретируемое агентом. Концепция, связывающая агентно-ориентированную технологию аглетов [7] и СеАМ-технологию, была предложена в работе [8]. Логическая, физическая и интегрированная физическая модели сетей хранения и обработки данных представлены на рис. 1.

Логическая модель (рис. 1, *а*) включает сеть абстрактных машин (СеАМ) и абстрактную структуру памяти, состоящую из множества функций и предикатов. В процессе функционирования модули СеАМ модифицируют информационные объекты, представляющие в памяти функции и предикаты. Физическая модель (рис. 1, *б*) базируется на представлении распределенной системы в виде двух взаимодействующих сетей — управляющей сети и операционной сети. Агенты, "работающие" на FS-пространстве, делятся на два типа — агенты-серверы и агенты-демоны. Агенты-серверы интерпретируют выражения СеАМ, а агенты-демоны реализуют рабочие процедуры операционной среды. Агенты-серверы выполняют свои действия в управляющей сети, а агенты-демоны — в операционной. Агенты различных видов согласуют свои действия через разделяемое FS-пространство или путем обмена сообщениями. Интегрированная физическая модель (рис. 1, *в*) допускает свободное размещение или перемещение агентов различных видов в обеих сетях.

Рис. 1. Логическая (*а*), физическая (*б*) и интегрированная физическая (*в*) структуры сети хранения и обработки данных

Организация взаимодействия модулей в сети

Реализуемые методами агентов модули разделяют общее (глобальное) FS-пространство или Global FS — GFS (рис. 2). В процессе выполнения каждый модуль m_i, $i = 1, 2, ..., n$, проверяет и модифицирует функции и предикаты, составляющие его локальное пространство Local FS_i — LFS_i. Это пространство является частью общего пространства GFS и локализуется (блокируется) только на время работы модуля (точнее, работы агента-сервера, интерпретирующего данный модуль). Локализация заключается в блокировке

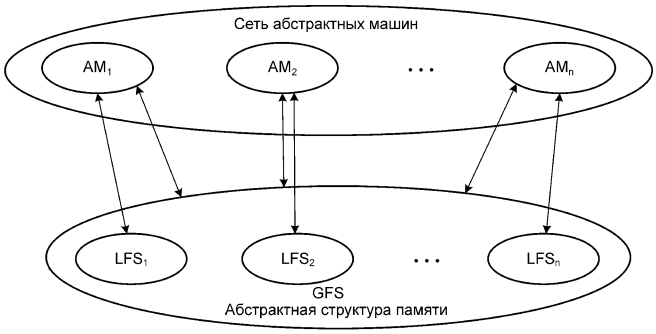

Рис. 2. Структурированная память

проверяемых и модифицируемых функций и предикатов, временно предотвращающей доступ к данным функциям других модулей. При выполнении некоторых модулей ряд предикатов и функций, с которыми имеет дело модуль, может и не подвергаться блокировке: например, предикат или функция могут только проверяться и не модифицироваться в дальнейшем или может быть заведомо известно, что модификации будут подвергнуты непересекающиеся фрагменты некоторого отношения.

Схемы взаимодействия модулей представлены на рис. 3 и рис. 4. В первом случае (рис. 3) взаимодействие модулей может осуществляться через две среды — коммуникационную среду агентов (AS — Agent Space) и среду хранения управляющей информации и данных — FS-пространство. В рамках FS-технологии AS-пространство играет вспомогательную роль: через него агенты могут обмениваться сообщениями или перемещаться по нему, создавая операционную среду. Например, агент-передатчик может сообщить агенту-приемнику о том, что он уже модифицировал FS-пространство и результатами данной модификации может воспользоваться данный агент-приемник. Тем самым обеспечивается ускоренная реакция агента-приемника на модификацию FS-пространства, т. е. на изменение состояния структурированной памяти.

Рис. 3. Реализация межмодульных причинно-следственных связей через разделенные AS- и FS-пространства

Рис. 4. Реализация межмодульных причинно-следственных связей через интегрированное AFS-пространство

Во втором случае (рис. 4) межмодульные причинно-следственные связи реализуются через интегрированную среду хранения и передачи данных и агентов (AFS-пространство — Agent-Function Space). Например, здесь легко реализуется режим, при котором "короткие" отношения, содержащие небольшое число кортежей, передаются с помощью сообщений, а к "длинным" отношениям просто обеспечивается доступ после уведомления. В данной среде агенты и сами могут перемещаться, перенося с узла на узел не только свой код, но и данные.

Организация сетевой операционной среды

Концептуальное представление о сетевой технологии хранения и обработки данных дают рис. 5 и рис. 6. На рис. 7 дано графическое изображение объектов FS-технологии. Данные рисунки иллюстрируют использование двух похожих сред для распределенного моделирования и физической реализации систем хранения и обработки данных. На рис. 5 представлена система, в которой мобильными агентами используются стационарные функции, а на рис. 6 функции сами могут перемещаться к агентам. Используемое пространство функций и предикатов является виртуальным — функции и предикаты могут размещаться на произвольных узлах вычислительной сети, храниться в произвольных базах данных, инкапсулироваться в передаваемые сообщения, храниться и перемещаться вместе с мобильными агентами. Размещением объектов в сети занимаются специальные служебные агенты хранения-размещения.

Алгоритмы функционирования модулей в FS-пространстве

Работу сети, представленной на рис. 5, опишем на примере реализации алгоритма блокировок. На рис. 8 представлена схема алгоритма A_1 интерпретации модуля m_i агентом-сервером a_i, выполняемо-

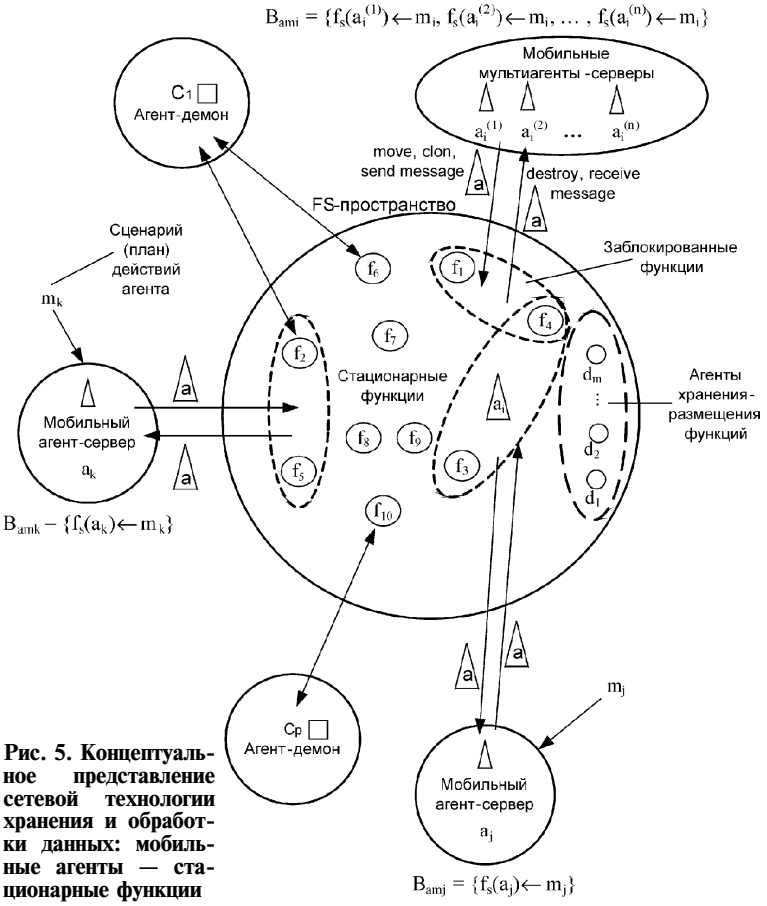

Рис. 5. Концептуальное представление сетевой технологии хранения и обработки данных: мобильные агенты — стационарные функции

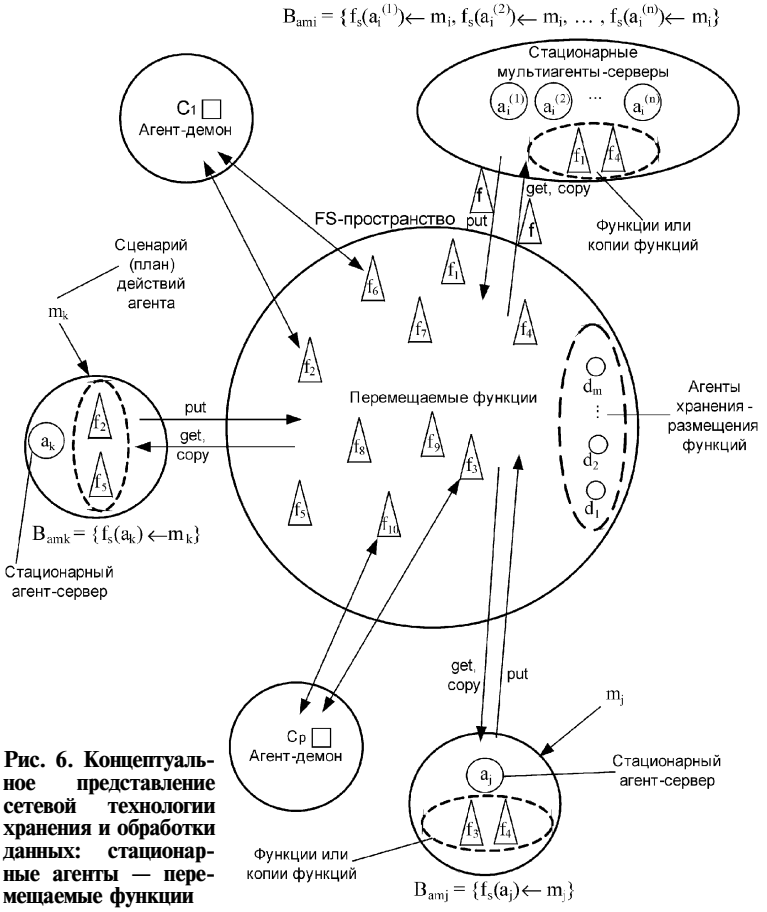

Рис. 6. Концептуальное представление сетевой технологии хранения и обработки данных: стационарные агенты — перемещаемые функции

го непосредственно после обновления стартовой функции $f_{\text{start}}(a_i) \leftarrow m_i$. Описание пунктов этого алгоритма представлено ниже.

1. Агент-сервер (в дальнейшем просто агент) a_i создает свои копии (клоны) и пересылает их на узлы, где хранятся функции и предикаты с именами из множества Σ (точнее, информационные объекты, представляющие данные функции и предикаты в FS-пространстве), проверяемые и обновляемые модулем m_i.

2. Проверка истечения времени моделирования; если это время истекло, то работа агента заканчивается.

3. Агенты-копии посылают сообщения о незаблокированности соответствующих функций и предикатов, а также о выполнении требуемых условий для каждой конкретной функции агенту-родителю a_i. Полагается, что при выполнении данного пункта алгоритма все функции и предикаты открыты для проверки.

4. Агент a_i проверяет условие запуска модуля m_i.

5. Агент a_i проверяет наличие функции f_T в FS-пространстве (функция f_T описана ниже в п. 7).

6. Агент a_i извлекает функцию f_T из FS-пространства.

7. Агент a_i проверяет условие $E_i \cap T = \varnothing$ незаблокированности функций и предикатов из множества E_i, проверяемых в условной части модуля m_i и, возможно, модифицируемых данным модулем, а T — множество заблокированных функций и предикатов. Здесь $E_i \subseteq X$, $T_i \subseteq X$, где X — множество всех функций и предикатов с именами из Σ. В терминах технологии CeAM агент a_i при выполнении данного пункта алгоритма проверяет истинность высказывания

$$f_{\text{isequal}}(f_{\text{inter}}(f_{Ei}, f_T), \text{emptyset}),$$

где f_{Ei} и f_T — характеристические функции множеств E_i и T соответственно, а emptyset — пустое множество.

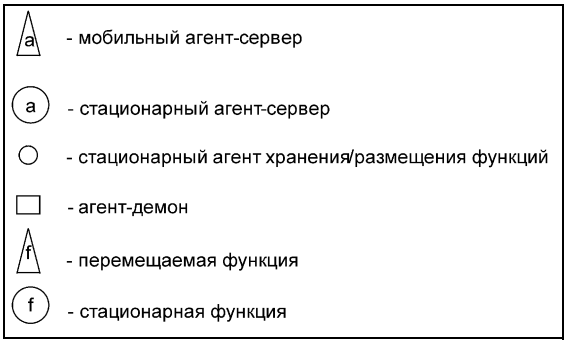

Рис. 7. Объекты FS-технологии

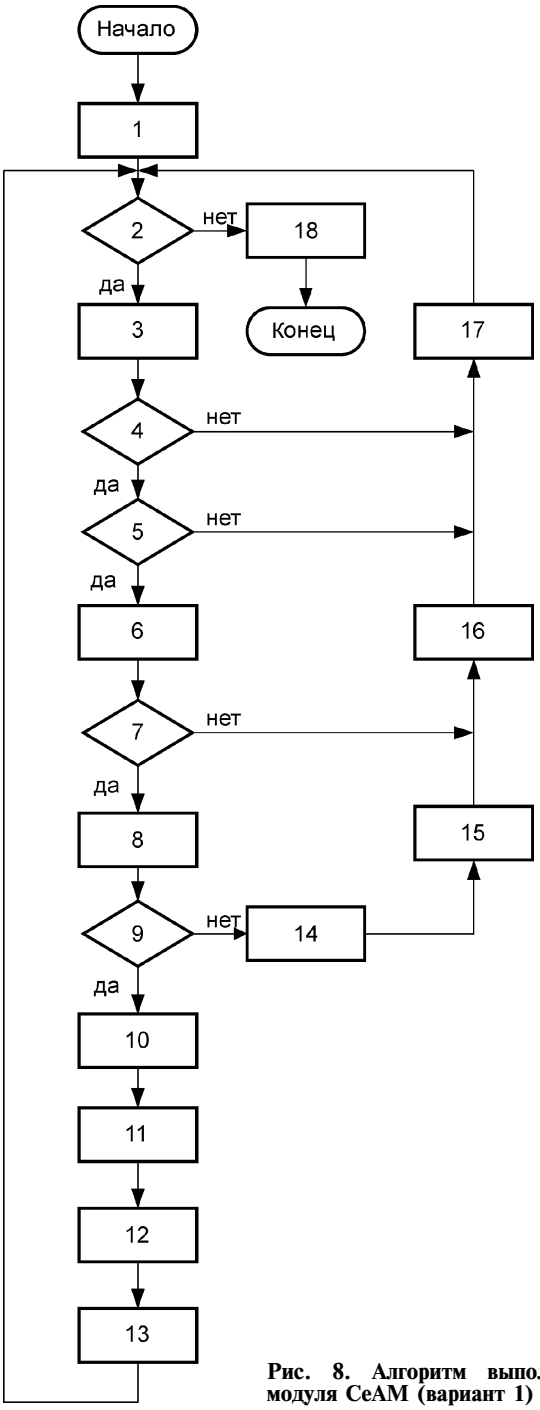

Рис. 8. Алгоритм выполнения модуля СеАМ (вариант 1)

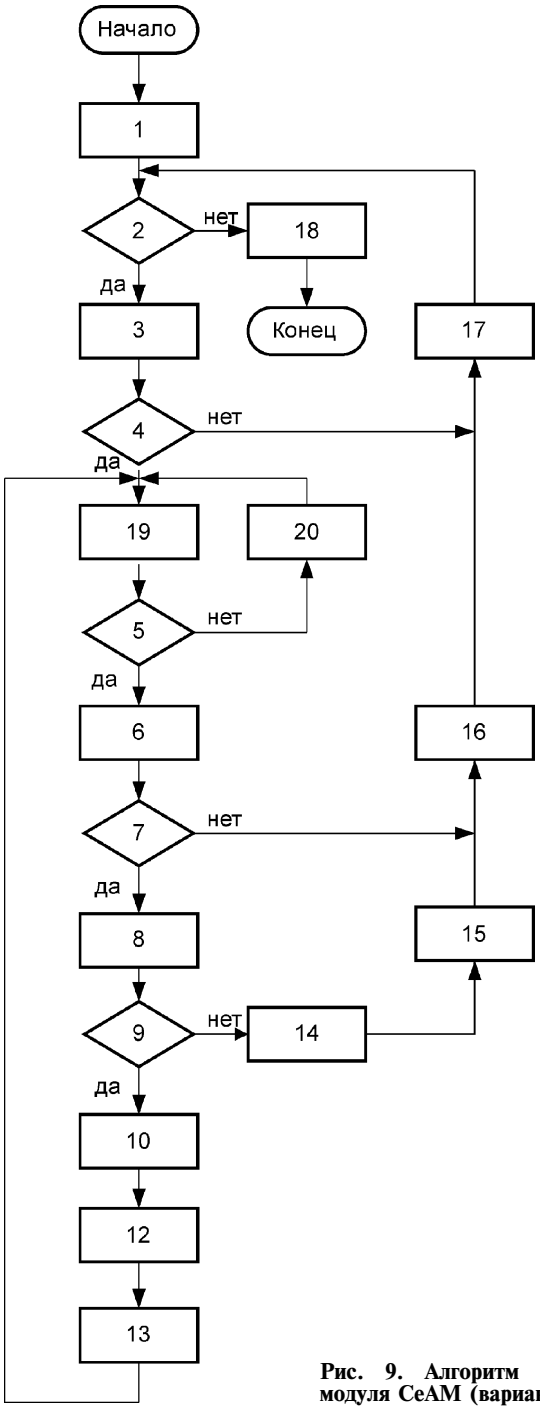

Рис. 9. Алгоритм выполнения модуля СеАМ (вариант 2)

8. Агент a_i посылает агентам-копиям сообщения ("команды") "заблокировать свои функции и предикаты и вернуть сведения о выполнении требуемых условий". После получения всех ответов агент a_i модифицирует множество T: $T \leftarrow T \cup E_i$, добавляя в него вновь заблокированные функции и предикаты. В терминах технологии СеАМ агент a_i при этом реализует правило обновления $f_T \leftarrow f_{\text{union}}(f_T, f_{Ei})$, а затем характеристическая функция f_T множества T возвращается агентом a_i в FS-пространство.

9. Агент a_i проверяет истинность всех условий, необходимых для выполнения модуля m_i и дает разрешение своим агентам-копиям на модификацию заблокированных ими функций и предикатов.

10. Агенты-копии выполняют все необходимые обновления заблокированных функций и предикатов, получив соответствующие сообщения ("команды") от агента-родителя a_i (возможно, таким образом, по инициативе агента a_i на различных узлах выполняется блок обновлений функций и предикатов, принадлежащих общему

FS-пространству). На этом выполнение модуля m_i завершается.

11. Агенты-копии осуществляют разблокирование функций и предикатов, заблокированных ими ранее, и уведомляют об этом родительский агент.

12. Агент a_i модифицирует множество T: $T \leftarrow \leftarrow T \backslash E_i$, удаляя из него заблокированные ранее функции и предикаты. В терминах формализма СеАМ это эквивалентно выполнению агентом a_i правила обновления $f_T \leftarrow f_{\text{diff}}(f_T, f_{Ei})$. Для выполнения указанных действий агент a_i снова выбирает функцию f_T из FS-пространства.

13. Характеристическая функция f_T множества T возвращается агентом a_i в FS-пространство.

14. Если условие, проверяемое в п. 9, ложно, то агент a_i выполняет обновление $f_T \leftarrow f_{\text{diff}}(f_T, f_{Ei})$.

15. Выполняются действия, аналогичные п. 11.

16. Выполняются действия, аналогичные п. 13.

17. Агент a_i ожидает в течение времени тайм-аута τ_i, а затем продолжает выполнение последовательности действий по запуску модуля m_i.

18. Уничтожение агента.

В описанном выше алгоритме функции f_{union}, f_{diff} и f_{inter} реализуют выполнение бинарных операций "объединение", "разность" и "пересечение" реляционной алгебры, определенных над областями истинности характеристических функций соответствующих множеств.

В выражениях для правил обновлений символ "$\leftarrow$" означает обновление функции после выполнения операции реляционной алгебры. В этом контексте символ "$\leftarrow$" является сокращающим и заменяет множество совместимых обновлений характеристической функции.

Другим способом решения данной проблемы является организация очереди агентов к функции f_T (рис. 9). В схему алгоритма A_2 на рис. 9 добавлены вершины 19 (постановка агента в очередь) и 20 (ожидание в течение тайм-аута τ_i). Остальные условные и операторные вершины и их интерпретация соответствуют схеме алгоритма A_1, представленного на рис. 8.

В некоторых случаях, когда несколько агентов должны интерпретировать один и тот же модуль, требование блокировки функций и предикатов может быть ослаблено. Например, агенты по очереди, выполняя последовательность квантифицированных операторов $\tilde{\exists}!$, связывают определенные значения предметных переменных и затем выполняют совместимые блоки обновления функций и предикатов. В таком случае используемые модулем функции и предикаты блокируются только от других агентов, не принадлежащих данному семейству. Агенты этого семейства, которые не смогли выбрать с помощью очередного оператора $\tilde{\exists}!$ значение предметной переменной, освобождают все выбранные ранее переменные и становятся в конец очереди.

Рассмотрим для примера выполнение семейством агентов $\{a_i', a_i'', a_i'''\}$ модуля

$$m_i = [(\tilde{\exists}!\, x \in X) f_1(x)]([(\tilde{\exists}!\, y \in Y) f_2(y)](\{f_1(x) \leftarrow \text{false}, f_1'(x) \leftarrow \text{true},$$

$$f_2(y) \leftarrow \text{false}, f_2'(y) \leftarrow \text{true}, f_3(x, y) \leftarrow \text{true}\} \vee R^E) \vee R^E).$$

Работа агентов указанного семейства инициируется следующим блоком обновления стартовой функции:

$$\{f_{\text{start}}(a_1') \leftarrow m_i, f_{\text{start}}(a_1'') \leftarrow m_i, f_{\text{start}}(a_1''') \leftarrow m_i\}.$$

Допустим, что агент a_i' выбрал, выполняя последовательные операторы $(\tilde{\exists}!\, x \in X) f_1(x)$ и $(\tilde{\exists}!\, y \in Y) f_2(y)$, значения $x = x_1$ и $y = y_1$. Следующий агент a_i'' этого же семейства, интерпретируя выражение для модуля вслед за первым агентом, например, выбрал новые значения $x = x_2$ и $y = y_2$, а для третьего агента a_i''' не нашлось нового истинного значения предиката $f_1(x)$. Это означает, что для третьего агента $(\underline{\tilde{\exists}!\, x \in X) f_1(x) = \text{false}}$ (напомним, что "$\underline{\quad}$" — символ операции "подчеркивание", определенной ранее), он прекращает проверку предиката $f_1(x)$ и перемещается в конец очереди агентов на интерпретацию данного выражения. Агенты a_i' и a_i'', в свою очередь, выполняют совместимые обновления предикатов в блоке:

$$\{f_1(x_1) \leftarrow \text{false}, f_1'(x_1) \leftarrow \text{true}, f_2(y_1) \leftarrow \text{false},$$

$$f_2'(y_1) \leftarrow \text{true}, f_3(x_1, y_1) \leftarrow \text{true}, f_1(x_2) \leftarrow \text{false},$$

$$f_1'(x_2) \leftarrow \text{true}, f_2(x_2) \leftarrow \text{false}, f_2'(x_2) \leftarrow \text{true},$$

$$f_3(x_2, y_2) \leftarrow \text{true}\}.$$

По завершении выполнения операций обновления в блоке агенты снова становятся в очередь к модулю. Кроме того, последний выполнивший обновление агент разблокирует все заблокированные ранее для всего семейства агентов функции и предикаты.

Дополнительные особенности интерпретации модуля семейством агентов ("мультиагентом") заключаются в следующем. В один и тот же момент времени функция или предикат могут использоваться лишь одним из агентов. Порядок начавшихся проверок функций или предикатов каким-либо агентом не может быть нарушен другим агентом и может быть прерван лишь вследствие ложного значения выражения в квадратных скобках. В противном случае может возникнуть взаимная блокировка агентов. Операции обновления функций в блоке должны быть совместимыми независимо от того, какой из агентов семейства подготовил их к выполнению.

Преимуществом предложенного подхода является то, что функции, необходимые для интерпретации модуля, сначала блокируются для агентов всего семейства (или, что эквивалентно, извлекаются из FS-пространства), а затем по строгой очередности проверяются значения функций в том порядке, в котором они встречаются в тексте модуля. Возможно использование и других способов организации блокировок информационных объектов в FS-пространстве. Похожие задачи решаются, например, при реализации распределенных баз данных.

На рис. 5 и 6 рядом с графическими изображениями для агентов-серверов и мультиагентов-серверов представлены выражения для блоков B_{ami}, B_{amk} и B_{amj} управляющих (стартовых) модулей, выполняющих запуск рабочих модулей.

Преимуществом предложенных алгоритмов, представленных на рис. 8 и 9, является то, что в их реализации активно участвуют мобильные агенты, согласующие свои действия через FS-пространство.

Для стационарных агентов (см. рис. 6) схема алгоритма A_3 выполнения модуля с учетом блокировок будет совпадать со схемой алгоритма A_1 для случая мобильных агентов, но интерпретации большинства операторных и условных вершин будут различаться. Различие обусловлено тем, что здесь агенты-серверы не клонируются, а обращаются путем передачи сообщений к специальным агентам хранения-размещения функций и предикатов в FS-пространстве. Реализующий такого рода действия алгоритм A_3 включает следующие шаги.

1. Агент-сервер a_i запрашивает у агентов хранения-размещения сведения о наличии требуемых функций и предикатов в FS-пространстве (точнее, информационных объектов, представляющих данные функции и предикаты в FS-пространстве), проверяемых и обновляемых модулем m_i.

2. Проверка истечения времени моделирования; если это время истекло, то работа агента заканчивается.

3. Агенты хранения-размещения посылают агенту a_i сообщения о наличии соответствующих функций и предикатов в FS-пространстве.

4. Агент a_i проверяет наличие сообщений от всех агентов хранения-размещения.

5. Агент a_i проверяет наличие функции f_R в FS-пространстве (данная функция описана в п. 7).

6. Агент a_i извлекает функцию f_R из FS-пространства.

7. Агент a_i проверяет условие $E_i \cap R = \varnothing$ наличия в FS-пространстве функций и предикатов из множества E_i, проверяемых в условной части модуля m_i и, возможно, модифицируемых данным модулем, а R — множество уже изъятых из FS-пространства функций и предикатов. Здесь $E_i \subseteq X$, $T_i \subseteq X$, где X — множество всех функций и предикатов с именами из Σ. В терминах технологии СеАМ агент a_i при выполнении данного пункта алгоритма проверяет истинность высказывания

$$f_{\text{isequal}}(f_{\text{inter}}(f_{Ei}, f_R), \text{emptyset}),$$

где f_{Ei} и f_R — характеристические функции множеств E_i и R соответственно, а emptyset — пустое множество.

8. Агент a_i посылает агентам хранения/размещения запрос на передачу на его сервер необходимых ему функций и предикатов. После получения всех ответов и приема запрошенных функций и предикатов агент a_i модифицирует множество $R: R \leftarrow R \cup E_i$, добавляя в него принятые на его сервер и удаленные из FS-пространства функции и предикаты. В терминах технологии СеАМ, агент a_i при этом реализует правило обновления $f_R \leftarrow \leftarrow f_{\text{union}}(f_R, f_{Ei})$, а затем характеристическую функцию f_R множества R агент a_i возвращает в FS-пространство.

9. Агент a_i проверяет истинность всех условий, необходимых для выполнения модуля m_i.

10. Агент a_i осуществляет необходимые модификации запрошенных и принятых им на свой сервер функций и предикатов. На этом выполнение модуля m_i завершается.

11. Агент a_i осуществляет передачу функций и предикатов, принятых им ранее, по местам их прежнего расположения, оповестив предварительно соответствующие агенты хранения/размещения.

12. Агент a_i модифицирует множество $R: R \leftarrow \leftarrow R \backslash E_i$, удаляя из него заблокированные ранее функции и предикаты. В терминах формализма СеАМ это эквивалентно выполнению агентом a_i правила обновления $f_R \leftarrow f_{\text{diff}}(f_R, f_{Ei})$. Для выполнения указанных действий агент a_i снова предварительно выбирает функцию f_R из FS-пространства.

13. Характеристическая функция f_R множества R возвращается агентом a_i в FS-пространство.

14. Если условие, проверяемое в п. 9, ложно, то агент a_i выполняет обновление $f_R \leftarrow f_{\text{diff}}(f_R, f_{Ei})$.

15. Выполняются действия, аналогичные п. 11.

16. Выполняются действия, аналогичные п. 13.

17. Агент a_i ожидает в течение времени таймаута ρ_i, а затем продолжает выполнение последовательности действий по запуску модуля m_i.

18. Уничтожение агента a_i.

Граф-схема данного алгоритма A_3 полностью идентична граф-схеме алгоритма A_1, представленной на рис. 8, и отличается от нее в существенной степени измененной интерпретацией ряда условий и операторов.

Блокировка локального пространства при работе модуля

Данные алгоритмы могут быть положены в основу и других модификаций алгоритмов интерпретации модулей-продукций и модулей-процедур. При интерпретации модулей-процедур, основанных на формализме РСеАМ, в ряде случаев удобна явная блокировка функций. Например, в описании РСеАМ-модулей символами "+" и "−" в верхних индексах имен функций мы будем обозначать блокировки и деблокировки функций до и после выполнения соответствующего модуля. Так, запись

$$f_k^+;\ f_e^+;\ f_m^+;\ m_i;\ f_k^-;\ f_e^-;\ f_m^- \qquad (1)$$

означает, что до выполнения некоторого модуля m_i выполняются последовательные блокировки функций f_k, f_e и f_m. После выполнения модуля m_i данные функций деблокируются. Механизмы тайм-аутов и "откатов", используемые во многих распределенных системах, при корректном применении позволят избежать тупиковых ситуаций и взаимоблокировок. Запись

$$f_i^+;\ m_i;\ f_i^- \qquad (2)$$

означает, что до выполнения модуля m_i должны быть заблокированы все используемые модулем функции. После выполнения модуля все заблокированные ранее функции деблокируются. Другой вариант записи —

$$\{f_k, f_e, f_m\}^+;\ m_i;\ \{f_k, f_e, f_m\}^- \qquad (3)$$

означает, что блокировки функций выполняются как одно атомарное действие. Аналогично же выполняется деблокировка функций. В выражениях вида (1)—(3) символ ";" эквивалентен символу определенной в РСеАМ операции "↑" (непосредственное следование), а символ "," также как и в блоках совместимых обновлений функций, соответствует операции "|" (выполнить "возможно одновременно").

Заключение

1. В работе предложено описание реализации технологии, которая соответствует такой тенденции создания сетевого программного обеспечения, при которой стираются грани между сетевой операционной системой, распределенной системой управления базой данных и распределенным приложением, а проект "видим" разработчику на всех его этапах.

2. Проиллюстрировано использование двух основных сред для распределенного моделирования и физической реализации систем хранения и обработки данных — среды, в которой мобильными агентами используются стационарные функции, и среды, в которой функции сами могут перемещаться к агентам.

3. Особенностью предлагаемого подхода является то, что используемое пространство функций и предикатов является виртуальным — функции и предикаты могут размещаться на произвольных узлах вычислительной сети, храниться в произвольных базах данных, инкапсулироваться в передаваемые сообщения, храниться и перемещаться вместе с мобильными агентами.

4. Предложены алгоритмы реализации причинно-следственных межмодульных связей, ориентированные на согласованную работу мобильных агентов, что позволяет организовать эффективную работу среды хранения и обработки данных.

Список литературы

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

2. **Зинкин С. А.** Сети абстрактных машин высших порядков в проектировании систем и сетей хранения и обработки данных (базовый формализм и его расширения) // Известия высших учебных заведений. Поволжский регион. Технические науки. — Пенза: Изд. Пенз. гос. ун-та. 2007. № 3. С. 13—22.

3. **Зинкин С. А.** Сети абстрактных машин высших порядков в проектировании систем и сетей хранения и обработки данных (механизмы интерпретации и варианты использования) // Известия высших учебных заведений. Поволжский регион. Технические науки. — Пенза: Изд-во Пенз. гос. ун-та. 2007. № 4.

4. **Gurevich Y.** Evolving Algebras 1993: Lipari Guide // Specification and Validation Methods / Ed. E. Börger. Oxford University Press, 1995. P. 9—36.

5. **Gurevich Y.** On Kolmogorov machines and related issues. The logic in computers science column // Bulletin of Europian Assoc. for Theor. Сотр. Science. 1998. N 35. P. 71—82.

6. **Глушков В. М., Цейтлин Г. Е., Ющенко Е. Л**. Алгебра. Языки. Программирование. — Киев: Наукова думка, 1978. 320 с.

7. **Lange D. B., Oshima M.** Programming and deploying Java mobile agents with aglets. Boston, Massachusetts, USA: Addison-Wesley, 1998. 225 p.

8. **Zinkin S. A.** Distributed evolving algebras meet Java aglets: executable specifications of virtual mobile metacomputing on the Internet platform // New Information Technologies and Systems. Proceedings of the Third International Conference of Science and Technology. Penza: Penza State Univ., 1998. P. 50—52.

УДК 004.822

С. М. Авдошин, канд. техн. наук, зав. каф.,
М. П. Шатилов, студент,
Государственный университет —
Высшая школа экономики, г. Москва

Информационные технологии онтологического инжиниринга

Рассмотрены основные понятия онтологического инжиниринга. Рассмотрены языки описания онтологий и занимаемая ими ниша в контексте семантической паутины. Проведен анализ инструментов онтологического инжиниринга. Описаны наиболее конкурентоспособные инструменты для построения, отображения и объединения онтологий.

Ключевые слова: *онтологический инжиниринг, классификация онтологии, инструмент разработки онтологии, семантическая паутина.*

Важно не количество знаний, а качество их. Можно знать очень многое, не зная самого нужного.
Л. Н. Толстой

Введение

"Отбросьте свои изношенные, ветхие идеи о лидерстве. Самая успешная корпорация 1990-х годов получит название обучающейся организации". Эти слова красовались на страницах журнала "*Fortune*", освещающего тенденции развития мирового бизнеса, — одного из лидеров деловой прессы. В сложном и динамично развивающемся мире конкурентное преимущество компании может быть достигнуто только благодаря использованию центральных идей, влияющих на современные тенденции развития бизнеса. Использование организацией ключевых технологий своего времени дает реальный пропуск в лидеры. Информационные порталы, электронная коммерция, системы сбалансированных показателей, реинжиниринг бизнес-процессов, управление знаниями — это лишь немногая часть тех идей, которые могут позволить компании одолеть своих соперников. Но по утверждениям многих специалистов технология управления знаниями делает их единственным источником надежного и стойкого конкурентного преимущества. Информация, накопленная годами сотрудниками компании, должна превращаться в ценные и осмысленные знания, которые в дальнейшем будут являться руководством к действию. Таким образом, конкурентное преимущество компании заключается в эффективном сборе, анализе и представлении данных для своих сотрудников, которые на основании полученных знаний смогут принимать верные решения.

Многие компании владеют огромными массивами информации, рассредоточенной по разным местам: в базах и хранилищах данных, на бумаге, в отчетах о проделанной работе и в деловой переписке, но самое главное — все компании обладают опытом, накопленным в течение многих лет и хранящимся в головах сотрудников. Проблема заключается в извлечении и концентрировании знаний в одном месте в удобной и доступной форме. Одним из важнейших и наиболее перспективных направлений в области формализации знаний является *онтологический инжиниринг.* Под онтологическим инжинирингом понимают процесс проектирования и разработки онтологии. При отсутствии общепринятой методологии и технологии этот процесс не является тривиальной задачей. Он требует от разработчиков профессионального владения технологиями инженерии знаний — от методов извлечения знаний до их структурирования и формализации [1].

Онтология — это *спецификация на концептуальном уровне* [2]. Данный термин произошел от названия раздела философии, изучающего проблемы бытия. По сути, онтология в широком смысле — это наука о бытие. В контексте информационных технологий этот термин приобрел несколько другой смысл, здесь онтология — это попытка всеобъемлющей и детальной формализации некоторой области знаний с помощью концептуальной схемы. Онтология включает в себя словарь указателей на термины предметной области и логические выражения, описывающие соотношение между ними, т. е. онтология — это сеть, содержащая термины и понятия и показывающая взаимосвязь между ними. Онтологии применяются в искусственном интеллекте и семантической паутине как форма представления знаний о реальном мире в цифровом формате.

Онтологии позволяют представить понятия в таком виде, что они становятся пригодными для компьютерной обработки, при этом остаются понятными не только машине, но и человеку. То есть по сути онтологии являются интерфейсом общения между людьми и компьютерами.

Классификация онтологий

Абстрагируясь от реализации языка описания онтологий, т. е. от программного продукта, на котором реализован этот язык, обычно используют три классификации онтологий [3]:

- по степени формальности;
- по наполнению содержимым;
- по цели создания.

Рассмотрим соответствующие классификации более подробно.

Спектр онтологий — классификация онтологий по степени формальности. На рис. 1 представлена классификация (спектр) онтологий в зависимости от деталей их реализации [3]. Косой чертой на рисунке отделены "машино-понятные" и "человеческо-понятные" описания.

Каталоги на основе ID — это конечный список терминов. Они представляют точную интерпретацию терминов. Например, говоря слово "онтология" мы будем использовать одно и то же значение, соответствующее некоторому ID в каталоге.

Словари терминов (*глоссарии*) дают грамматическую и стилистическую характеристики терминов, примеры употребления и другие сведения. Они непригодны для автоматической обработки программными агентами.

Тезаурусы — нормативные словари ключевых слов и дескрипторов (словарных единиц в виде слов, словосочетаний или кодов, называющих класс условной эквивалентности, в который включены эквивалентные и близкие по смыслу ключевые слова) [4].

Формальные таксономии [*греч.* taxis — расположение по порядку + nomos — закон) — это онтологии, которые включают точное определение отношения *isA* (класс-подкласс). Здесь соблюдается отношение транзитивности [3]: если *B* является подклассом класса *A*, то каждый подкласс класса *B* тоже является подклассом класса *A*. *B неформальных таксономиях* это отношение выполняется не всегда.

Рис. 1. Спектр онтологий

Формальные экземпляры. Не все онтологии содержат только имена классов, многие содержат на нижнем уровне экземпляры (индивиды) классов. Такого рода онтологии относятся к данной категории.

Онтологии, в структурных элементах которых содержатся *фреймы*, будут рассмотрены ниже.

В целом с необходимостью выразить больше информации выразительные средства онтологии (и ее структура) усложняются. Например, может потребоваться заполнять значение какого-либо свойства экземпляра, используя математическое выражение, основанное на значениях других свойств и даже других экземплярах. Многие онтологии позволяют объявлять два и более классов *дизъюнктивными* (непересекающимися). Это означает, что у данных классов нет общих экземпляров.

Классификация онтологии по цели создания. В данной классификации выделяют следующие типы онтологий.

- *Онтологии представления*. Цель создания таких онтологий — описать семантику языков, которые будут в дальнейшем использоваться для описания метаонтологий, онтологий предметной области и прикладных онтологий. Например, средствами RDF/RDFS [5] описаны понятия языка OWL [6].
- *Метаонтологии* не зависят от выбранной предметной области. Ими характеризуются общие понятия, на основании которых строятся онтологии более низкого уровня. Такого рода онтологии повторно используются в предметных областях.
- С помощью *онтологий предметной области* формально описываются предметные области, в которых уточняются понятия, определенные в метаонтологиях. Эти онтологии могут повторно использоваться в рамках той предметной области, для которой они были созданы.
- *Прикладные онтологии* описывают концептуальную модель конкретной задачи или приложения. Возможность повторного использования отсутствует.

Классификация онтологий по содержимому. Отличие этой классификации от предыдущей заключается в том, что здесь рассматривается реальное содержимое онтологии, а не абстрактная цель ее создания. Здесь различают *общие онтологии, онтологии конкретных задач и предметные онтологии*.

Элементы онтологии

Все современные онтологии, независимо от спецификации языка и его программной реализации, строятся примерно одинаково. Основными компонентами онтологий являются *концепты* (классы, понятия, сущности, категории), *свойства* концептов (слоты, атрибуты, роли), *отношения* между концептами и некоторые *ограничения* (*фацеты ограничения ролей*). Экземпляр концепта служит для представления элемента предметной области, группу элементов которого описывает концепт (например, человек — это концепт, а Агафья Петровна уже экземпляр концепта). Сама онтология и множество экземпляров составляют *базу знаний*.

Концепт — это шаблон, содержащий множество правил, определяющих форму экземпляра, т. е. то каким образом может быть построен экземпляр. Возвращаясь к примеру с Агафьей Петровной, человек является концептом, потому что это некоторая обобщенная сущность, не имеющая четких очертаний, но обладающая правилами, описывающая, каким образом эти очертания должны быть заданы.

Концепты могут иметь *атрибуты* — имена или структуры полей записи. Атрибуты характеризуют размер или тип информации, содержащейся в поле. Они используются для хранения информации об экземпляре концепта. *Значение атрибута* может быть как сложным, так и простым типом данных. К примеру, возраст Агафьи Петровны — это атрибут, а полное число лет, присвоенное атрибуту "возраст", является значением атрибута.

Отношениями между концептами называются зависимости между экземплярами онтологий. Обычно отношением является атрибут, ссылающийся на другой экземпляр. Например, друзья Сидор и Борислав любят поиграть во дворе Агафьи Петровны, при этом Сидор приходится ей внуком, в таком случае значением атрибута "Внук" у Сидора будет равно "Агафья Петровна".

Читатели, знакомые с идеями объектно-ориентированного программирования (ООП), могли заметить некоторую схожесть описания классов ООП с описанием классов онтологий. Объектно-ориентированное программирование сосредотачивается главным образом на методах классов — программист принимает проектные решения, основанные на операторных свойствах класса, тогда как разработчик онтологии принимает эти решения, основываясь на структурных свойствах класса [7]. Поэтому структура класса и отношения между классами в онтологии в значительной степени отличаются от структуры подобной предметной области, описанной в терминах ООП.

Формальная модель онтологии

Под формальной моделью онтологии понимают $O = < T, R, F >$, где T — конечное множество терминов предметной области, которую описывает онтология O; R — конечное множество отношений между терминами описываемой предметной области; F — конечное множество функций интерпретации, которые задаются на терминах и отношениях онтологии [8].

Здесь появляется еще одна классификация — классификация онтологий по их структуре [1], в которой модели онтологий разделяются на три группы: *простые* (в которых множества R и F могут отсутствовать), *на основе фреймов, на основе логик*.

Понятие фрейм было предложено М. Минским, одним из основателей теории искусственного интеллекта. Оно обозначает структуру данных для представления стереотипной ситуации (обозначения структуры знаний для восприятия пространственных сцен [1]). По сути, фрейм — это некоторая логическая запись, где каждому слоту ставятся в соответствие значения, присоединенные процедуры или другие фреймы. Они служат для описания ситуаций, объектов и других понятий, а также для описания взаимозависимостей между ними. Совокупность фреймов позволяет представлять и сохранять знания аналогично представлению знаний человеческим мозгом. При этом эти знания очень тесно переплетаются между собой.

Примером является все та же Агафья Петровна. Ее описание на основе сети фреймов приводится на рис. 2.

Другим типом организации структуры онтологии является описание ее структуры на основании логических моделей. В представлении знаний выделяют формальные логические модели, основанные на классическом исчислении предикатов первого порядка (*first-order logic*) и дескриптивной логике (*description logic*). В первом случае предметная область описывается конечным набором аксиом, которые являются общими для любой области объектов исследования с заданными на этих объектах предикатами (т. е. свойствами и отношениями). Поскольку такая логика предъявляет

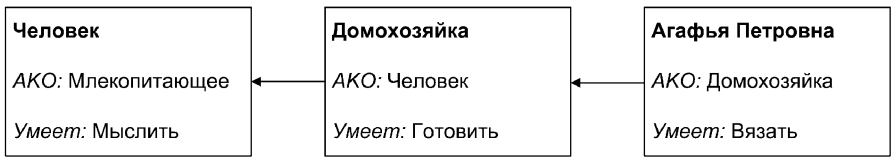

Рис. 2. Сеть фреймов (AKO — A Kind OF)

очень высокие требования и ограничения к предметной области, используется она мало. Наибольшую популярность в настоящее время получил класс дескриптивных логик.

Дескриптивные логики — это семейство формализмов для представления знаний. Они основаны на таких понятиях как концепты и роли и характеризуются конструкциями, позволяющими строить сложные концепты и роли из более простых. Основная выгода от этого семейства формализмов заключается в том, что с их помощью могут быть описаны устойчивые и полные алгоритмы для решения проблем категоризации и выполнимости. Механизм рассуждений в дескриптивных логиках решает проблемы эквивалентности, выполнимости и категоризации [8].

Применение онтологий

Исследования в области онтологий и онтологических систем довольно перспективны. С одной стороны, это развитие Семантического Веба, а с другой — искусственного интеллекта.

Одним из самых крупных проектов в области Семантического Веба является Дублинское ядро (*Dublin Core*), служащее стандартом метаданных для описания большого диапазона сетевых ресурсов [9]. Этот стандарт состоит из двух уровней: простого (*Simple*) и квалифицированного (*qualified*). Если простой уровень содержит в себе 15 элементов, то квалифицированный включает в себя еще три дополнительных элемента и группу квалификаторов, уточняющих семантику элементов. Все это качественно повышает уровень поиска необходимых данных. Например, системы помощи GNOME [10] и KDE [11] основаны на проекте "Дублинское ядро".

Проект *Enterprise project* ставит перед собой целью улучшение существующих методов моделирования работы организации [12]. Ведется разработка инструментальных средств, которые были бы достаточными для полного описания работы предприятия. Однако здесь возникают семантические конфликты, при которых под одним и тем же термином понимают разные вещи. Решением этой проблемы является построение онтологии, в которой заданы основные термины предметной области, где ведется разработка.

Онтологии используют и в системе образования, например, в проекте по разработке государственных образовательных стандартов [13].

Помимо описанных выше, существуют и другие проекты, как более мелкие, так и более крупные (например, категоризация продаваемых товаров и их характеристики на сайте Amazon.com [14]). Все это свидетельствует о том, что онтологии востребованы во многих областях человеческой деятельности.

Объем информации, содержащейся в сети Интернет, растет гигантскими темпами. Например, по данным поисковой машины Яndex в феврале 2004 г. насчитывалось 428 571 000 веб-документов в русской части Интернета, в апреле 2005 г. — 1 555 937 000, а в декабре этого же года — уже 2 538 893 000. Такой лавинообразный рост является свидетельством того, насколько быстро растут терабайты информации в русскоязычном секторе Интернета. Следовательно, алгоритмы поиска и анализа данных все более усложняются. Поисковая машина выдает на запрос тысячи вебстраниц, в которых найти нужные становится большой проблемой. Должен произойти какой-то качественный скачок в развитии сети Интернет. *Семантическая паутина* является результатом такого скачка. Если раньше компьютер мог читать документы, то теперь он может их понимать, а следовательно качество поиска сильно возрастает. Идеолог и создатель этой идеи Тим Бернс-Ли утверждает, что "Семантический Веб разрабатывает языки для выражения информации в форме, доступной для машинной обработки" [15]. По сути, Интернет останется снаружи почти таким же, каким был раньше для людей, но кардинально изменится для машин, которые смогут разбираться в "бездонной помойке", делая ее все более и более полезной для людей.

В настоящее время Семантический Веб все быстрее набирает обороты, но развит пока довольно слабо, поскольку активная работа над ним началась совсем недавно. Тем не менее, есть уже первые результаты — *Extensible Markup Language* (XML) или расширяемый язык разметки [16]. Данный язык представляет собой универсальный синтаксический фундамент, на котором могут строиться решения проблем представления данных, а также отношений между ними. Помимо XML существуют и другие, более молодые, а как следствие менее распространенные технологии: XHTML [17], XSLT [18], RDF, OWL и другие. Рассмотрение использования разработок в области XHTML и XSLT выходит за рамки данной статьи, а языки описания онтологий RDF и OWL будут рассмотрены ниже.

Языки описания онтологий

Для реализации широкого диапазона онтологий требуются мощные и гибкие языки представления. Одним из ключевых моментов в создании онтологической модели является выбор конкретного языка (*Ontology specification language*). Таких языков довольно много и разобраться в том, какое место они занимают в разработке онтологий, про-

блематично. Во-первых, язык должен быть достаточно выразительным и удобным, а во-вторых, он должен помочь пользователю абстрагироваться от низкоуровневых проблем, позволяя тем самым сосредоточиться на первостепенной задаче — проектировании онтологии.

Языки описания онтологий классифицируют следующим образом:

- *Языки, традиционно используемые сообществом создателей онтологий* (*ontology community*). Это такие языки, как DOGMA (*Developing Ontology-Grounded Methods and Applications*) [19], KIF (*Knowledge Interchange Format*) [20], OCML (*Operational Conceptual Modelling Language*) [21], LOOM [22], CycL [23], F-Logic [24] и OKBC (*Open Knowledge Base Connectivity*) [25].
- *Языки, созданные в контексте Интернет-среды* и рекомендованные консорциумом W3C: XML, RDF, RDFS, DAML [26], OIL [27], OWL. Последний является наиболее поздним из всех созданных языков (рис. 3).

Рассмотрим некоторые языки из представленных классов более подробно.

Язык **LOOM** используется для создания экспертных систем и других интеллектуальных приложений. Данный язык основан на дескриптивной логике и относится к KL-ONE-подобной группе языков [28]. Важно, что LOOM значительно выразительнее и мощнее своих предшественников − языков, основанных на фреймах, — OCML, OKBC и F-Logic. Например, очень важным преимуществом такого подхода является возможность задавать множество ограничений на созданные концепты.

В рамках проекта семантической интерпретации информационных ресурсов Интернет (Семантический Веб) предложен стандарт описания метаданных о документе **RDF** (*Resource Description Framework*), использующий XML-синтаксис. RDF использует базовую модель данных "*объект — атрибут — значение*" и способен сыграть роль универсального языка описания семантики ресурсов и взаимосвязей между ними. Важной особенностью стандарта является расширяемость: он позволяет задать структуру описания источника, используя существующие и создавая новые классы, свойства, типы, коллекции.

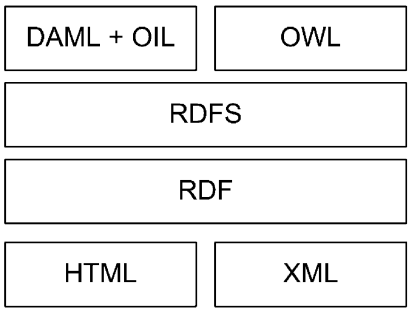

Рис. 3. Иерархия языков для веб-среды

RDF Schema (**RDFS**) — стандарт, предложенный по инициативе W3C для представления онтологических знаний. Он относится к онтологии представления и специфицирует множество допустимых схем данных. RDFS предоставляет хорошие базовые возможности для описания словарей типов предметных областей.

DAML + OIL — семантический язык разметки веб-ресурсов, расширяющий стандарты RDF и RDF Schema. Последняя версия DAML + OIL обеспечивает богатый набор конструкций для создания онтологии и разметки информации таким образом, чтобы их могла читать и понимать машина. Данный стандарт спецификации возник на базе DAML (*DARPA Agent Markup Language*) и European Commission OIL (*Ontology Inference Layer*), которые были разработаны для поддержки процесса обмена знаниями и интеграции знаний. Онтология DAML + OIL состоит из: заголовков (*headers*), элементов классов (*class elements*), элементов свойств (*property elements*), экземпляров (*instances*).

OWL (*Web Ontology Language*) — язык представления веб-онтологий, являющийся средством описания веб-документов и веб-приложений. Он расширяет возможности XML, RDF, RDF Schema и DAML + OIL. Этот проект предусматривает создание мощного механизма семантического анализа.

Управление неструктурированной информацией в сети Интернет невозможно без поддержки мощных инструментов. Чтобы структурировать эти данные, компьютерным агентам требуются машиночитаемые описания содержимого документов и характеристик доступных веб-ресурсов. Такого рода метаданные должны присутствовать в любых источниках данных. Язык OWL и является языком, представляющим возможности такого описания. Онтологии OWL — это последовательности аксиом и фактов, а также ссылок на другие онтологии. Они содержат компоненту для записи авторства и другой подробной информации.

Сами по себе языки представления онтологий не были бы так сильно востребованы, если бы не возникало необходимости автоматически обрабатывать онтологии, наполнять их содержимым и выполнять запросы к хранилищам онтологий. Наиболее популярными среди языков запросов к RDF-хранилищам являются языки RDQL [29], SPARQL [30]. В настоящий момент в *Stanford Knowledge Systems Laboratory* разрабатывается язык запросов к OWL — OWL-QL [31].

Инструментальные средства обработки онтологий

В течение последних семи лет появилось большое число инструментов для разработки и использования онтологий. Инструментальная поддержка одновременно важна как для процесса разработки (построение онтологии, ее комментирование, объединение с другими онтологиями и т. д.), так и для использования онтологических моделей в различных областях, таких как электронная коммерция, управление знаниями, Семантический Веб и др.

При создании онтологий целесообразно пользоваться подходящими инструментами. Какой тип инструментов наиболее удобен для построения онтологий? В каком формате хранятся онтологии в выбранном инструменте? В какие форматы можно экспортировать и из каких форматов можно импортировать онтологии? При выборе инструментов для работы с онтологиями необходимо ответить на эти и многие другие вопросы.

Общие характеристики редакторов

Модель знания. Под формализмом понимается теоретическая часть, которая лежит в основе способа представления онтологических знаний. Примерами формализмов могут быть описанные выше логика предикатов первого порядка, дескриптивная логика и фреймы. Помимо этого существует еще целый ряд способов представления онтологий (концептуальные графы и т. п.). Формализм влияет не только на внутреннюю структуру данных, но также может оказывать существенное воздействие на формат представления и даже на пользовательский интерфейс [3].

Формат представления онтологии задает вид хранения и способ передачи онтологических описаний. Под форматом подразумеваются языки представления онтологий: RDFS, OWL, DAML + OIL и проч.

Архитектура программного обеспечения. Различают трехуровневую, многоуровневую архитектуру и архитектуру типа клиент-сервер.

Трехуровневая архитектура предполагает наличие трех компонентов приложения: клиентское приложение; подключенное к серверу приложений; использующее сервер базы данных [32].

Многоуровневая архитектура приложения разделяет пользовательские сервисы, прикладные сервисы и сервисы данных.

В архитектуре клиент-сервер прикладные и пользовательские сервисы реализованы на клиентской рабочей станции, а данные централизованно хранятся на сервере. В этой модели клиенты подключаются непосредственно к серверу на все время работы приложений [33].

Функциональность. Важной характеристикой программного средства является его функциональность, т. е. множество сценариев его использования. В базовый набор функций входят:

- сохранение проекта в нужном формализме и формате (экспорт);
- возможность чтения различных форматов и формализмов (импорт);
- редактирование метаданных проекта (например, настройка формы редактирования и представления данных);
- создание и редактирование онтологии.

В дополнительный набор функций могут входить, например, возможность поддержки какого-либо языка запросов, анализ целостности, возможность представления онтологии в виде графа с его последующим анализом и редактированием, извлечением информации и проч.

Инструменты

Инструменты работы с онтологиями сгруппированы в таблицу. Рассмотрим наиболее мощные из них:

- редакторы общего назначения Protégé и Ontolingua [34];
- редактор, ставящий своей целью развитие Семантического Веба, Altova Semantic Works 2008 [35];
- плагин PROMPT к Protégé, позволяющий объединять и группировать онтологии [45];
- инструмент CmapTools [36], позволяющий конструировать, управлять, делить и критиковать модели знаний, представленных в форме конецептуальных карт (*concept maps*).

Наибольший интерес представляет редактор Protégé. Это свободно распространяемая Java-программа, созданная лабораторией медицинской информатики школы медицины Стендфордского университета. Первоначально Protégé предназначался для моделирования в области медицины, но теперь используется и в других предметных областях. Поскольку данный продукт является продуктом с открытым исходным кодом, для него написано множество плагинов. Также для него имеется довольно обширная база знаний, а объединение людей вокруг этого продукта на официальном сайте Protégé [37] насчитывает более 80 тысяч человек. В настоящее время данный продукт является наиболее перспективным редактором онтологий из всех существующих.

Protégé предназначен для построения (создания, редактирования и просмотра) онтологий прикладных областей. Инструментальные средства, входящие в состав продукта, включают редактор онтологии, позволяющий проектировать онтологии, создавая их иерархическую структуру аб-

страктных и конкретных классов. На основе сформированной онтологии Protégé позволяет генерировать формы получения знаний для введения экземпляров классов и подклассов [38].

В рамках проекта на основе Protégé создана линейка, состоящая из двух продуктов: Protégé-Frames и Protégé-OWL. Protégé-Frames использует фреймовую модель представления знаний OKBC (*Oven Knowledge Base Connectivity*), это позволило адаптировать его и для редактирования моделей предметной области, представленных не в OWL, а в других форматах (UML, XML, SHOE [39], DAML+OIL, RDF и RDFS и т. п.). В основе Protégé-OWL лежит дескриптивная логика, что по-

Название программы	Разработчик	Версия	Дата последнего релиза	Доступность	Расширяемость	Форматы импорта/экспорта онтологий	Представление в виде графа	Извлечение информации	Ссылки
Редакторы									
Protégé	Stanford Medical Informatics	3.3.1	Август 2007	Open source	+	RDF, RDFS, DAML + OIL, XML, OWL, Clips, UML, FLogic, Java, XMI	Да	Нет	www.protege.stanford.edu
OntoEdit	Ontoprizse	2.6.6	Март 2004	Свободная лицензия	+	OXML, RDF(S), DAML + OIL, FLogic	Да	Нет	www.ontoprise.de/products/ontoedit
Ontolingua	KLS, Stanford University	1.0.650	Октябрь 2002	Свободный доступ	—	Ontolingua, KIF, CML, DL, LOOM, CLIPS, Epikit, Prolog, IDL	Нет	Нет	www.ksl.stanford.edu/software/ontolingua
Apollo	Knowledge Media Institute of Open University, UK	1.0	Август 2004	Open source	+	RDF, XML, Meta, OCML		Нет	www.apollo.open.ac.uk m.koss@open.ac.uk
SemanticWorks	Altova	2008	2008	Лицензия	—	RDF, XML, OWL	Да	Нет	www.altova.com/products/semanticworks/semantic_web_rdf_owl_editor.html
LinkFactory	Language and Computing nv	3.3 19-Nov	2003	Лицензия	—	XML, RDF(S), DAML + OIL/OWL	Нет, но возможно посмотреть графическое представление с помощью Java Beans	Да	www.landc.be info@landc.be
Объединение и отображение онтологий									
PROMPT Плагин к Protégé	Noy and Musen	2.4.8	Июнь 2005	Open source	—	см. Protégé	см. Protégé	см. Protégé	www.protege.stanford.edu/plugins/prompt
OntoMerge	Yale University	0.1 alpha	Апрель 2002	—		RDF, DAML + OIL, OWL, WSDL, XML	Нет	Нет	www.cs.yale.edu/homes/dvm/daml/ontology-translation.html
Visio 2007	Microsoft Corp.	12.0.4518	Декабрь 2006	Лицензия	+	XML	Да	Нет	www.microsoft.com
Chimaera	KSL, Stanford University	0.1.45	Август 2003	Свободный доступ	—				www.kls.stanford.edu./software/chimaera
Извлечение информации									
CmapTools	Institute for Human and Machine Cognition, University of West Florida	4.11	Июль 2007	Свободный доступ	—	Отсутствуют	Да	Да	www.ihmc.us
OntoX	Technion — Israel Institute of Technology	1.0	Май 2004	Свободный доступ	—	XML с поддержкой BizTalk; (RDF & OWL planned)	Только просмотр	Извлечение онтологий из веб-ресурсов с использованием элементов form и ссылок	www.iew3.technion.ac.il/OntoBuilder/
VisualText Conseptual Grammar KB Editor	Text Analysis International, Inc.	1.7	Октябрь 2003	Свободный доступ	—	HTML, XML. Возможность выгрузки в БД	Да	Да	www.textanalysis.com
K-Infinity Knowledge Builder	Intelligent Views GmbH	2.0	Сентябрь 2003	Свободный доступ	+	RDFS и XM	Да	Да	www.i-views.de

зволяет использовать его в контексте Семантического Веба.

Редактор **Protégé-OWL** является расширением Protégé, позволяющим пользователю:

- открывать и сохранять OWL и RDF онтологии;
- редактировать и впоследствии визуализировать классы, свойства и правила SWRL (*Semantic Web Rule Language*) [40];
- определять логические характеристики класса как выражения OWL;
- выстраивать механизм рассуждений таким образом, как это сделано в рамках классификаторов дескриптивной логики.

Гибкая архитектура **Protégé-OWL** позволяет легко конфигурировать и расширять редактор с помощью плагинов. Protégé-OWL тесно интегрирован с каркасом Java приложений Jena. Protégé-OWL имеет открытый исходный код API, написанный на Java, с помощью которого можно разрабатывать всевозможные компоненты или создавать различные сервисы и ресурсы Семантического Веба [41].

Редактор **Protégé-Frames** предоставляет полноценный пользовательский интерфейс и базу знаний (*knowledge server*) для разработки и хранения онтологий. В редакторе существует возможность настройки форм для ввода данных об объектах класса. Возможности редактора включают:

- разнообразие элементов пользовательского интерфейса, которые можно видоизменять так, чтобы пользователь посредством запросов к серверу мог как получать, так и отправлять данные, при этом информация, которую вводит пользователь, понятна как человеку, так и компьютеру;
- архитектура продукта построена таким образом, чтобы в программу можно было встраивать дополнительные компоненты. Примерами таких компонентов являются графические компоненты, включающие графики и таблицы; медиа-компоненты, включающие звуки, изображения и видео; различные форматы хранения данных (RDF, XML, HTML и т. д.).
- API, написанный на Java, предоставляет доступ к использованию и отображению онтологий, созданных с помощью редактора, другим приложениям и встроенным в Protégé-Frames плагинам [42].

Altova SemanticWorks 2008 является революционным продуктом в области создания и обработки RDF/OWL онтологии [16]. Действительно на данный момент этот продукт является единственным представителем в классе визуальных редакторов онтологий для Семантического Веба. Поскольку код генерируется автоматически согласно графическому виду создаваемой онтологии, пользователь может сфокусироваться на логической и семантической составляющих проектирования. По мере работы с программой пользователь может переключаться между графическим и текстовым видами редактора в целях просмотра и редактирования сгенерированного кода. Программный продукт имеет следующие возможности:

- графические RDF/RDFS/OWL редакторы;
- поддержка OWL Lite, OWL Full и OWL DL;
- специальные средства для просмотра больших онтологий;
- проверка документа на семантическую правильность.

Для использования данного продукта требуется лицензия.

Ontolingua. Кроме собственно редактора онтологии, система Ontolingua содержит: сетевой компонент **Webster,** предназначенный для определения концептов; сервер, обеспечивающий доступ к онтологиям Ontolingua по протоколу OKBC; **Chimaera** — инструментарий для анализа и объединения онтологий [38]. К сожалению, в настоящее время система больше не развивается. Большую часть продуктов для работы с онтологиями постигла та же участь. В 2004 году программных средств, поддерживаемых и развиваемых, было около сотни, а к настоящему времени их осталось не более двадцати [44].

PROMPT является дополнением к системе Protégé, реализованным в виде плагина. PROMPT позволяет совершать следующие операции:

- сравнивать две версии похожих онтологий и на основании этого дополнять одну онтологию другой;
- объединять две онтологии в одну;
- извлекать некоторую часть онтологии с целью дальнейшего использования.

По двум онтологиям, которые надо объединить, PROMPT строит список операций (например, объединение терминов или их копирование в новую онтологию) и передает его пользователю, который может выполнить одну из предлагаемых операций. Затем список операций модифицируется, и создается список конфликтов и их возможных решений. Это повторяется до тех пор, пока не будет готова новая онтология.

Инструмент **IHMC CmapTools,** разработанный IHMC (*Institute for Human and Machine Cognition*), позволяет конструировать, управлять, делить и критиковать модели знаний, представленных в форме концептуальных карт (*concept maps, c-map*). Данная программа позволяет пользователям, помимо множества других возможностей, создавать собственные концептуальные карты на ПК и затем отсылать их на сервер (*Cmap Server*), соединять их с другими концептуальными картами на сервере, автоматически создавать веб-страницы из них, а также разрабатывать концептуальные карты синхронно с другими пользователями в сети Интернет.

Заключение

В таблице приведены сведения об основных и наиболее важных параметрах, которые могут повлиять на решение о выборе инструмента. В настоящее время число инструментов онтологического инжиниринга превышает сотню, но развивающихся и поддерживаемых продуктов очень мало. Поэтому в данной таблице рассмотрены только те инструменты, которые поддерживаются и развиваются в настоящее время, а также наиболее мощные средства, интересные с научной точки зрения, поддержка которых, к сожалению, не ведется.

Проблемы выбора инструмента для работы с онтологиями, по мнению авторов, как таковой не существует, поскольку в каждой из групп существуют явные фавориты. Так, например, наиболее удобными редакторами онтологий являются Protégé и Altova Semantic Works 2008, а лидером в области объединения и отображения онтологии является плагин PROMT к Protégé.

Список сокращений

ООП — объектно-ориентированное программирование.

AKO — A Kind OF — ссылка на фрейм, определяющая отношение наследования свойств.

API — *Application Programming Interface* — интерфейс прикладного программирования, набор стандартных программных прерываний, вызовов процедур (функций, методов) и форматов данных, которые могут использовать прикладные программы для запроса и получения от операционной системы или программного интерфейса связанного с ними обслуживания.

Data Mining — технология анализа хранилищ данных, базирующаяся на методах искусственного интеллекта и инструментах поддержки принятия решений.

DAML — *DARPA Agent Markup Language* — язык разметки Семантического Веба.

DARPA — *The Defense Advanced Research Projects Agency* — Агентство перспективных исследований МО США. В 1973 г. агентство создало сеть ARPAnet, которая впоследствии дала начало Internet.

DOGMA — *Developing Ontology-Grounded Methods and Applications*

GNOME — *GNU Network Object Model Environment* — свободная среда рабочего стола для UNIX-подобных операционных систем.

HTML — *Hyper Text Makup Language* — язык гипертекстовой разметки.

ID — *Identifier* — идентификатор.

KDE — *K Desktop Environment* — графический пользовательский интерфейс фирмы Corel.

KIF — *Knowledge Interchange Format.*

LOOM — язык и среда для создания интеллектуальных приложений.

OCML — *Operational Conceptual Modeling Language.*

OKBC — *Open Knowledge Base Connectivity* — прикладной интерфейс программирования для доступа к базам знаний систем представления знаний.

OMG — *Object Management Group* — группа управления объектами, некоммерческое объединение, разрабатывающее стандарты для создания интероперабельных (платформо-независимых) приложений уровня предприятий.

RDF — *Resource Description Framework* — технология разработана W3C для описания содержимого сайтов, более глобально — для построения семантической сети Интернета.

RDFS — *RDF Schema* — стандарт, предложенный по инициативе W3C для представления онтологических знаний.

RDQL — *RDF query language* — язык запросов к RDF.

OWL — *Web Ontology Language* — язык онтологии для Интернета.

OWL-QL — *OWL query language* — язык запросов к OWL.

SHOE — *Simple HTML Ontology Extensions* — XML-совместимый язык представления знаний для Web.

SPARQL — *SPARQL Protocol and RDF Query Language* — разрабатывающийся стандарт семантической паутины, в настоящее время проходящий стандартизацию консорциума W3C. К настоящему времени уже существует несколько различных реализаций.

SWRL — *Semantic Web Rule Language.*

UML — *Unified Modeling Language* — унифицированный язык моделирования.

W3C — *World Wide Web Consortium* — организация, разрабатывающая и внедряющая технологические стандарты для Всемирной паутины.

XHTML — *Extensible HTML*. Язык XHTML разработан W3C и предназначен для поддержки XML в Web-страницах.

XML — *eXtensible Markup Language* — расширенный язык разметки.

XSL — *Extensible Stylesheet Language* — расширяемый язык таблиц стилей.

XSLT — *Extensible Stylesheet Language Transformations* — часть спецификации XSL, задающая язык преобразований XML-документов.

Список литературы

1. **Гаврилова Т. А., Хорошевский В. Ф.** Базы знаний интеллектуальных систем. СПб.: Питер, 2001.
2. **Tom Gruber.** What is an Ontology? // Available at: www.ksl.stanford.edu/kst/what-is-an-ontology.html
3. **Добров Б. В., Иванов В. В., Лукашевич Н. В., Соловьев В. Д.** Онтологии и Тезаурусы. Казань, 2006.
4. **Мильчин А. Э.** Издательский словарь-справочник. М.: ОЛМА-Пресс, 2006.
5. **Resource** Description Framework (RDF) // Available at: www.w3.org/RDF/
6. **Web Ontology** Language (OWL) // Available at: www.w3.org/2004/OWL/
7. **Noy N. F., Deborah L.** Ontology Development 101: A Guide to Creating Your First Ontology. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, 2001.
8. **Gao Shu, Omer F. Rana, Nick J. Avis, Chen Dingfang.** Ontology-based semantic matchmaking approach // Advances in Engineering Software 38 (2007). P. 59—67.
9. **The Dublin** Core Metadata Initiative // dublincore.org/
10. **GNOME**// www.gnome.org/
11. **KDE** // www.kde.org/
12. **The Enterprise** Ontology // www.aiai.ed.ac.uk/project/enterprise/enterprise/ontologv.html
13. **Никитин В. В.** Информационно-методическое обеспечение. М.: МАКС Пресс, 2005.
14. **Amazon.com** // amazon.com/
15. **Transcript** of Tim Berners-Lee's talk to the LCS 35th Anniversary celebrations, Cambridge Massachusetts, 1999/April/14 // Available at: www.w3.org/1999/04/13-tbl.html
16. **Extensible** Markup Language (XML) // www.w3.org/XML
17. **XHTML2** Working Group Home Page // www.w3.org/MarkUp/
18. **The Extensible** Stylesheet Language Family (XSL) // www.w3.org/Style/XSL/
19. **DOGMA** // en.wikipedia.org/wiki/DOGMA
20. **Knowledge** Interchange Format (KIF) // www.ksl.stanford.edu/knowledge-sharing/kif/
21. **OCML** Operational Conceptual Modeling Language // kmi.open.ac.uk/projects/ocml/
22. **MacGregor R.** Inside the LOOM classifier// SIGART bulletin. 1991. № 2. P. 70—76.
23. **CycL** // en.wikipedia.org/wiki/CycL
24. **F-Logic** // en.wikipedia.org/wiki/F-Logic
25. **Open** Knowledge Base Connectivity // www.ai.sri.com/~okbc/

26. **DAML** //www.daml.org/about.html
27. **OIL**// www.ontoknowledge.org/oil/
28. **Wikipedia**// en.wikipedia.org/wiki/KL-ONE
29. **RDF** Query Survey // www.w3.org/2001/11/13-RDF-Query-Rules/
30. **SPARQL** Query Language for RDF // www.w3.org/TR/rdf-sparql-query/
31. **OWL-QL** Project for the Stanford Knowledge Systems Laboratory // www-ksl.stanford.edu/projects/owl-ql/
32. **Википедия.** Свободная энциклопедия // www.wikipedia.org
33. **Словарь** по естественным наукам. Глоссарий.ру // www.glossary.ru
34. **Ontolingua** // www.ksl.stanford.edu/software/ontolingua/
35. **Altova** Semantic Works 2008 // www.altova.com/products/semanticworks/semantic_web_rdf_owl_editor.html
36. **IHMC** CmapTools // cmap.ihmc.us/download/
37. **The Protégé** Ontology Editor and Knowledge Acquisition System // protege.stanford.edu/
38. **Гладун А. Я., Рогушина Ю. В.** Онтологии в корпоративных системах // Корпоративные системы. 2006. № 1.
39. **SHOE** // www.cs.umd.edu/projects/plus/SHOE/
40. **SWRL:** A Semantic Web Rule Language Combining OWL and RuleML // www.w3.org/Submission/SWRL/
41. **Protege** OWL // Available at: protege.stanford.edu/overview/protege-owl.html
42. **Protege** Frames // Available at: protege.stanford.edu/overview/protege-frames.html
43. **Altova** Semantic Works 2008 // www.altova.com/products/semanticworks/semantic_web_rdf_owl_editor.html
44. **Michael Denny.** Ontology Building: A Survey of Editing Tools // Available at: www.xml.com/pub/a/2002/11/06/ontologies.html
45. **Prompt** // Available at: protege.cim3.net/cgi-bin/wiki.pl?Prompt

УДК 004.89

Л. В. Найханова, канд. техн. наук, зав. кафедрой,
Восточно-Сибирский государственный
технологический университет

Генерация множества ядер продукционных правил в задаче автоматического построения библиотеки декларативных методов

Предлагается решение проблемы автоматического построения продукционных правил на основе применения генетического программирования. Сгенерированные правила предназначены для формирования библиотеки методов построения онтологий.

Ключевые слова: *продукционные правила, генетическое программирование, построение онтологии, структура хромосомы*

Введение

Для задач, имеющих слабоструктурированные предметные области, обычно отсутствует объективная модель решения. Как правило, такие задачи решает эксперт. Разработка методов автоматического создания правил решения задачи на основе конструктивных знаний эксперта позволит сформировать библиотеку декларативных методов, обладающую свойствами долговечности и масштабируемости. Это актуально в связи с тем, что экспертам зачастую трудно сформулировать правила, которые они используют при решении задач, так как экспертное знание в большинстве случаев является подсознательным. Именно подсознательный характер экспертного знания вызывает трудности при построении экспертных систем, а извлечение экспертных знаний считается "узким местом" искусственного интеллекта [1]. В работе [2] для выявления экспертных решающих правил в базах знаний использована методология эволюционного моделирования.

В данной работе предлагается генетический алгоритм генерации множества ядер продукционных правил для решения некоторой прикладной задачи из области естественно-языковой обработки информации. Множество ядер используется для формирования системы продукций, представляющей собой декларативное представление определенного метода естественно-языковой обработки научных текстов. Совокупность методов должна составить библиотеку методов, необходимую для автоматического построения онтологий предметной области. К ним относятся и методы, предназначенные для извлечения знаний из терминологических словарей и научных текстов, суть которых заключается в извлечении знаний о терминах и отношениях между ними. В дальнейшем будем приводить примеры по методу извлечения знаний о терминосистеме из терминологического словаря с неявно выраженной структурой словарных статей.

Описание проблемы

Продукционная модель или модель, основанная на правилах, позволяет представить знания в виде утверждений, имеющих вид предложений естественного языка типа: "Если A, то B". Под антецедентом A и консеквентом B понимается некоторое множество фактов. Каждый факт имеет структуру простой ядерной конструкции языка ситуационного моделирования xRy, где x, y — термины, R — семантическое отношение.

Для полноты картины рассмотрим утверждение, которое может использоваться для выявления квалитативного отношения агрегации "Часть-целое" в предложении "Аванс составляет часть общей стоимости договора" [3]. На некотором заданном подмножестве естественного языка оно имеет следующее представление:

ЕСЛИ

<предложение > s	{содержит}	<термин> t_1	И
<термин> t_1	{имеет}	<индекс> i	И
<предложение> s	{содержит}	<глагол> v	И
<глагол> v	{имеет}	<индекс> $(i + 1)$	И
<глагол> v	{соответствует}	<СемОтношение> r	И
<СемОтношение> r	{относится к}	<ГруппеОтношений> ["ЧастьЦелое"]	И
<предложение > s	{имеет}	<термин> t_2	И
<термин> t_2	{имеет}	<индекс> $(i + 2)$	И
<термин> t_2	{имеет}	<терм-спутник> ts_{xy}	И
<терм-спутник > ts_{xy}	{имеет}	<значение> ["часть"]	

ТО

<термин> t_1	{является}	<категория> ["Целое"]	И
<термин> t_2	{является}	<категория> ["Часть"]	И
<СемОтношение> r	{является}	<категория> ["ЦелоеЧасть"]	

Данное утверждение записано в обобщенном виде и позволяет выявить, что в исходном предложении к категории "целое" относится термин "общая стоимость договора", а к категории "часть" — термин "Аванс", а также существующее между ними отношение "Целое → Часть". Структура простой ядерной конструкции каждого факта утверждения просматривается в явном виде. Например, в первом факте "<предложение> s содержит <термин> t_1" x = "<предложение> s", R = "содержит", y = "<термин> t_1".

Аналогичным образом можно выявлять родовидовые и многие другие отношения. Естественно, что подобное утверждение не покрывает все возможные ситуации, складывающиеся в предложениях и отражающие какое-либо отношение. Это означает, что в принципе необходимо, во-первых, определить структуру элементов ядерной конструкции, во-вторых, сформировать множество возможных элементов, составляющих x, R, y.

Будем считать, что известно множество термов T, обозначающих x и y, значений V или знаков термов S, соответствующих рассматриваемому методу. Вся информация представляется в XML-описании метода.

На основе этой информации необходимо сформировать допустимое множество ядер продукций NPK (*Nature Production Kernel*) в виде утверждений на ограниченном подмножестве естественного языка при следующих ограничениях: утверждение должно содержать не менее двух и не более 18 фактов; факты, составляющие утверждение, должны быть корректными.

Окончательное утверждение на ограниченном подмножестве естественного языка должно удовлетворять следующим требованиям:

1) термы должны быть заключены в угловые скобки < >;

2) семантические отношения должны быть заключены в фигурные скобки {};

3) значения должны быть заключены в квадратные скобки [];

4) в утверждении после терма должно следовать его обозначение;

5) выражение должно записываться в круглых скобках;

6) служебные слова ЕСЛИ, ТО и логические связки И, ИЛИ должны записываться без кавычек;

7) термы должны быть нормализованы, т. е. находиться в именительном падеже и единственном числе;

8) вектор должен записываться как функция.

Для решения поставленной задачи используем методологию генетического программирования.

Основные положения генетического алгоритма

В данной работе за основу взяты следующие положения генетического алгоритма.

1. Ядро продукционного правила представляется в виде хромосомы.

2. Хромосома состоит из динамического набора молекул ДНК в соответствии с выбранным представлением.

3. Молекула ДНК состоит из статического количества генов (минимальная неделимая составная часть молекулы ДНК).

4. Первая популяция $P(0)$ генерируется случайным образом, при этом значения каждого гена выбираются из соответствующей области допустимых значений, заданной при описании рассматриваемого метода.

5. Каждое новое поколение $P(t)$ генерируется с помощью оператора кроссинговера. Родительские пары выбираются оператором репродукции по методу заданной шкалы.

6. В каждой новой хромосоме некорректные и повторяющиеся молекулы подвергаются мутации.

7. Все повторяющиеся хромосомы из популяции удаляются.

8. Решением генетического алгоритма является не одна особь, а совокупность особей, удовлетворяющих приведенным требованиям.

9. Оцениваются молекулы ДНК, особи и совокупности особей.

Функция оценки каждого объекта оценки строится на основе использования эвристических правил. Совокупность особей оценивается посредством использования нечетких продукционных правил. После каждого вида оценивания выполняется оператор мутации или кроссинговера с целью улучшения особей популяции. Когда функция оценки совокупности особей достигнет или превысит значение 0,9, проводится окончательная их оценка, которая обеспечит изменение популяции, направленное на решение конкретной прикладной задачи. Для оценки совокупности особей используются объекты, внешние для генетического алгоритма: нечеткий регулятор *FuzzyRegulator* и система логического резолютивного вывода *LogResDed* (*Logical Resolutive Deduction*). Вторая система определяет приспособленность совокуп-

ности особей к решению поставленной прикладной задачи, другими словами, определяет возможность представлять собой множество NPK.

Определение структуры хромосомы

Структура хромосомы определяется следующим шаблоном утверждения:

PP = <ЕСЛИ, (*PX*, *PR*, *PY*), <логическая связка>, ..., <логическая связка>, (*PX*, *PR*, *PY*), ТО, (*PX*, *PR*, *PY*), <логическая связка>, ..., <логическая связка> , (*PX*, *PR*, *PY*)> , где

ЕСЛИ, ТО — служебные слова;

PX, *PR*, *PY* — шаблоны компонентов *x*, *R*, *y* простой ядерной конструкции;

И, ИЛИ — логические связки.

Так как несущую часть утверждения составляет простая ядерная конструкция, то рассмотрим шаблоны *PX*, *PR*, *PY*. Шаблоны *PX* и *PY* состоят из двух частей: терм метода, обозначение терма или выражения для терма. В отличие от шаблона *PX* в шаблон *PY* могут входить константы, которые имеют три вида: текстовая, символьная или числовая. Терм метода обоих шаблонов имеет два типа — базовый и собственный. Кроме того, базовый или собственный терм может содержать терм-спутник термина. Шаблон *PR* состоит из глагола и терм-спутника семантического отношения.

Таким образом, исходной информацией для генерации множества ядер продукций по заданному методу являются: множество базовых термов T_B; множество собственных термов T_O метода; множество терм-спутников T_{Sxy} термина; множество типов констант C_T; множество допустимых значений переменных *Val*; множество допустимых операций выражения *Op*; множество глаголов *Verb*, являющихся основой семантического отношения *R*; множество терм-спутников T_{SR} семантического отношения *R*; множество семантических отношений $R = \{(v, t) | v \in Verb, t \in T_{SR}\}$ и множество терм-признаков T_C. Так, в приведенном выше примере к базовым термам относятся предложение, термин, терм-спутник, глагол, индекс, значение, категория, СемОтношение, ГруппаОтношений; к терм-спутникам T_{Sxy} — часть; к терм-спутникам T_{SR} — предлог „к". Собственные термы метода и терм-признаки в данной продукции не используются.

Таким образом, хромосома состоит из динамического множества молекул ДНК. ДНК образована из генов, которые имеют десятичное представление: ген 1 — ключевое слово или логическая связка; ген 2 — тип терма (компонент *x*); ген 3 — базовый или собственный терм, или терм-спутник $t \in T_{Sxy}$, или терм-признак (компонент *x*); ген 4 — тип значения терма (компонент *x*); ген 5 — обозначение терма или выражение (компонент *x*); ген 6 — глагол (компонент *R*); ген 7 — терм-спутник (компонент *R*); ген 8 — тип терма (компонент *y*); ген 9 — базовый или собственный терм, или терм-спутник $t \in T_{Sxy}$, или терм-признак (компонент *y*); ген 10 — тип значения терма (компонент *y*); ген 11 — обозначение терма или выражение для терма, или константа (компонент *y*).

Определение мощностей перечисленных выше конечных множеств осуществляется автоматически по данным о методе, записанным в конфигурационном файле, имеющем XML-описание.

Схема алгоритма генерации продукционных правил

Для решения задачи был разработан генетический алгоритм, основанный на методологии эволюционного программирования [4]. Цель алгоритма — построить минимальное по мощности множество правил $R_1, ..., R_n$ такое, что они, являясь базовыми правилами метода, корректно решают прикладную задачу, соответствующую методу. Схема алгоритма представляется следующим образом.

1. Создание начальной популяции.
2. Первоначальное оценивание, редукция и мутация.
3. Селекция, мутация, скрещивание, редукция.
4. Оценивание молекулы ДНК; если функция оценки каждой *i*-й молекулы $F_i^{DNK} = \text{Max}$, то переход на пункт 5, иначе — на пункт 3.
5. Оценивание особи; если функция оценки *i*-й особи $F_i^{Chr} = \text{Max}$, то переход на пункт 6, иначе — на пункт 3.
6. Оценивание совокупности особей с помощью системы *FuzzyRegulator*; если функция оценки $F^{Pop} \geqslant 0{,}9$, то переход на пункт 7, иначе — на пункт 3.
7. Оценивание совокупности особей с помощью системы *LogResDed*, если *Fitness*-функция $F(PS) = 1$, то останов, иначе — на пункт 3.

Особью является ядро продукционного правила, представленное множеством молекул ДНК. В соответствии с положениями генетического алгоритма вначале случайным образом создается начальная популяция $P(0)$. При этом необходимо проследить, чтобы начальная популяция имела набор особей, непосредственно описывающих верхние и нижние граничные значения генов. Это позволит создать разнообразную (с хорошей вариативностью) и качественную (с хорошим фенотипом) популяцию.

Оценивание популяции

Оценивание сгенерированной популяции разделяется по объекту оценки на следующие виды: оценивание молекулы ДНК, оценивание особи, оценивание популяции.

Кроме того, осуществляется первичное оценивание каждой особи после создания начальной популяции.

Первичное оценивание особи. В каждой особи должны присутствовать молекулы ДНК со служебными словами ЕСЛИ и ТО, причем молекула ДНК со служебным словом ЕСЛИ должна быть первой. Также осуществляется проверка наличия повторяющихся молекул ДНК. Если какая-либо молекула ДНК особи не удовлетворяет выставленным требованиям, то делается пометка с указанием типа некорректности, для устранения которой выполняются генетические операторы. Например, каждая повторяющаяся молекула подвергается мутации. Функция оценки особи имеет аддитивный характер:

$$F_i^1 = \sum_{j=1}^{n} \sum_{l=1}^{m} e_{ijl},$$

где e_{ijl} — l-я ошибка в j-й молекуле ДНК i-й особи; m — число ошибок в молекуле; n — число ошибок в хромосоме.

Оценивание молекулы ДНК. Для оценивания молекулы ДНК применяются эвристические правила оценки, посредством которых проверяется и оценивается корректность сгенерированного факта:

1) корректность использования базового терма:

- в компоненты x и y одной ядерной конструкции могут входить термы $t \in T_B$ или $t \in T_O$;
- в одной ядерной конструкции в компонент x может входить терм $t \in T_B$, а в компонент y — терм $t \in T_O$ и наоборот;

2) корректность использования терм-спутника T_{Sxy};

- если в компонент x входит терм $t \in T_{Sxy}$, то в компонент y должен входить либо $t \in T_B$, либо $t \in T_O$ или наоборот;

3) корректность использования терм-спутника T_{SR}:

- R может состоять только из термов $v \in Verb$ и $t \in T_{SR}$;
- термы $t \in T_{SR}$ и $v \in Verb$ должны соответствовать друг другу;

4) корректность сгенерированного выражения:

- операнды выражения могут быть обозначениями термов, числовыми, текстовыми или символьными значениями, при этом типы констант должны соответствовать типам значений;
- операции должны принадлежать допустимому множеству операций Op и соответствовать типу констант;
- если в выражении несколько констант, то они должны быть совместимого типа;

5) корректность обозначений:

- каждый терм метода должен иметь свое обозначение;
- обозначения одинаковых термов должны различаться индексом;

6) корректность констант:

- константа может быть составной частью только y;
- константа должна принадлежать допустимому множеству значений терма Val, стоящего в позиции слева.

При применении любого правила к оцениваемому факту ему приписывается +1, если оно выполняется, в противном случае назначается штраф — 1. Общая оценка молекулы ДНК вычисляется по формуле:

$F_i^{DNK} = \sum_{j=1}^{m} b_j$, где m — число примененных правил к факту; b_j — оценка выполнимости правил. Тогда первичная оценка хромосомы будет иметь вид:

$F_l^{Chr} = \sum_{i=1}^{n} F_i^{DNK}$, где n — число молекул ДНК в хромосоме, l — индекс хромосомы в популяции. Некорректные молекулы ДНК должны помечаться с указанием типа некорректности.

Оценивание хромосомы. Для этого также используются эвристические правила определения корректности созданной особи:

1) если терм метода t входит в состав компонента y и появился в i-м факте впервые, то в j-м факте ($j > i$) он должен входить в компонент x;

2) объектом логического вывода утверждения может быть терм $t \in T_O$ или $t \in T_B$, который всегда должен присутствовать в факте консеквента;

3) если $k = 1$ (k — число фактов, входящих в консеквент), то терм метода $t \in T_O$ или $t \in T_B$, входящий в состав компонента x факта консеквента, должен обязательно входить в компоненты x или y хотя бы одного факта антецедента;

4) если $k > 1$, то $m - r$ термов $t_i \in T_O$ или $t_i \in T_B$ ($i = 1, ..., m$, m — число термов в консеквенте, r — число объектов логического вывода) должны присутствовать в компонентах x и y фактов антецедента.

Общая оценка хромосомы вычисляется по следующей формуле:

$F_l^{Chr} = \sum_{j=1}^{m} b_j$, где m — число примененных правил к хромосоме;

$$b_j = \begin{cases} +1, \text{ если правило применимо к факту и выполнимо,} \\ -1, \text{ если правило применимо к факту, но невыполнимо.} \end{cases}$$

Все хромосомы, у которых $F_l^{Chr} < 0$, удаляются из популяции. Тогда первичная оценка популяции будет иметь вид:

$F^{Pop} = \sum_{i=1}^{n} F_i^{Chr}$, где n — число хромосом в популяции.

Некорректные особи должна помечаться с указанием типа или типов некорректности.

Оценивание группы особей, как модели системы продукций. Цель вычисления *Fitness*-функции $F(PS)$ заключается в определении полноты пред-

ставления метода, т. е. совокупность особей должна покрывать множество терм-спутников, множество терм-признаков, множество собственных термов метода, и хотя бы частично множество базовых термов метода.

В работе вычисление $F(PS)$ осуществляется на основе использования нечеткого регулятора *FuzzyRegulator*. Для нечеткого логического вывода оценки $F(PS)$ будем использовать метод нечеткого регулирования Мамдани [5].

Компоненты нечеткого вывода рассмотрим на примере *оценивания полноты покрытия множества терм-спутников метода*. База правил нечетких продукций *FPr1* состоит из следующих элементов.

Имя продукции (N): номер нечеткой продукции $N = 1, ..., 81$.

Сфера применения (Q): оценивание степени покрытия множества TS множеством TS', где TS — множество терм-спутников, TS' — множество терм-спутников, использованных в сгенерированной системе продукций.

Условие применимости (P): $TS \neq \varnothing$.

Условие ядра (A): составное нечеткое высказывание вида "$\alpha TS = \Delta t'$ И $\alpha TS' = \Delta t''$", где αTS и $\alpha TS'$ — обозначение входных лингвистических переменных, являющихся мощностями множеств TS и TS'; терм $t' \in T_1 = \{$Маленький, Средний, Большой$\}$, модификатор $\Delta \in M = \{$НЕ, ОЧЕНЬ$\}$.

Заключение ядра (B): нечеткое высказывание вида "$DegCov = \Delta t''$", где $DegCov$ — обозначение выходной лингвистической переменной *степень достоверности гипотезы о полноте покрытия множества терм-спутников;* терм $t'' \in T_2 = \{$Низкая, Средняя, Высокая$\}$; модификатор $\Delta \in M$.

В качестве метода определения количественного значения степени истинности заключения ядра (5) используется метод *min*-активизации.

Коэффициент определенности нечеткой продукции (F): по умолчанию для всех правил нечетких продукций $F = 1$.

Постусловие продукции (H): выполнение процедуры анализа результатов оценки совокупности особей.

Для фаззификации нечетких термов входных лингвистических переменных используются сигмоидальные и колоколообразные функции. Процедура агрегирования нечеткого логического вывода сводится к выбору минимального значения функций принадлежности нечетких переменных. На этапе активизации определяется степень истинности для терма выходной лингвистической переменной *DegCov*. На этапе аккумуляции определяется функция принадлежности выходной лингвистической переменной методом *max*-объединения. Для дефаззификации используется метод центра тяжести.

Таким образом, в результате нечеткого логического вывода определяются значения $F(PS)$ особи. Особи популяции ранжируются в соответствии со значением *Fitness*-функции. Черта подводится над особью, у которой значение *Fitness*-функции $< 0{,}9$. Особи, находящиеся над чертой считаются вошедшими в совокупность особей, составляющих модель системы продукций, и предназначены для оценки их в системе *LogResDed*.

Функция приспособленности совокупности особей. Цель вычисления *Fitness*-функции совокупности особей в данном случае заключается в оценивании корректности работы метода, представленного в виде множества продукционных правил. Для этого сгенерированная и отобранная совокупность особей (модели ядер продукционных правил) подвергается ряду отображений:

- особь как модель утверждения (ядро продукционного правила) из десятичного представления — в естественно-языковое;
- утверждение на естественном языке — на язык логики предикатов первого порядка;
- формула на языке логики предикатов — во множество дизъюнктов.

После этого к ядру продукции добавляются остальные компоненты продукционного правила. Сформированная система продукций подается на вход аппарата активации продукционных правил, основой которого является система логического резолютивного вывода *LodResDed*.

Если в 95 % тестовых примеров система продукций решает поставленную прикладную задачу, то считается, что проблема генерации системы продукций завершена, иначе — генерация должна быть продолжена.

Генетические операторы

Оператор редукции. Текущая популяция подвергается естественному отбору посредством оператора редукции, который реализует три функции:

1) исключение из популяции некорректных особей или ДНК особей, которые могли возникнуть в результате генерации начальной популяции и применения операторов скрещивания и мутации, а также особей, не удовлетворяющих ограничениям задачи;

2) удаление из популяции одинаковых индивидов для избежания вырождения популяции, когда потомки одного индивида с хорошей приспособленностью заполняют популяцию, уменьшая разнообразие генетического материала;

3) ограничение размера популяции в случае, если задан ее максимальный размер, при этом из популяции удаляются особи с наименьшим значением *Fitness*-функции.

Оператор мутации. Мутации будем подвергать молекулы ДНК хромосомы. Цель мутации — улучшить характеристики ДНК. Основные операции мутации:

- упорядочение ДНК в хромосоме;
- удаление одинаковых молекул ДНК за исключением одной;
- добавление молекул ДНК, например при отсутствии служебного слова ТО;

- модификация молекул ДНК — изменение левого или правого терм-спутника при их равенстве; изменение одного из термов, если *x* есть терм-спутник, то *y* должен быть базовым термом и наоборот; изменение правого терма на терм "глагол", так как, если *x* есть терм-спутник *R*, то *y* должен быть базовым термом "глагол", и т. п.

Оператор селекции. Цель оператора селекции — выбрать из множества индивидов популяции $P(t)$ пару особей для выполнения оператора кроссинговера. В алгоритме предлагается использовать метод селекции на основе заданной шкалы. Особи PS_i популяции $P(t)$ ранжируются по значению $F_i(PS)$. В соответствии со стратегией перехода из одной подобласти альтернативных решений в другую, повышающую эффективность поиска оптимального решения, назначение значений шкалы будем осуществлять одновременно сверху вниз и снизу вверх, что обеспечит возможность выбора пары, состоящей из лучшей и худшей особей. Таким образом, для скрещивания выбирается пара особей с одинаковыми значениями шкалы.

Оператор кроссинговера. Оператор скрещивания (кроссинговера) используется для получения новых особей на базе уже имеющихся. Для скрещивания используются двух-, трех- и четырехточечные операторы кроссинговера.

Заключение

Для определения достоверности положений работы проведены вычислительные эксперименты с помощью разработанного программного обеспечения, которое включает интерфейс пользователя для ввода данных и формирования спецификации предметной области прикладной задачи, а также XML-описания метода *UserInterface*; генетический алгоритм *GenAlg*; нечеткий регулятор *FussyRegulator*; систему логического резолютивного вывода *LogicResDeduct*.

Искомый результат был получен в 23-м поколении и имел также место в следующем поколении. Можно утверждать, что качество популяции улучшалось от поколения к поколению ввиду того, что корректные хромосомы появились в популяции только в 4-м поколении, и на протяжении дальнейшей работы алгоритма число искомых хромосом с полностью корректными молекулами ДНК увеличивалось.

Таким образом, описанный в работе способ генерации множества ядер продукций в конечном итоге позволит сформировать библиотеку декларативных методов естественно-языковой обработки научных текстов, которая может быть использована для автоматического построения онтологий предметных областей.

Список литературы

1. **Асанов, А. А.** Исследовано в России [Электронный ресурс]. — http://zhurnal.ape.relarn.ru/articles/2002/155.pdf.
2. **Ларичев, О. И.** Выявление экспертных знаний [Текст] / О. И. Ларичев, А. И. Мечитов, Е. М. Мошкович, и др. — М.: Наука, 1996. — 128 с.
3. **Багудина, Е. Г.** Экономический словарь [Текст] / Е. Г. Багудина, и др.; отв. ред. А. И. Архипов. — М.: Изд-во Проспект, 2005. — 624 с.
4. **Гладков, Л. Д.** Генетические алгоритмы [Текст] / Л. Д. Гладков, В. В. Курейчик, В. М. Курейчик. — М.: Физматлит, 2006. — 320 с.
5. **Асаи, К.** Прикладные нечеткие системы [Текст] / Д. Ватага, С. Иваи и др. — М.: Мир, 1993. — 344 с.

УДК 004.5; 004.8

А. С. Клещев, д-р физ.-мат. наук, зав. отделом,
Институт автоматики и процессов управления
ДВО РАН, г. Владивосток

Роль онтологий в программировании. Часть 1. Аналитика*

Обсуждается ряд направлений в программировании, где использование онтологий может привести к существенному прогрессу. В первой части рассматриваются основные понятия, связанные с онтологиями, и возможности использования онтологий при анализе предметных областей и построении их математических моделей.

Ключевые слова: *онтология, концептуализация, база знаний, разработка программного обеспечения, аналитика, математическая модель предметной области, метаонтология, онтологический анализ.*

Введение

В настоящее время одним из интенсивно изучаемых понятий в области искусственного интеллекта являются *онтологии* [1]. Уже имеется ряд направлений, в которых онтологии играют заметную роль. К ним относятся создание Интернет-порталов на основе онтологий, использование онтологий как средства навигации по большим массивам информации в сети Интернет [2] и как средства интеллектуализации программных агентов [3]. Однако потенциал возможного использования онтологий в программировании значительно шире. Целью данной статьи является обсуждение ряда направлений в программировании, в которых использование онтологий может привести к существенному прогрессу.

* Работа выполнена при финансовой поддержке ДВО РАН в рамках Программы Президиума РАН № 14 "Фундаментальные проблемы информатики и информационных технологий", проект 06-1-П14-051 "Интеллектуальные системы, основанные на многоуровневых моделях предметных областей".

Основные понятия

Под *информацией* в данной работе будет пониматься некоторая совокупность идей. Таким образом, информация рассматривается здесь исключительно как содержательное понятие.

Вербальным представлением назовем отображение конечного множества терминов $t_1, ..., t_m$ в (бесконечное) множество всех возможных значений V. Иными словами, вербальное представление — это таблица, состоящая из двух столбцов и m строк, в первом столбце которой расположены термины $t_1, ..., t_m$, а во втором — их значения из множества V. Таким образом, вербальное представление — это формальный язык с простейшим синтаксисом.

Будем говорить, что информация вербализуема, если она может быть передана адекватно некоторым вербальным представлением. Данное определение устанавливает связь между содержательным и формальным понятиями. Слово "адекватно" означает, что при передаче информации с помощью вербального представления получатель извлечет из вербального представления ту информацию, которую закодировал отправитель.

Примерами вербального представления информации, используемого в повседневной жизни, являются различные анкеты (начальник отдела кадров извлекает из листка по учету кадров ту же информацию, которую в него вложил человек, заполнивший этот листок). Примером вербального представления информации, используемого в математической логике, является функция интерпретации сигнатуры языка исчисления предикатов. Примером вербального представления информации, используемого в программировании, является представление состояния вычислительного процесса в терминах идентификаторов программы и их значений. Последний пример показывает, что на существующих компьютерах может обрабатываться только вербализуемая информация, и эта обработка осуществляется только в вербальном представлении.

Для того чтобы связать с вербальным представлением некоторую семантику, необходимо определить семантику значений (элементов множества V) и конечных совокупностей терминов $t_1, ..., t_m$.

Семантика значений может быть определена следующим образом. Будем считать, что множество всех возможных значений V есть объединение величин. *Величиной* будем называть некоторое множество значений (подмножество множества V), замкнутое относительно конечной совокупности операций, функций и предикатов, определенных для этой величины. Аналогами величин в математике являются алгебраические системы (в том числе и многосортные), а в программировании — типы данных. Можно говорить о стандартных величинах, которым соответствуют предопределенные типы данных языков программирования: размерных величинах (числовых типах данных), скалярных величинах (скалярных типах данных), структурных величинах (типах записей), величинах множеств (типах множеств), величинах отображений (массивах) и величинах последовательностей (последовательных файлов). Нестандартным величинам соответствуют классы (абстрактные типы данных).

Семантика конечной совокупности терминов $t_1, ..., t_m$ может быть точно определена с помощью концептуализации, т. е. множества всех имеющих смысл вербальных представлений с одним и тем же множеством терминов $t_1, ..., t_m$. Иными словами, *концептуализация* — это подмножество множества всех возможных таблиц, все элементы которого обладают содержательным свойством (имеют смысл). Судить о том, обладает ли конкретная таблица этим свойством, могут лишь специалисты, знающие смысл терминов $t_1, ..., t_m$, т. е. семантика конечной совокупности терминов неявно определяется через все возможные осмысленные использования терминов этой совокупности.

Онтология (терминологическая база знаний) — это возможно более точная спецификация концептуализации, т. е. явное представление содержательного свойства "таблица имеет смысл". Если концептуализация мыслится как экстенсиональное задание множества всех осмысленных вербальных представлений с фиксированным множеством терминов, то онтология является интенсиональным заданием этого множества. Она представляет собой явно заданный предикат, который истинен для всех элементов концептуализации и ложен для всех вербальных представлений, не входящих в концептуализацию. Онтология определяет семантику конечной совокупности терминов явно, но приближенно (указанный предикат не всегда может иметь явное точное представление).

Для того чтобы онтология была понятна специалистам, ее обычно строят в виде конечной совокупности определений терминов и онтологических соглашений. Определение термина задает объем понятия, обозначенного этим термином (множество возможных значений этого термина), а онтологические соглашения задают связи между значениями разных терминов в вербальных представлениях.

Онтология не может быть правильной или неправильной, она лишь явно задает смысл совокупности терминов (а о смысле терминов, как известно, не спорят). Однако сама онтология, в свою очередь, должна быть определена в каких-либо терминах. Это может быть некоторый выделенный набор абстрактных терминов (класс, пример, часть, целое, атрибут, значение и т. п.), смысл которых считается известным, множество терминов, связанных с величинами (обозначения констант, операций, функций и предикатов), или множество математических терминов (которые точно определяются в математике с помощью систем аксиом и математических определений).

Знания, выражаемые в терминах $t_1, ..., t_m$, всегда относятся к какому-либо собственному (бесконечному) подмножеству концептуализации, элементы которого обладают некоторым содержательным свойством. Знания об этом подмножестве — это возможно более точная спецификация данного подмножества в терминах концептуализации (онтологии) [4].

Идеи, так или иначе связанные с какой-нибудь профессиональной деятельностью, будем называть *профессиональными*. В множестве всех профессиональных идей можно выделить подмножества, которые называются *предметными областями*. Каждая профессиональная идея относится к одной или нескольким предметным областям. Сложная предметная область может содержать идеи из нескольких более простых предметных областей.

Одной из важнейших характеристик предметной области является используемая в ней совокупность терминов — терминология. *Термин* — это слово или словосочетание, имеющее в предметной области специальный смысл. Терминология однозначно характеризует пред-

метную область. Понимание профессиональных идей, относящихся к предметной области, без понимания смысла ее терминологии невозможно.

С учетом изложенного выше терминология предметной области позволяет говорить о концептуализации и онтологии предметной области. Действительность предметной области — это бесконечное множество вербальных представлений информации о фрагментах реального мира в терминах предметной области, которые имели место в прошлом, имеют место в настоящем, и будут иметь место в будущем. Элементы этого множества будем называть *ситуациями*. Таким образом, действительность предметной области — это собственное подмножество ее концептуализации (можно себе представить информацию о невозможных фрагментах мира, которые не имели места ни в прошлом, ни в настоящем, ни будут иметь места в будущем), все элементы которого обладают указанным в предыдущем предложении содержательным свойством. Знания предметной области — это знания о ее действительности (возможно более точная спецификация множества ситуаций) [5].

Аналитика

Разработка каждой программы (тем более, информационной системы) начинается с анализа той информации о предметной области, которая необходима для проектирования этой программы. С этой деятельностью, называемой *аналитикой*, обычно связаны три группы специалистов:

- эксперты предметной области, которые владеют концептуализацией и системой знаний предметной области, а также могут описать прикладные задачи;
- проектировщики информационной системы, которые нуждаются в понятной им модели предметной области и спецификациях задач в терминах этой модели;
- аналитики, которые должны получить информацию от экспертов и удовлетворить потребности проектировщиков.

В настоящее время существуют два подхода к аналитике. Один из них ориентирован на систему понятий прикладной математики, другой — на систему понятий, лежащую в основе средств реализации.

В рамках подхода, *связанного с прикладной математикой*, аналитик (прикладной математик) выделяет в ситуациях объекты, являющиеся входными данными, результатами решения или промежуточными значениями в прикладных задачах, обозначает их именами вводимых им переменных и ставит им в соответствие математические объекты — числа, множества, функции, предикаты, графы и т. п. Понятиям предметной области ставятся в соответствие те же переменные, для которых определяются области их возможных значений. В этом случае система знаний предметной области чаще всего моделируется некоторой системой математических соотношений между этими переменными, а прикладные задачи сводятся к математическим задачам, сформулированным в терминах этих моделей. Для многих математических задач уже существуют известные методы их решения.

Примерами методов анализа, *связанных с системой понятий*, лежащей в основе средств реализации, являются объектно-ориентированный анализ [6] и инженерия знаний [7]. В первом случае модель предметной области строится в виде набора классов, объектов и методов, во втором — в виде множества правил. И в том, и в другом случае результаты анализа содержат описания методов функционирования разрабатываемой информационной системы.

Онтологии могут быть положены в основу еще одного подхода к аналитике — *онтологического анализа*. Этот подход ориентирован на систему понятий анализируемой предметной области. Модель предметной области строится в виде моделей онтологии и системы знаний этой предметной области; постановки прикладных задач формулируются в терминах модели онтологии. Разработка методов решения этих задач составляет самостоятельный этап анализа. Целью онтологического анализа предметной области является поиск ее концептуализации, построение ее онтологии, а также построение системы знаний в терминах этой концептуализации, явно и наиболее точно описывающей действительность этой предметной области.

Первым этапом онтологического анализа предметной области является поиск ее концептуализации. Эксперты (по просьбе аналитиков) должны сформировать возможно более полный список терминов, используемых для представления действительности, а также представительный список описания ситуаций действительности в этих терминах. Далее аналитики с помощью экспертов пытаются вербально представить ситуации из этого списка, а эксперты определяют, насколько адекватно эти вербальные представления передают информацию о ситуациях. Составляется список уже использованных значений. В процессе этой работы списки терминов, значений и ситуаций могут пополняться. Аналитики фиксируют смысл используемых терминов и значений, а также принципы адекватного представления с их помощью ситуаций. Самостоятельную часть этого этапа составляет анализ списка значений. Каждое значение должно быть отнесено к некоторой величине, стандартной или нестандартной. Составляется список всех использованных величин, отдельно определяются все нестандартные величины. Если все ситуации из списка уже адекватно представлены как элементы концептуализации, величины выделены, а смысл всех терминов и принципы адекватного представления с их помощью ситуаций понятны аналитикам, то можно считать, что этот этап успешно завершен и концептуализация предметной области найдена.

Вторым этапом онтологического анализа предметной области является построение онтологии для найденной концептуализации. Для этого аналитики с помощью экспертов должны построить определения всех терминов концептуализации, используя в этих определениях термины, связанные с величинами, и термины концептуализации, уже получившие определения в онтологии. Поиск ошибок в определениях онтологии может быть выполнен с помощью полученного на предыдущем этапе списка вербальных представлений ситуаций. Если значение некоторого названия понятия в вербальном представлении некоторой ситуации знаний выходит за пределы объема этого понятия, определенного в онтологии, то определение этого понятия неправильно. Наконец, особую трудность представляет формулировка онтологических соглашений. Некоторые онтологические соглашения могут быть предложены экспертами, однако нет никаких надежд на то, что таким образом будут выделены все соглашения. Более систематическим способом их выделения является составление с помощью экспертов списка бессмысленных ситуаций, представленных вер-

бально, но не входящих в концептуализацию. Попытка составления такого списка и обсуждение с экспертами причин, по которым эти ситуации не входят в концептуализацию, могут привести к формулировке новых онтологических соглашений. Если же оказывается, что такая ситуация согласуется с онтологией действительности, то либо некоторые онтологические соглашения должны быть уточнены, либо должны быть сформулированы новые онтологические соглашения.

Третьим этапом онтологического анализа предметной области является построение системы знаний, возможно более точно определяющей действительность. Система знаний строится в той же форме, что и система онтологических соглашений (в терминах онтологии). Заключительная часть этого этапа анализа состоит в поиске ошибок в системе знаний предметной области. Проверяется, согласуется ли каждое вербальное представление ситуации с системой знаний. Затем с помощью экспертов составляется список ситуаций, которые входят в концептуализацию (имеют смысл и, тем самым, согласуются с онтологией), но не входят в действительность (являются невозможными с точки зрения знаний предметной области). Если оказывается, что такие ситуации согласуются с системой знаний предметной области, то эта система знаний должна быть уточнена. Формализация онтологии и системы знаний предметной области завершает построение ее модели.

Сравнение этих трех подходов к аналитике позволяет сделать следующие выводы. Подход, ориентированный на систему понятий прикладной математики, имеет сравнительно ограниченное применение, поскольку во многих предметных областях, для которых разрабатываются информационные системы, не существует традиции использования прикладной математики. В этом отношении подходы, ориентированные на систему понятий, лежащую в основе средств реализации (объектно-ориентированный анализ, инженерия знаний), и на систему понятий предметной области (онтологический анализ), являются более универсальными. Подход, ориентированный на систему понятий, лежащую в основе средств реализации, вынуждает аналитиков выполнять часть работы проектировщиков и навязывать им свои проектные решения, причем проектировщики не имеют возможности изменить эти решения, поскольку не имеют иной информации о предметной области. Понятно, что чем сложнее предметная область, тем интеллектуально сложнее выполнить анализ предметной области в рамках этого подхода. Используя подход, ориентированный на систему понятий предметной области, аналитик занимается лишь вопросами анализа и построения модели предметной области, за счет чего объем его работы по сравнению с предыдущим случаем сокращается и, тем самым, он может выполнить анализ более сложных предметных областей. Проектировщик же в этом случае выполняет всю работу по проектированию, в результате чего объем его работы возрастает. Однако при этом он не связан какими-либо чужими проектными решениями. Отсюда следует, что область применения онтологического анализа — это сложные предметные области или их разделы (обычно связанные с научной деятельностью), в которых нет традиции применения математических методов. Если результат онтологического анализа предметной области — модели онтологии и системы знаний — представляется в математических терминах, то можно усмотреть определенное сходство между онтологическим анализом и подходом, связанным с прикладной математикой (результаты и того, и другого представляются в математических терминах). Разница состоит в том, что в случае анализа, использующего прикладную математику, делается попытка так упростить систему знаний предметной области, чтобы непосредственно построить ее математическую модель, а в случае онтологического анализа системы понятий предметной области (в общем случае без упрощений) строится математическая модель онтологии и затем — математическая модель системы знаний (с использованием модели онтологии) [8].

Если результат онтологического анализа формализуется, то для этого используется какой-либо язык представления онтологий. Разработчики каждого такого языка должны ответить на следующие основные вопросы:

- в каких терминах определяются онтологии (как отмечалось выше, эти термины должны быть понятны проектировщику; это могут быть либо термины технологии программирования, поскольку проектировщик — специалист в этой области, либо математические термины, поскольку предполагается, что проектировщик имеет математическое образование);
- какие свойства онтологий могут быть представлены (для этого нужно знать, какими свойствами обладают реальные онтологии [9]);
- насколько сложность представления онтологии соответствует сложности ее содержания (для языков с фиксированным синтаксисом могут возникнуть случаи, когда простые идеи требуют их громоздкого формального выражения).

По своему назначению языки для представления онтологий можно разделить на языки инструментальных средств (онтологии, представленные на таких языках, предназначены для непосредственного использования в тех или иных программных проектах) [10] и языки для исследования онтологий (онтологии, представленные на таких языках, предназначены для публикации). Языки первой группы обычно испытывают те или иные ограничения реализации, от чего свободны языки второй группы. Примером языка, предназначенного для исследования онтологий, является язык прикладной логики [11]. Этот язык является расширяемым и синтаксически, и семантически, а онтология, представленная на нем, определяется в математических терминах.

Поскольку анализ (в том числе и онтологический) сложных предметных областей является трудоемким и интеллектуально сложным видом деятельности, возможность повторного использования его результатов является весьма желательным свойством. Онтология является повторно используемой, если она может быть использована в проектах разных информационных систем. Важным классом повторно используемых онтологий являются реальные онтологии, которые определяют смысл системы терминов, применяемой специалистами в некоторой реально выполняемой профессиональной деятельности. Публикация таких онтологий делает доступными для проектировщиков информационных систем результаты онтологического анализа соответствующих предметных областей. Примерами реальных онтологий могут служить онтология медицинской диагностики острых заболеваний [12], онтология физической [13] и органической [14] химии (в объеме университетского курса), онтология рентгенофлюоресцентного анализа [15], онто-

логия преобразований программ [16], онтология графического пользовательского интерфейса [17].

Метаонтологией называется онтология, в терминах которой может быть описана другая онтология. Обобщая это определение, можно говорить о метаонтологиях разных уровней. Особый интерес представляют прикладные (не универсальные) метаонтологии. Примером предметной области, имеющей прикладную трехуровневую метаонтологию, является химия. Широкой предметной областью будем называть такую предметную область, для которой онтологии всех ее разделов могут быть описаны в терминах одной и той же прикладной метаонтологии этой области. Примером широкой предметной области является медицинская диагностика. Классом предметных областей будем называть совокупность предметных областей (или их разделов), онтологии которых могут быть описаны в терминах одной и той же прикладной метаонтологии этого класса. Обнаружение прикладных метаонтологий позволяет выявлять внутреннее сходство между внешне различными предметными областями или их разделами. Другие определения сходства онтологий и отношений между ними рассмотрены в [8].

Онтологический анализ, рассмотренный выше, — это процесс построения онтологии по некоторому конечному подмножеству ее концептуализации — списку вербальных представлений ситуаций, т. е. процесс "снизу вверх". Прикладные метаонтологии могут быть использованы как основа другого процесса — метаонтологического анализа, т. е. процесса построения онтологии предметной области или ее раздела "сверху вниз" на основе прикладной метаонтологии (широкой предметной области или класса предметных областей). Таким способом были построены онтологии некоторых разделов химии и медицины.

Список литературы

1. **Uschold M.** Knowledge Level Modeling: Concepts and Terminology // The Knowledge Engineering Review. 1998. Vol. 13. № 1. P. 5—29.
2. **http://www.alphaworks.ibm.com/contentnr/semanticsfaqs**
3. **Wayner P.** Free Agents // Byte. 1995. № 3. P. 105—114.
4. **Клещев А. С., Шалфеева Е. А.** Классификация свойств онтологий. Онтологии и их классификации // Научно-техническая информация. Сер. 2. 2005. № 9. С. 16—22.
5. **Клещев А. С., Артемьева И. Л.** Математические модели онтологий предметных областей // Научно-техническая информация. Сер. 2. 2001. Ч. 1. № 2. С. 20—27; Ч. 2. № 3. С. 19—28; Ч. 3. № 4. С. 10—15.
6. **Буч Г.** Объектно-ориентированный анализ и проектирование. М.: Бином, 1999. 560 с.
7. **Уотермен Д.** Руководство по экспертным системам. М.: Мир, 1989. 388 с.
8. **Клещев А. С., Артемьева И. Л.** Необогащенные системы логических соотношений. Ч. 2 // Научно-техническая информация. Сер. 2. 2000. № 8. С. 8—18.
9. **Шалфеева Е. А.** Классификация свойств онтологий. Свойства онтологий и их классификации // Научно-техническая информация. Сер. 2. 2005. № 11. С. 9—16.
10. **Corcho O., Gómez-Pérez A.** A Roadmap to Ontology Specification Languages. — http:// www.cs.man.ac.uk/ ~ocorcho/ documents/ ekaw00_CorchoGomezPerez.pdf
11. **Клещев А. С., Артемьева И. Л.** Необогащенные системы логических соотношений. Ч.1 // Научно-техническая информация. Сер. 2. 2000. № 7. С. 18—28.
12. **Клещев А. С., Москаленко Ф. М., Черняховская М. Ю.** Модель онтологии предметной области "Медицинская диагностика" // Научно-техническая информация. Сер. 2. Ч. 1. 2005. № 12. С. 1—7; Ч. 2. 2006. № 2. С. 19—30.
13. **Артемьева И. Л., Цветников В. А.** Фрагмент онтологии физической химии и его модель // Электронный журнал "Исследовано в России". 2002. № 3. С. 454—474, (http://zhurnal.ape.relarn.ru/ articles/ 2002/042.pdf).
14. **Артемьева И. Л., Высоцкий В. И., Рештаненко Н. В.** Модель онтологии предметной области (на примере органической химии) // Научно-техническая информация. Сер. 2. 2005. № 8. С. 19—27.
15. **Артемьева И. Л., Мирошниченко Н. Л.** Модель онтологии рентгенофлуоресцентного анализа // Информатика и системы управления. 2005. № 2. С. 78—88.
16. **Князева М. А., Купневич О. А.** Модель онтологии предметной области "Оптимизация последовательных программ" // Научно-техническая информация. Сер. 2. 2005. Ч. 1. № 2. С. 17—21; Ч. 2. № 4. С. 14—22.
17. **Грибова В. В., Тарасов А. В.** Модель онтологии предметной области "Графический пользовательский интерфейс" // Информатика и системы управления. 2005. № 1. С. 80—90.

УДК 004.056.55

В. В. Коробицын, канд. физ.-мат. наук, доц., **С. С. Ильин,** студент,
Омский государственный университет им. Ф. М. Достоевского

Реализация симметричного шифрования по алгоритму ГОСТ-28147 на графическом процессоре

Предлагаются реализации симметричного шифрования алгоритма ГОСТ-28147 с использованием прикладных интерфейсов DirectX и OpenGL. Описывается подход к организации процесса вычислений и приводятся несколько способов реализации базовых операций алгоритма. Представлены результаты вычислительных экспериментов на различных видеокартах, подтверждающие применимость данной технологии для организации системы потокового шифрования с высокой скоростью.

Ключевые слова: *симметричное шифрование, параллельная обработка данных, графические процессоры, пиксельные шейдеры.*

Введение

Эффективность графических процессоров (GPU) уже была показана при решении ряда задач, но до недавнего времени реализация криптографических алгоритмов на GPU оставалась в стороне. С появлением более общих моделей программирования для GPU решение многих задач перестало быть невозможным или неэффективным. В частности, алгоритм шифрования AES [1] уже был реализован на GPU [2, 3, 4]. Мы предла-

гаем способы реализации алгоритма симметричного шифрования ГОСТ-28147 [5] на современных GPU с использованием графических систем OpenGL 2.1 [6] и Direct3D 9.0c [7].

Графические процессоры в последнее время все чаще используются для решения задач, которые позволяют вести параллельную обработку. Вычислительные мощности видеокарт последнего поколения превосходят производительность центральных процессоров (CPU), и темпы их роста также более высокие, чем у CPU. Это связано с более удачными архитектурными решениями GPU. Они обладают распределенной структурой с большим числом однотипных вычислительных блоков, способных работать параллельно. Похожая тенденция заметна и в развитии современных CPU с той лишь разницей, что число ядер в центральных процессорах варьируется от 2 до 8, а у современных видеокарт число процессоров достигает 320.

Но наличие большого вычислительного потенциала само по себе ничего не дает. Необходимы средства для его использования и приспосабливания к решению разного рода задач. Такими средствами являются интерфейсы прикладного программирования (API) для GPU. Особенно важна возможность запуска на GPU собственных программ, которая появилась с введением пиксельных и вершинных шейдеров — программ-алгоритмов, которые определяют, каким образом графическому процессору следует вести вычисления. До их появления работа GPU контролировалась путем вызова функций, которые позволяли задавать текущий режим обработки входных данных, в явном виде использовать инструкции процессоров GPU было нельзя.

Важным аспектом при проведении вычислений на GPU является доставка входных данных на видеокарту и считывание результатов вычислений. Теоретическая пропускная способность интерфейса PCI-E 16x (интерфейса современных видеокарт) составляет 6 Гбайт/с [8]. В действительности для видеокарт класса GeForce 6800 и 7800 скорость отправки данных не превышает 1,5 Гбайт/с [9], скорость считывания — 0,7 Гбайт/с, что определяет достижимую максимальную скорость обмена данными с видеокартой, равную 0,47 Гбайт/с.

Подходы к реализации алгоритма ГОСТ-28147

Стандарт предусматривает три режима использования алгоритма, но только один из них подходит для реализации на GPU — это режим ECB [10]. Использование других режимов подразумевает последовательную обработку данных, что не позволит в полной мере воспользоваться вычислительной мощностью GPU. На базе режима ECB

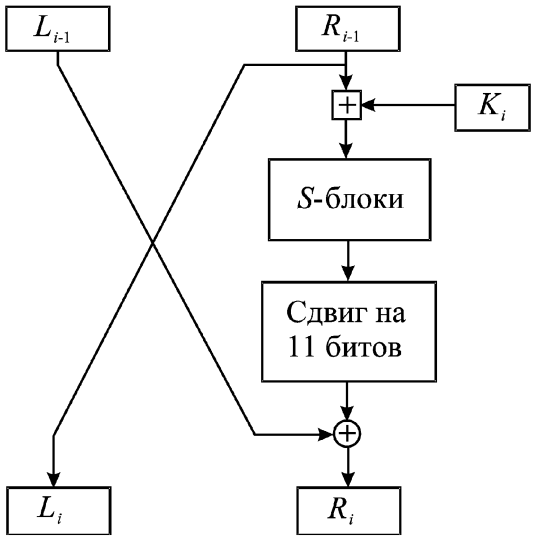

Рис. 1. Схема базовой функции алгоритма ГОСТ-28147

возможна реализация более сложных режимов работы, таких как CTR [10] (режим счетчика) или CWC [11], позволяющих вести параллельную обработку данных.

Из схемы на рис. 1 видно, что для реализации алгоритма ГОСТ-28147 на GPU необходимо осуществить следующие операции:

1. Доставка на видеокарту L_{i-1} и R_{i-1};
2. Доставка на видеокарту K_i для каждого раунда;
3. Сложение по модулю 2^{32} (на схеме знак +);
4. Подстановка с использованием S-блоков;
5. Циклический сдвиг влево на 11 битов;
6. Выполнение XOR для 32 битов (на схеме знак $\oplus$).

Мы применяли два графических API для реализации одной и той же функциональности: с использованием DirectX 9.0c (рис. 2) и OpenGL (рис. 3).

В обеих схемах сначала происходит заполнение двух входных текстур TexR и TexL (использовались квадратные текстуры размера $N \times N$ пикселей и имели целочисленный 32-битный формат RGBA, где N варьировалось от 32 до 1024) значениями старших и младших частей 64-битных блоков входных данных.

Вместе с исходными данными на видеокарту отсылаются данные о треугольниках, которые необходимо нарисовать. В нашем случае это были два треугольника, покрывающие квадрат $N \times N$ пикселей, что при отрисовке приводило к формированию $N \times N$ фрагментов. Для каждой вершины треугольника указывались текстурные координаты для чтения из входных текстур TexR и TexL таким образом, что при их интерполяции каждому фрагменту соответствовала точка текстуры, расположенная в ней так же, как сам фрагмент в выходном буфере. Это дает возможность использовать входные текстуры в качестве выходных буферов и

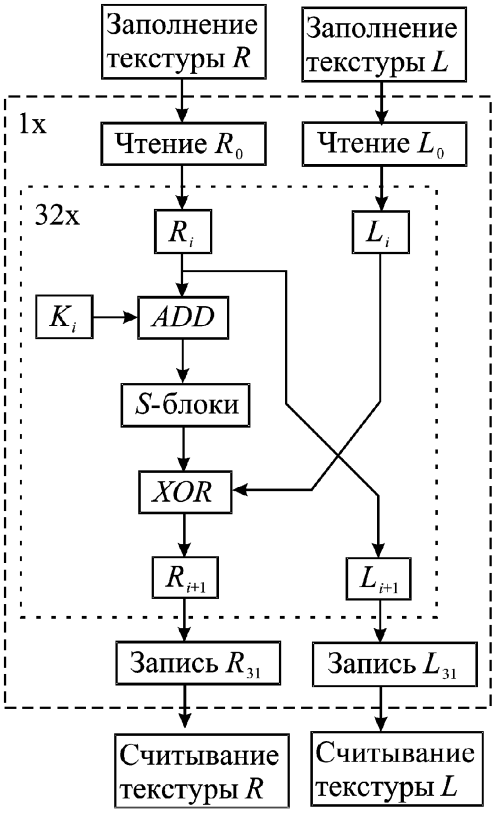

Рис. 2. Схема реализации алгоритма с использованием Direct3D

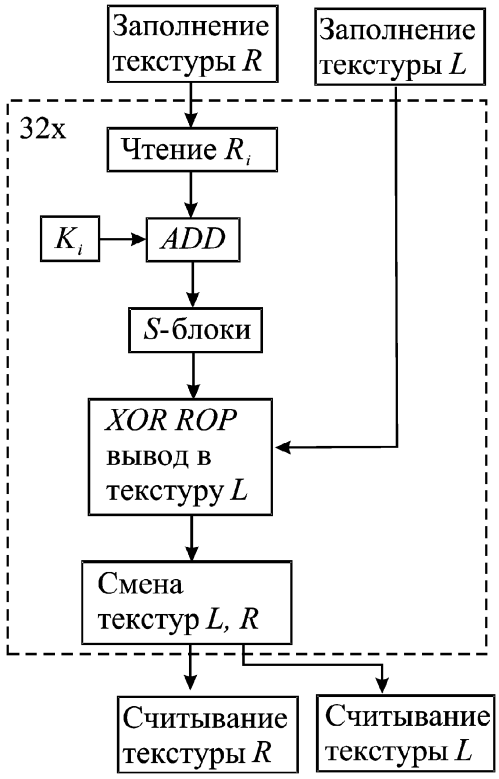

Рис. 3. Схема реализации алгоритма с использованием OpenGL

обеспечивает простое отображение "входные-выходные данные". Помимо передачи этой информации необходимо иметь доступ к информации о подключах K_i во время выполнения шейдера.

В варианте с OpenGL используется возможность выполнения логических операций на последнем этапе цикла отрисовки. Взглянув на схему алгоритма ГОСТ, можно заметить, что для реализации одного раунда достаточно выполнить операции сложения, замены с *S*-блоками и циклического сдвига для всех элементов входной текстуры TexR, после чего провести запись полученных вычислений в режиме XOR в выходной буфер, расположенный в текстуре TexL и "поменять" текстуры местами. Для реализации ГОСТ необходимо выполнить данную процедуру 32 раза. Просто нарисовать 32 квадрата за один раз не получится — каждый раз необходимо использовать разные выходные буферы и разные входные текстуры, приходится 32 раза проводить отрисовку, каждый раз меняя входную текстуру и выходной буфер. Информация о подключах K_i в данной схеме поставлялась в качестве одного дополнительно атрибута у входных вершин квадрата. Для всех вершин одного квадрата этот атрибут имел одинаковое значение.

В схеме с Direct3D шифрование данных входных текстур TexR, TexL проводилось путем однократной отрисовки квадрата. В DirectX 9.0c нет возможности выполнять операцию XOR ни в самих пиксельных тендерах, ни на последнем этапе отрисовки, поэтому данную операцию приходится реализовывать с использованием текстур, в которых хранятся заранее просчитанные результаты XOR. Для каждого обрабатываемого фрагмента операции выполняются в следующем порядке:

- считывание данных из текстур TexR,TexL;
- выполнение операций базовой функции ГОСТ (32 раза);
- запись выходных данных в текстуры TexR, TexL.

Для данной схемы необходимо использовать два выходных буфера. Информация о подключах K_i хранилась в константах пиксельного шейдера (32 четырехкомпонентных вектора). Установку констант достаточно выполнить один раз для всего набора данных, которые шифруются одним ключом, после чего значения этих констант можно считывать в процессе выполнения шейдера.

Реализация базовых операций алгоритма ГОСТ-28147

Операции, используемые для обеих схем реализации ГОСТ на GPU:

- сложение по модулю 2^{32};
- подстановка с использованием *S*-блоков;
- циклический сдвиг влево на 11 бит.

Также будет рассмотрена "эмуляция" операции XOR для 32-битных данных.

Несмотря на свою простоту эти операции непросто выполнить напрямую в связи с отсутствием поддержки операций с целыми числами в используемых API. Входные данные для операций — два 32-битных числа в форме двух четырехкомпонентных вектора со значениями компонент в интервале [0; 1].

Для реализации операции сложения по модулю 2^{32} используется преобразование входных четырехкомпонентных векторов в двухкомпонентные, описывающие по 16 бит целого числа. После этого выполняется сложение двухкомпонентных векторов. Делается проверка на переполнение (превышение 256). Затем осуществляется обратная распаковка векторов. "Упаковка" в двухкомпонентные векторы позволяет избежать выполнения трех дорогостоящих условных операций *if/then* для учета бита переноса при сложении четырехкомпонентных векторов.

Для обеспечения подстановки с использованием *S*-блоков необходимо заменить восемь четырехбитных *S*-блоков на четыре восьмибитовых *S*-блока. Тогда применение восьмибитового *S*-блока к компоненте входного вектора можно реализовывать считыванием из однокомпонентной одномерной текстуры с координатой, равной значению преобразуемой компоненты. Такой вариант дает хорошие результаты, но их можно немного улучшить, преобразуя одномерную текстуру 1 × 256 в двумерную 16 × 16 и для считывания значения используя текстурные координаты (x * 16, x) * 0,998, где x — значение, к которому необходимо применить *S*-блок. При этом необходимо выставить "wrap"-режим адресации текстуры, при котором берется дробная часть значений текстурных координат. Умножение на 0,998 требуется для устранения ошибок при переводе текстурных координат в индекс текселя, подлежащего считыванию. Всего понадобится четыре таких текстуры.

Есть еще один вариант — индексировать две двухкомпонентных текстуры 256 × 256 и использовать значения сразу двух компонент входного вектора — одну для координаты x, другую для y, и считывать сразу два значения — результат применения двух восьмибитовых *S*-блоков. Однако несмотря на меньшее число текстур и операций считывания из них возникают проблемы с кэшированием текстурных данных, и производительность оказывается значительно ниже.

Операция циклического сдвига влево на 11 бит также реализуется с помощью арифметических операций над числами с плавающей точкой. Сдвиг на 8 бит реализуется циклической заменой компонент вектора *rgba* → *argb*. Сдвиг на оставшиеся три бита осуществляется делением на число 8. А учет дробной части позволяет определить старшие биты после сдвига.

Операцию XOR необходимо реализовывать при работе по схеме Direct3D. Эффективная реализация XOR с помощью вычислительных операций, доступных в DirectX 9.0c, не представляется возможной. Единственный вариант — использование выборок из текстур.

Первый подход заключается в использовании однокомпонентной текстуры размером 256 × 256 точек, в которой хранились бы результаты операции XOR для двух восьмибитовых значений. Данный подход был реализован, но из-за нарушения механизмов кэширования скорость работы была неприемлемой. Падение производительности было даже более заметным, чем при аналогичной ситуации с *S*-блоками.

Такая проблема уже возникала при реализации алгоритма AES. Тогда было предложено использовать текстуру 16 × 16, хранящую результаты выполнения четырехбитного XOR. А для реализации 32-битового XOR достаточно выполнить восемь считываний из такой текстуры и одну операцию сложения для упаковки полученных младших и старших четырехбитных результатов XOR в восьмибитовые компоненты результирующего вектора. Данный подход не приводит к нарушению механизмов работы кэш-памяти (вся текстура 16 × 16 в ней помещается), а выполнение считывания из такой текстуры требует одного цикла TMU (блок выборки текстур). Кроме того, потребуется всего одна такая текстура для реализации всех восьми выборок.

Совмещение базовых операций

Были реализованы два варианта совмещения операций в одну неделимую операцию.

Реализация *S*-блоков + циклический сдвиг влево на 11 бит. Для совмещения данных операций, достаточно в текстурах для реализации *S*-блоков хранить не просто результат применения *S*-блоков, а использовать двухкомпонентные текстуры для хранения результатов применения *S*-блока, сдвинутых влево на 3 бит — в одной компоненте, и вправо на 5 бит — в другой. Для этого потребуются текстуры вдвое большего объема, и это увеличение объема потенциально способно нарушить работу механизмов кэширования, но при проведении вычислительных экспериментов использование данного подхода привело к более высоким результатам, ведь в отношении *S*-блоков ничего не изменяется, а вот операций, необходимых для циклического сдвига, можно полностью избежать.

Сложение по модулю 2^{32} + реализация *S*-блоков + циклический сдвиг влево на 11 бит. Подобное совмещение очень похоже на только что описанное, с той разницей, что текстуры становятся четырехкомпонентными, имеющими размеры 1×512. Две компоненты служат для реализации *S*-блоков одновременно с циклическим сдвигом, а еще в одной компоненте хранится "бит переноса", для реализации операции сложения, четвертая компонента не используется.

Входные векторы покомпонентно складываются, давая нам сумму без учета переносов. После этого проводится последовательная обработка частей данного вектора от младших к старшим. Биты переноса считываются из текстуры по очереди для каждой из четырех компонент вектора (текстура адресуется значением компоненты вектора, деленной на два), одновременно с ними происходит считывание результатов применения *S*-блока к текущей компоненте. После получения "бита переноса" происходит корректировка значения следующей компоненты.

Такой подход позволяет избежать дополнительных операций, необходимых для реализации учета переносов при сложении, но требует для реализации текстуры вчетверо большего объема по сравнению с предыдущим способом совмещения. При проведении экспериментов на некоторых видеокартах это привело к серьезному падению производительности.

Результаты вычислительных экспериментов

Для проведения вычислительных экспериментов использовались четыре компьютера с видеокартами: GeForce 6800 GO, GeForce 7800 GT, GeForce 8600 GT, GeForce 8800 GTS и центральными процессорами: Pentium-M 1.6, Pentium-4 620, Core2 6300, Core 2 6600 соответственно. Характеристики указаны в табл. 1.

Тестирование проводилось путем многократного выполнения цикла шифрования. Перед каждым циклом входные текстуры заполнялись произвольными данными, после чего выполнялось собственно шифрование и считывание результатов. Все входные данные для всех циклов подготавливались заранее и хранились в оперативной памяти, а затем последовательно загружались в текстуры.

Результаты тестирования систем приведены в табл. 2. Помимо скорости шифрования указана пропускная способность для "холостого" цикла, состоящего из операций записи входных данных и считывания их обратно (RT — round trip). Этот показатель очень важен, так как он является верхней границей достижимого быстродействия и может стать существенным лимитирующим фактором. При проведении экспериментов влияние данного фактора приводило к снижению результирующей скорости шифрования в 1,5—2 раза. Если же уменьшить число раундов в ГОСТе до 24 или до 16, то это влияние станет еще более значимым.

Таблица 1

Характеристики тестируемых систем

Характеристика	6800 GO	7800 GT	8600 GT	8800 GTS
Частота ядра, МГц	370	430	540	600
Частота памяти, МГц	800	1000	1200	1600
Частота шейдерного домена, МГц	370	430	1200	1200
Разрядность шины памяти, бит	256	256	128	320
Пропускная способность памяти, Мбайт/с	25 600	32 000	22 400	64 000
Скорость текстурирования, Мтекс/с	4400	8600	8600	24 000
Скорость отрисовки, Мпикс/с	4400	6400	4300	10 000
Число пиксельных процессоров	12	20	32	96
Число блоков выборки текстур	12	20	16	24
Число блоков ROP	12	16	8	20
CPU	PM1.6	P4 620	Core2 6300	Core2 6600
Тактовая частота процессора, МГц	1600	3200	2500 × 2	3000 × 2

Таблица 2

Средняя скорость шифрования тестируемых систем

Реализация алгоритма	6800 GO	7800 GT	8600 GT	8800 GTS
Direct3D RT, Мбайт/с	390	415	200	660
Direct3D с совмещением, Мбайт/с	38	80	43	95
Direct3D, Мбайт/с	33	—	56	130
OpenGL RT, Мбайт/с	396	420	215	1300
OpenGL с совмещением, Мбайт/с	72	107	60	175
OpenGL, Мбайт/с	44	—	75	280
CPU	PM 1.6	P4 620	Core2 6200	Core2 6600
CPU, Мбайт/с	18	38	38	43
CPU потенциальное, Мбайт/с	30	60	120	120

В этой таблице также можно увидеть, что результаты для видеокарт GeForce 8600 GT и GeForce 8800 с применением совмещения операций оказываются ниже, чем без применения совмещенных операций. Для видеокарт GeForce 6800 GO и GeForce 7800 GT ситуация меняется на противоположную. Скорее всего это связано с тем, что в видеокартах GeForce серии 8 используются другие схемы кеширования данных.

Необходимо заметить, что для шифрования на CPU использовалась собственная реализация ГОСТ, в которой отсутствуют оптимизации под современные наборы команд SIMD (SSE, SSE2) и поддержка многоядерности. Для коммерческих реализаций, в которых все это есть, скорости шифрования в 1,5—2 раза выше.

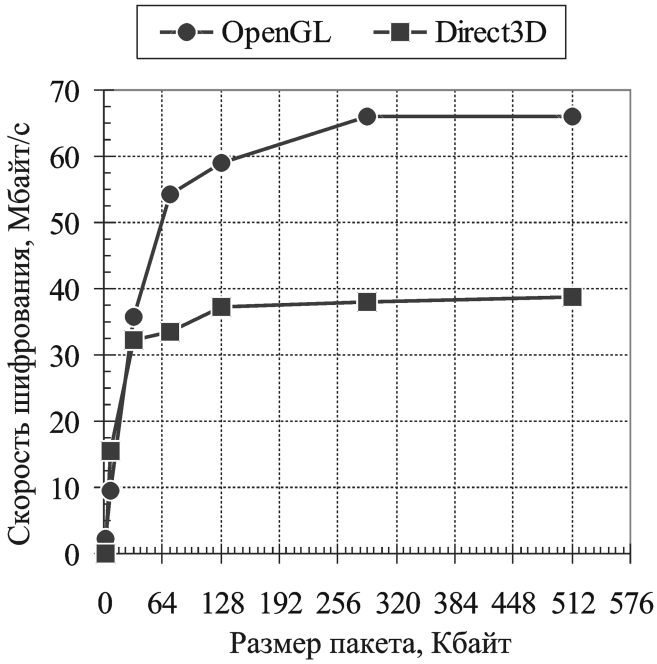

Рис. 4. Зависимость скорости шифрования от размера входного пакета

На рис. 4 приведена зависимость скорости шифрования от размера обрабатываемых пакетов. Пакет в данном случае — это данные, которые отправляются на видеокарту, подвергаются там шифрации и сразу же считываются обратно. Для проведения тестов использовались текстуры с размерами, кратными 32 × 32 (32 × 32, 64 × 64, 96 × 96, 128 × 128, 192 × 192 и 256 × 256). Видно, что оптимальным является использование текстур с размерами, равными или больше 128 × 128 (размер пакета при этом равен 128 Кбайт). Использование текстур меньшего размера приводит к дополнительным затратам ресурсов CPU вследствие необходимости проводить больше вызовов функций графического API. Использование текстур большего размера не дает прироста, а на деле приводит к более высокому уровню загруженности CPU.

Выводы

Предложенные варианты реализации симметричного шифрования алгоритма ГОСТ-28147 на графическом процессоре могут успешно использоваться в потоковом режиме шифрования для обработки больших (более 128 Кбайт) пакетов данных. Средняя скорость шифрования на графическом процессоре вдвое больше, чем на центральном процессоре. Для видеокарты GeForce 8800 GTS достигается скорость шифрования 2 Гбит/с.

Список литературы

1. **Announcing** Approval of Federal Information Processing Standard (FIPS) 197. Advanced Encryption Standard (AES). Federal Register. 2001. Vol. 66. N 235. 12—06. — Notice. P. 63369—63371.

2. **Cook D., Keromytis A.** CryptoGraphics // Exploiting Graphics Cards For Security. Berlin: Springer-Verlag, 2006. 139 p.

3. **Harrison O., Waldron J.** AES Encryption Implementation and Analysis on Commodity Graphics Processing Units // 9th Workshop on Cryptographic Hardware and Embedded Systems (CHES 2007), Berlin, Heidelberg: Springer-Verlag, 2007. LNCS Vol. 4727. P. 209—226.

4. **Rosenberg U.** Using Graphic Processing Unit in Block Cipher Calculations: Master's Thesis. Tartu: University of Tartu, 2007. 48 p.

5. **ГОСТ 28147—89.** Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. Введ. 1990-01-01. М.: Изд-во стандартов, 1989.

6. **Brown P.** The OpenGL Graphics System: A Specification (Version 2.1). 2006-06-30. Silicon Graphics, 2006. 380 p.

7. **The DirectX** Software Development Kit Documentation. Microsoft Press, 2006.

8. **Kilgariff E., Fernando R.** The GeForce 6 Series GPU Architecture // GPU Gems 2: Programming Techniques for High-Performance Graphics and General-Purpose Computation / Eds. Pharr M., Fernando R. Addison Wesley Professional, 2005. P. 471—491.

9. **Elhassan I.** Fast Texture Downloads and Readbacks using Pixel Buffer Objects in OpenGL. Technical Brief. NVIDIA, 2005. 9 p.

10. **Шнайер Б.** Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си. М.: Триумф, 2002. 816 с.

11. **Kohno T., Viega J., Whiting D.** CWC: A high-performance conventional authenticated encryption mode // Fast Software Encryption: 11th International Workshop, 2004. Berlin, Heidelberg: Springer-Verlag, 2004. LNCS Vol. 3017. P. 408—426.

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ В ЭКОНОМИКЕ И УПРАВЛЕНИИ

УДК 007; 004.3

В. В. Сафронов[1)], д-р техн. наук, проф.,
И. В. Григорьев[2)], **А. Н. Попов**[2)],
А. В. Ткачук[2)]
[1)] ОАО "КБ Электроприбор", г. Саратов;
[2)] Военный институт, г. Саратов

Решение задач совершенствования системы образования с использованием методов ранжирования

Рассматриваются некоторые задачи, возникающие в процессе совершенствования системы образования. Они сводятся к многокритериальным и многовекторным задачам ранжирования. На примере задачи подбора кандидатов на должности руководящего и преподавательского состава предлагается метод решения. Рассмотрен численный пример.

Ключевые слова: *совершенствование системы образования, многовекторные задачи ранжирования, подбор руководящего состава.*

Введение

Информационные технологии, методы принятия решений находят все большее применение в образовании. Высшая школа, в частности, и система образования страны в целом претерпевают серьезные изменения, которые обусловлены как внешними, так и внутренними факторами. В плане решения проблем реформирования системы образования возможны постановки следующих задач.

1. Известны n систем образования, каждая из которых характеризуется m ($m \geqslant 2$) критериями, например, объем знаний, время обучения, стоимость обучения и т. д.

Необходимо: выбрать лучшую по совокупности показателей качества систему образования, либо расположить рассматриваемые системы в порядке убывания приоритета (построить кортеж Парето).

2. Проводится анализ деятельности n вузов по m ($m \geqslant 2$) следующим критериям:

процент преподавателей, имеющих ученые степени доктора и кандидата наук;

качество научных исследований;

состояние учебно-лабораторной базы;

уровень подготовки студентов (выпускников);

востребованность выпускников на рынке труда и т. д.

Необходимо: расположить вузы в порядке убывания приоритета по совокупности показателей качества.

3. Известны n активных методов обучения, каждый из которых характеризуется m ($m \geqslant 2$) критериями:

качество и быстрота усвоения учебного материала;

количество сообщаемой информации за определенное время:

сложность применения;

стоимость и т. д.

Необходимо: выбрать лучший, по совокупности показателей качества, активный метод обучения, либо расположить рассматриваемые методы в порядке убывания приоритета.

4. Осуществляется подбор руководящего и преподавательского состава (РПС) в высшее учебное заведение или в образовательное учреждение дополнительного профессионального образования (ДПО). Все множество критериев разбито на четыре векторные компоненты. В свою очередь, каждая из векторных компонент характеризуется множеством скалярных критериев.

Необходимо: расположить кандидатов на должности в порядке убывания приоритета по совокупности векторных компонент. Это позволит лицу, принимающему решение (ректору, декану, заведующему кафедрой и т. д.), сделать обоснованный выбор.

Рассмотрим более подробно четвертую задачу, поскольку в математическом плане первые три задачи более простые. Для решения четвертой задачи необходимо:

разработать информационные модели ("паспорта") должностей руководящего и преподавательского состава;

на основе применения методов теории принятия решений осуществить обоснованный подбор кандидатов и расположить их в порядке убывания приоритета по совокупности критериев;

выработать практические рекомендации лицу, принимающему решение о назначении на должности РПС ВУЗа.

1. Разработка "паспорта" должностей руководящего и преподавательского состава

Анализ содержания деятельности руководящих и преподавательских кадров показывает, что среди должностей руководящего и преподаватель-

ского состава целесообразно выделить три категории. Их характер можно отразить следующими терминами: руководитель, руководитель-преподаватель и преподаватель.

К категории *руководитель* относятся должности, которые имеют явно выраженные функции руководителя, предполагающие непосредственное руководство многоуровневым коллективом, а также выполняющие функции преподавателя, но они явно не обозначены и реализуются частично в определенных условиях. К такой категории относят должности проректоров, деканов.

Руководитель-преподаватель имеет явно выраженные функции руководителя, но непосредственно подчиненный ему коллектив имеет значительно меньшее число сотрудников, значительно однороднее по составу и ориентирован на решение задач по подготовке специалистов в соответствии с особенностями учебного заведения. Кроме того, у руководителя-преподавателя более явно обозначены функции преподавателя. В рассматриваемом составе должностей к ним относятся заведующие кафедрой, заместители заведующего кафедрой.

Остальные из рассматриваемых должностей относятся к категории *преподаватель.* Эти должности предполагают явно выраженные функции преподавателя, а функции руководителя четко не обозначены. К подобным должностям относятся: профессора, доценты, старшие преподаватели, преподаватели, ассистенты.

Под *информационной моделью* ("паспортом") *должности* понимается совокупность векторных и скалярных критериев, характеризующих категории должностей и коэффициенты важности векторных и скалярных компонент (групп качеств и самих качеств). Векторные компоненты — это профессиональные, деловые, морально-психологические и организаторские качества (группы качеств). Каждая из векторных компонент включает несколько скалярных компонент.

Методические особенности этапов разработки информационных моделей ("паспортов") должностей РПС заключаются в следующем.

На первом этапе в качестве основных исходных данных используют частные перечни, полученные из анализа официальных источников. Результатом первого этапа является формирование обобщенного перечня качеств. Оценки экспертов были основаны, в частности, на анализе мнений авторов 83 литературных источников.

Второй этап формирования итогового перечня качеств для оценки и выбора кандидатов на должности РПС базируется на экспертном оценивании соответствия обобщенного перечня качеств требованию объективности и реализуемости процедуры подбора кандидатов на должности руководящего и преподавательского состава.

Третий, заключительный, этап разработки информационных моделей ("паспортов") должностей РПС основан на экспертном опросе, в результате которого оценивается значимость каждой векторной компоненты и значимость скалярных компонент. В состав экспертной группы привлекалось 20 специалистов из числа профессорско-преподавательского состава и руководителей вуза.

После определения значений коэффициентов важности была определена степень согласованности мнений всех экспертов по оценке важности критериев. В качестве меры согласованности выступал коэффициент конкордации [1]. В нашем случае коэффициент конкордации изменялся в диапазоне от 0,83 до 0,88. Это означает, что мнения экспертов хорошо согласованы. В результате получены информационные модели ("паспорта") должностей руководящего и преподавательского состава (см. таблицу).

Информационные модели ("паспорта") должностей руководящего и преподавательского состава ВУЗа

№ качества	Наименование качеств и их групп	Коэффициенты значимости качеств и групп качеств паспортов должностей:		
		руководителя	руководителя-преподавателя	преподавателя
	Профессиональные качества	0,27	0,26	0,26
1	Профессиональные знания	0,090	0,089	0,085
2	Профессиональные умения и навыки	0,061	0,060	0,046
3	Знание руководящих документов, умение с ними работать	0,065	0,069	0,067
4	Работоспособность и целеустремленность	0,104	0,104	0,103
5	Умение реализовать профессиональный опыт на занимаемой должности	0,032	0,031	0,031
6	Умение убеждать	0,024	0,030	0,051
7	Умение оперативно анализировать, обобщать информацию	0,144	0,142	0,143
8	Умение оперативно формировать мысли, вырабатывать рациональные решения	0,168	0,167	0,166
9	Умение четко и ясно доводить информацию, ставить задачи	0,132	0,131	0,130
10	Умение ориентироваться в сложной обстановке, быстро реагировать на изменение обстановки	0,180	0,177	0,178

№ качества	Наименование качеств и их групп	Коэффициенты значимости качеств и групп качеств паспортов должностей:		
		руководителя	руководителя-преподавателя	преподавателя
	Деловые качества	0,24	0,24	0,25
1	Организованность и собранность	0,115	0,115	0,140
2	Ответственность и исполнительность	0,105	0,104	0,114
3	Инициатива и предприимчивость	0,060	0,065	0,080
4	Самостоятельность решений и действий	0,142	0,140	0,098
5	Руководство подчиненными	0,178	0,177	0,087
6	Качество конечного результата	0,168	0,167	0,176
7	Стремление к повышению своей квалификации	0,051	0,057	0,085
8	Способность аналитически мыслить	0,081	0,083	0,088
9	Умение видеть главное, освободиться от текучки	0,075	0,068	0,065
10	Умение убеждать, отстаивать новое, передовое	0,025	0,024	0,067
	Морально-психологические качества	0,25	0,25	0,25
1	Гуманность, доброжелательность	0,179	0,177	0,177
2	Способность к самооценке, самокритичность	0,032	0,031	0,097
3	Тактичность, этика поведения	0,024	0,025	0,103
4	Дисциплинированность	0,114	0,108	0,091
5	Справедливость, честность, принципиальность	0,105	0,105	0,105
6	Лидерство	0,061	0,060	0,052
7	Общительность	0,123	0,131	0,099
8	Внутренняя уверенность, оптимизм	0,168	0,167	0,159
9	Выдержка, самообладание	0,068	0,070	0,023
10	Решительность и твердость	0,126	0,126	0,094
	Организаторские способности	0,24	0,25	0,24
1	Умение воспитывать подчиненных	0,165	0,163	0,162
2	Умение увлечь подчиненных, мобилизовать на решение поставленных задач	0,177	0,160	0,087
3	Умение направлять и контролировать работу подчиненных	0,166	0,168	0,159
4	Последовательность и настойчивость при проведении решений в жизнь	0,068	0,067	0,065
5	Способность доверять подчиненным, использовать их инициативу	0,027	0,024	0,136
6	Умение сплотить коллектив	0,060	0,059	0,051
7	Авторитетность руководителя	0,128	0,129	0,120
8	Способность понять, оценить и спрогнозировать позицию и поведение подчиненных	0,094	0,095	0,093
9	Умение выбирать стиль руководства в соответствии с обстановкой и характером коллектива	0,085	0,104	0,103
10	Умение взаимодействовать со смежными организациями и учреждениями	0,030	0,031	0,024

2. Математическая постановка и метод решения задачи многовекторного ранжирования

Допустим, известны:

множество $S = \{S_\alpha, \alpha = \overline{1, n}\}$ — множество кандидатов на должности РПС — множество систем;

$K(S_\alpha) = \{K_q(S_\alpha), q = \overline{1, r}\}$ — множество векторных компонент $K_q(S_\alpha)$, характеризующих кандидата $S_\alpha \in S$;

$K_q(S_\alpha) = \{K_{qi}(S_\alpha), i = \overline{1, r_q}\}$ — множество скалярных компонент, характеризующих кандидата $S_\alpha \in S$.

Требуется найти упорядоченное множество P эффективных систем (кортеж Парето [2]), для элементов $S_p^0 \in P$ которого справедливо

$$K(S_p^0) = \min_{S_\alpha \in S_D} K(S_\alpha), \qquad (1)$$

где $P = \{S_{k_1}^0, S_{k_2}^0, ..., S_{k_{n^\pi}}^0\}$ — упорядоченное множество эффективных систем. Элементы кортежа ранжированы в соответствии с решающими правилами так, что выполняется условие

$$S_{k_1}^0 > S_{k_2}^0 > ... > S_{k_{n^\pi}}^0,$$

где ">" — знак отношения доминирования, $k_i \in \{\overline{1, n^\pi}\}$. Длина кортежа равна n^π;

$S_D \subseteq S$ — допустимое множество, для элементов S_α которого выполняются условия

$$D_i(S_\alpha) \leqslant D_i^0, \quad i = \overline{1, M}, \qquad (2)$$

где $D(S_\alpha) = \{D_i(S_\alpha), i = \overline{1, M}\}$ — множество требований, которым должен удовлетворять кандидат $S_\alpha \in S$; D_i^0 — допустимое значение i-го требования, M — число требований.

Допустим, нам известен метод ранжирования подсистем по совокупности скалярных компонент каждой векторной компоненты $K_q(S_\alpha)$,

$q = \overline{1, r}$, например, метод "жесткого" ранжирования [2]. После его применения будут построены частные кортежи Парето, которые позволят однозначно определить расположение системы S_α относительно других систем по каждой векторной компоненте. Причем выявляются как доминирующие (доминируемые), так и эквивалентные системы. Это позволяет придать всем векторным компонентам некоторые числа, значения которых зависят от расположения систем: для доминируемых систем эти значения больше, чем для доминирующих, а для эквивалентных систем эти значения будут равными. Назовем такие числа ***псевдозначениями*** векторных компонент. Введение таких псевдозначений позволяет вновь применить метод "жесткого" ранжирования и построить искомый кортеж Парето. При этом необходимо иметь информацию о множестве $A = \{a_q, q = \overline{1, r}\}$ коэффициентов важности векторных компонент, причем $\sum_{q=1}^{r} a_q = 1$.

Таким образом, для решения задачи многовекторного ранжирования метод "жесткого" ранжирования применяется $[r + 1]$ раз.

Рассмотрим более подробно метод "жесткого" ранжирования, который позволяет строить упорядоченное множество эффективных вариантов — кортеж Парето либо подкортеж Парето с заданным числом элементов. Пусть число допустимых вариантов равно n.

В ходе решения задачи будем анализировать множество упорядоченных пар $S_k \in S_D$, $S_l \in S_D$ ($k = \overline{1, n}$; $l = \overline{1, n}$; $k \neq l$), а результат анализа заносить в специальную оценочную матрицу $\|C_{kl}\|$.

Сущность метода заключается в следующем.

1. На основе попарного сравнения вариантов S_k, S_l ($k = \overline{1, n}$; $l = \overline{1, n}$; $k \neq l$) определяем элементы C_{kl} оценочной матрицы $\|C_{kl}\|$. Значения элементов C_{kl} подбирают таким образом, чтобы отсечь неэффективные варианты. У эквивалентных вариантов S_k, S_l все соответствующие критерии равны. Полагаем $C_{kl} = 1$.

К числу неэффективных отнесем варианты, у которых:

а) все значения критериев k-го варианта хуже, чем у l-го варианта, тогда полагаем $C_{kl} = N_2 \gg 1$;

б) значения $m (m < r)$ критериев k-го варианта хуже соответствующих значений критериев l-го варианта при равных соответствующих значениях остальных критериев этих систем. Тогда полагаем $C_{kl} = N_3$, $1 \ll N_3 < N_2$.

Если же для систем k, l имеем лучшие, худшие и, возможно, равные критерии, то значение C_{kl} определим по методу, изложенному в работе [3].

Запишем выражения для элементов оценочной матрицы. Обозначим N_{kl}^{+}, N_{kl}^{-}, $N_{kl}^{=}$ — соответственно подмножества номеров лучших, худших и равных критериев для каждой пары вариантов S_k, S_l ($k = \overline{1, n}$; $l = \overline{1, n}$, $k \neq l$). Будем попарно сравнивать системы S_k, S_l на основе анализа критериев $K_j(S_k)$, $K_j(S_l)$, $j = \overline{1, r}$. Для возможных значений подмножеств номеров N_{kl}^{+}, N_{kl}^{-}, $N_{kl}^{=}$ соответственно лучших, худших и равных критериев введем следующие значения элементов оценочной матрицы $\|C_{kl}\|$:

$$\text{если } N_{kl}^{+} = \varnothing,\ N_{kl}^{-} = \varnothing,\ N_{kl}^{=} = \{\overline{1, r}\},\ \text{то } C_{kl} = 1,\ C_{lk} = 1; \tag{3}$$

$$\text{если } N_{kl}^{+} = \{\overline{1, r}\},\ N_{kl}^{-} = \varnothing,\ N_{kl}^{=} = \varnothing,\ \text{то } C_{kl} = N_2,\ C_{lk} = 0,\ N_2 \gg 1; \tag{4}$$

$$\text{если } N_{kl}^{+} = \varnothing,\ N_{kl}^{-} = \{\overline{1, r}\},\ N_{kl}^{=} = \varnothing,\ \text{то } C_{kl} = 0,\ C_{lk} = N_2; \tag{5}$$

$$\text{если } N_{kl}^{+} \neq \varnothing,\ N_{kl}^{-} = \varnothing,\ N_{kl}^{=} \neq \varnothing,\ \text{то } C_{kl} = N_3,\ C_{lk} = 0,\ 1 \ll N_3 < N_2; \tag{6}$$

$$\text{если } N_{kl}^{+} = \varnothing,\ N_{kl}^{-} \neq \varnothing,\ N_{kl}^{=} \neq \varnothing,\ \text{то } C_{kl} = 0,\ C_{lk} = N_3; \tag{7}$$

$$\text{если } N_{kl}^{+} \neq \varnothing,\ N_{kl}^{-} \neq \varnothing,\ |N_{kl}^{=}| \geqslant 0, \tag{8}$$

то C_{kl} (для скалярных критериев каждой q-й векторной компоненты) определим в виде [3]

$$C_{kl} = \sum_{i \in N_{kl}^{+}} a_{qi} \Big(\sum_{i \in N_{kl}^{-}} a_{qi} \Big)^{-1},\quad q = \overline{1, r},\ C_{lk} = C_{kl}^{-1}, \tag{9}$$

где a_{qi}, $i = \overline{1, r_q}$ — коэффициенты важности скалярных компонент, причем

$$\sum_{i=1}^{r_q} a_{qi} = 1,\ q = \overline{1, r}.$$

2. Для формулировки решающих правил введем числа: H_l — число элементов в l-м столбце оценочной матрицы, значение которых больше единицы; M_l — число элементов в l-м столбце той же матрицы, значение которых меньше единицы; $C_{kl\max}$ — максимальное значение элемента в l-м столбце матрицы $\|C_{kl}\|$.

Физический смысл чисел: H_l показывает, сколько вариантов из рассматриваемого множества превышают l-й; M_l — сколько вариантов доминирует l-я система; $C_{kl\max}$ определяет, во сколько раз l-й вариант "превышается" k-м ($k \in \{\overline{1, n}\}$, $k \neq l$).

3. Для реализации "жесткого" ранжирования перейдем от одношагового процесса поиска приоритетного расположения альтернатив к многошаговому процессу [4]. На каждом шаге t ($t = 1, 2, ..., n - 1$) выбираем j-ю альтернативу, лучшую с точки зрения предлагаемого ниже решающего правила. Затем ее номер включаем в множество P и в последующем рассмотрении j-я альтернатива больше не участвует (в матрице $\|C_{kl}\|$ вычеркиваем j-ю строку и j-й столбец). Это позволяет исключить влияние варианта S_j на выбор лучшей альтернативы, проводимой уже на шаге $(t + 1)$.

При формулировке решающих правил вновь используем, но теперь на каждом шаге t, числа $H_l^{(t)}$, $M_l^{(t)}$, $C_{kl\max}^{(t)}$, которые имеют оговоренный выше физический смысл.

Решающие правила "жесткого" ранжирования:

1. Ранжирование необходимо проводить среди эффективных альтернатив по шагам. Число шагов $t \leqslant (n - 1)$.

2. На каждом шаге t ($t = 1, 2, ..., n - 1$):

- найти числа $H_l^{(t)}$, $M_l^{(t)}$, $C_{kl\max}^{(t)}$ и определить лучшую альтернативу S_j с минимальным значением $H_j^{(t)}$ и $C_{lj} \geqslant 1$ $\forall l \in \{\overline{1, n}\}$, $l \neq j$;
- номер j занести в множество P;
- исключить из оценочной матрицы j-ю строку и j-й столбец.

Если альтернативы с номерами $l_j \in L_{k(t)} = \{l_1, l_2, ..., l_j, ..., l_{k(t)}\}$ имеют одинаковые минимальные значения $H_{l_j}^{(t)}$, то лучшей является альтернатива S_{l_j} с максимальным значением $M_{l_j}^{(t)} = \max\limits_{l_j \in L_{k(t)}} M_{l_j}^{(t)}$.

3. Если варианты с номерами $l_j \in L_{k(t)} = \{l_1, l_2, ..., l_j, ..., l_{k(t)}\}$ имеют соответственно одинаковые значения $H_{l_j}^{(t)}$, $M_{l_j}^{(t)}$, то лучшей является альтернатива S_{l_j} с минимальным значением $C_{kl_j\max}^{(t)}$.

4. Если лучшие системы имеют соответственно равные значения $H_{l_j}^{(t)}$, $M_{l_j}^{(t)}$, $C_{kl_j\max}^{(t)}$, то такие системы считают эквивалентными.

3. Численный пример

Исходные данные, сформированные экспертной группой, были использованы для решения задачи подбора перспективных кандидатов на должности руководящего и преподавательского состава по совокупности критериев качества. В вузе ДПО существовала необходимость замещения 11 вакантных должностей РПС следующих категорий: *руководитель* — 2 должности; *руководитель-преподаватель* — 2 должности; *преподаватель* — 7 должностей.

Каждому из 20 кандидатов экспертной группой по каждому критерию были выставлены оценки в соответствии с предложенной методикой. Результаты были сведены в три таблицы для категорий должностей: *руководитель, руководитель-преподаватель, преподаватель*. На основе метода многовекторного ранжирования была решена задача ранжирования 20 кандидатов на должности РПС с построением трех кортежей Парето.

Для категории *руководитель* кандидаты расположились в последовательности: 1, 6, 12, 9, 19, 5, 3, 2, 16, 20, 4, 7, 8, 10, 11, 13, 14, 15, 17, 18, т. е. кортеж Парето

$P(1) = <1, 6, 12, 9, 19, 5, 3, 2, 16, 20, 4, 7, 8, 10, 11, 13, 14, 15, 17, 18>$.

Для категории *руководитель-преподаватель* кортеж Парето выглядит следующим образом:

$P(2) = <1, 6, 9, 16, 19, 3, 5, 2, 4, 7, 8, 10, 11, 12, 13, 14, 15, 17, 18, 20>$.

Для категории *преподаватель* получен следующий кортеж Парето:

$P(3) = <3, 4, 12, 1, 6, 14, 5, 2, 19, 10, 7, 8, 9, 11, 13, 15, 16, 17, 18, 20>$.

Анализ результатов решения задачи показал:

кандидаты 1 и 6 готовы к выполнению функциональных обязанностей любой категории должностей руководящего и преподавательского состава вуза ДПО;

кандидаты 9 и 19 готовы к выполнению функциональных обязанностей в должностях категорий *руководитель* и *руководитель-преподаватель*;

кандидат 12 готов к выполнению функциональных обязанностей в должностях категорий *руководитель* и *преподаватель*;

кандидат 16 готов к выполнению функциональных обязанностей в должностях категории *руководитель-преподаватель*;

кандидаты 3, 4 и 14 готовы к выполнению функциональных обязанностей в должностях категории *преподаватель*;

кандидаты 2 и 5 заслуживают внимания в связи с равнозначной готовностью к выполнению функциональных обязанностей в должностях любой категории;

кандидаты 7, 8, 10 и 11 показали среднюю, но стабильную готовность к выполнению функциональных обязанностей в должностях любой категории;

кандидаты 13, 15, 17, 18 и 20 показали худшую пригодность к выполнению функциональных обязанностей всех трех категорий должностей и отнесены к неперспективным.

Заключение

Рассмотрены постановки задач, возникающих в процессе совершенствования системы образования. Для решения подобных задач предлагается применять методы многокритериального и многовекторного ранжирования. Метод многовекторного ранжирования раскрывается на фоне задачи подбора кандидатов на должности руководящего и преподавательского состава вуза. Для ее решения необходимо предварительно сформировать паспорта должностей, что позволяет обоснованно подготовить необходимую исходную информацию. Использование предлагаемого подхода позволяет руководителю быть объективным при подборе профессиональных кадров. Метод многовекторного ранжирования может быть применен для решения и других задач, возникающих в сфере образования, в частности, указанных во введении.

Список литературы

1. **Бешелев С. Д., Гурвич Ф. Г.** Экспертные оценки. М.: Наука, 1973. 159 с.
2. **Сафронов В. В.** Гипервекторное ранжирование сложных систем // Информационные технологии. 2003. № 5. С. 23—27.
3. **Руа Б.** Проблемы и методы решений в задачах со многими целевыми функциями // Вопросы анализа и процедуры принятия решений. М.: Мир, 1976. С. 20—58.
4. **Белкин А. Р., Левин М. Ш.** Принятие решений: комбинаторные модели аппроксимации информации. М.: Наука, 1990. 160 с.

УДК 336:004 + 519.7

С. А. Горбатков, д-р техн. наук, проф.,
зав. региональной кафедрой,
Всероссийский заочный
финансово-экономический институт,
филиал в г. Уфа
Д. В. Полупанов, канд. техн. наук,
ст. преподаватель, Башгосуниверситет, г. Уфа

Интеллектуальные информационные технологии отбора налогоплательщиков для проведения выездных проверок на основе гибридных нейросетевых моделей

Разрабатывается информационная технология налогового контроля, базирующаяся на гибридной нейросетевой модели. Указанная модель состоит из нейросетевой модели аппроксимации производственной функции кластера налогоплательщиков и вероятностной модели ранжирования налогоплательщиков, учитывающей искажения отчетной документации, предысторию и масштаб деятельности. Предлагаются и подтверждаются вычислительными экспериментами оригинальные методы предпроцессорной обработки данных, повышающие качество и адекватность разработанных моделей.

Ключевые слова: *технология налогового контроля, гибридная нейросетевая модель, нейросетевая модель аппроксимации производственной функции в кластере, вероятностная модель ранжирования, предпроцессорная обработка данных, метод вложенных математических моделей.*

Введение

Данная работа продолжает наши исследования по совершенствованию информационных технологий налогового контроля на основе нейросетевых моделей (НСМ). Недостатки существующих технологий налогового контроля, отдельные аспекты построения НСМ с позиций общесистемных закономерностей кибернетики приведены в [1—4]. В настоящее время только порядка половины выездных налоговых проверок приводят к существенным доначислениям [1]. Поэтому разработка интеллектуальных информационных технологий отбора налогоплательщиков для проведения выездных проверок актуальна.

Постановка задачи

С точки зрения моделирования задача отбора налогоплательщиков для проведения выездных проверок — это задача ранжирования стохастических объектов с сильно зашумленными данными. Пусть ретроспективному промежутку времени T, измеряемому в интервалах (квантах) времени отчета наблюдений, соответствует база данных (БД) $Z = <\boldsymbol{X},\ Y>_i,\ i = \overline{1, N},\ N = GT$, составленная на основе деклараций примерно однородной группы из G налогоплательщиков. Требуется выбрать наиболее существенные входные факторы — вектор $\boldsymbol{X} = (X_1, ..., X_n)$, например, такие, как сумма основных средств, себестоимость, среднесписочная численность сотрудников, коммерческие расходы и др., а также выходную величину Y, в качестве которой может выступать выручка, и построить нейросетевую модель (НСМ)

$$\hat{y} = F(\boldsymbol{x},\ W(\boldsymbol{x},\ y)). \qquad (1)$$

В формуле (1) и далее, согласно принятой в эконометрике нотации [5], строчными буквами обозначена конкретная численная реализация соответствующих случайных величин, обозначенных прописными буквами; знаком "^" над строчной буквой — реализация с помощью НСМ соответствующей выходной величины. Здесь $\boldsymbol{x} = (x_1, ..., x_n)$ — конкретная численная реализация вектора входных факторов $\boldsymbol{X}$; y — декларируемая налогоплательщиком конкретная численная реализация выходной величины Y; $\hat{y}$ — расчетное по модели (1) значение Y; $\{W\}$ — множество синаптических весов.

Концепция предлагаемого подхода к построению модели основана на двух предположениях.

1. Нарушения в налоговой декларации эффективнее выявляются сравнением производственных функций достаточно однородного кластера налогоплательщиков. Сравнительный анализ реализуется путем порождения *эталонного* (среднего для кластера) значения оценки моделируемой производственной функции (1) и вычисления для всех налогоплательщиков относительных отклонений в i-м наблюдении на момент оценки:

$$\delta_i = (\hat{y}_i - y_i)/y_i. \qquad (2)$$

2. Для учета предыстории и масштаба деятельности налогоплательщиков предложена вероятностная модель ранжирования (ВМР) на основе ψ-критерия:

$$\psi_g = \tilde{\delta}_{gt}|_{t = t^{TC}} P(\Delta_g \geqslant \bar{\tilde{\delta}}) M_g, \qquad (3)$$

где $\tilde{\delta}_{gt} = \delta_{gt} + U$ — значение верхней границы доверительного интервала для $\delta_{gt} \equiv \delta_i$ (в записи отклонения (2) фиксируется номер налогоплательщика g и момент наблюдения t); U — полуширина доверительного интервала для случайной величины δ_i по (2); волна сверху величины $\tilde{\delta}_{gt}$ в (3) означает смещение вверх относительного отклонения δ_{gt} на величину U (подробно о данном смещении см. ниже); $P(\Delta_g \geqslant \bar{\tilde{\delta}})$ — вероятность того, что ожидаемое значение отклонения Δ_g моделируемой случайной величины $\hat{y}$ будет не меньше выборочного среднего $\bar{\tilde{\delta}}$ с учетом его смещения вверх на полуширину доверительного интервала; в момент времени t^{TC}, соответствующий последнему кварталу подачи декларации налогоплательщиком, осуществляется ранжирование; M_g — экспертно задаваемый коэффициент масштаба g-го налогоплательщика.

Ранжирование заключается в присвоении каждому налогоплательщику в соответствии с (3) ранга, показывающего степень нарушения им налогового законодательства. Требуется найти множество номеров налогоплательщиков, планируемых для проведения выездной проверки:

$$\Theta = \left\{ g : \Psi = \sum_{v=1}^{G^*} \psi_{gv} \to \max_g \right\}, \qquad (4)$$

где G^* — число проверяющих бригад в инспекции Федеральной финансово-налоговой службы (ФНС) территориального уровня; v — порядковый номер налогоплательщика в синтезируемом на основе (4) плане отбора. Фактически (4) — это план выездных проверок в аспекте ожидаемых доначислений.

Смысл введения доверительного интервала для отклонений иллюстрируется рис. 1. Пусть Y^{true} — неизвестное нам истинное значение выходной величины. Тогда $\Delta_i = (y_i^{true} - y_i)/y_i$ (кривая *1*) — истинное значение относительного отклонения. По смыслу задачи наиболее вероятно, что налогоплательщик стремится искажать налогооблагаемую базу в сторону уменьшения. Бессмысленно платить налогов больше, чем требуется законодательством. Отсюда следует, что $\Delta_i \geqslant \delta_i$. Тогда введение доверительного интервала для отклонений приподнимает δ_i (кривая *2*) на полуширину доверительного интервала $\tilde{\delta}_i = \delta_i + U$ (кривая *3*), приближая его к Δ_i. В процедуре ранжирования согласно (3) используется смещенное на U значение отклонения $\tilde{\delta}_i$ и вычитается вероятность того, что истинное значение отклонения Δ_i не меньше условного математического ожидания $\bar{\tilde{\delta}}_i$. Последняя величина постулируется равной нулю, что позволяет считать на оси абсцисс $\bar{\delta}_i = 0$. Отклонения, лежащие ниже оси абсцисс, не участвуют в последующей процедуре ранжирования.

Аналитическая оценка доверительного интервала U на основе t-критерия Стьюдента [5] для НСМ принципиально невозможна. В оптимальной структуре НСМ число параметров (нейронов, синаптических весов и активационных функций) соизмеримо с числом измерений и даже превышает его, поэтому проблематично оценить число степеней свободы. Авторы предлагают способ оценки U, основанный на использовании метода обобщенного перекрестного подтверждения (ОПП), подробно рассмотренного в

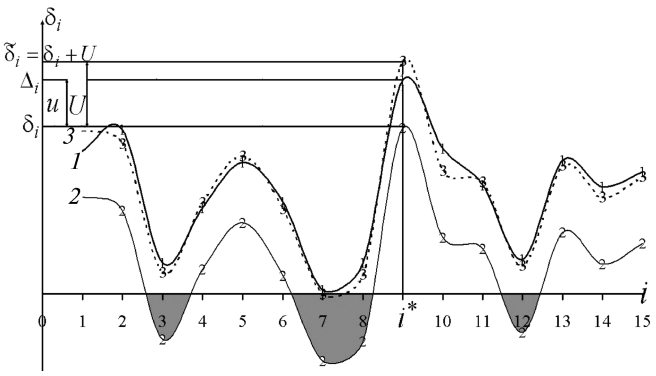

Рис. 1. Построение доверительного интервала для отклонений

разделе "Оценка и анализ адекватности модели" настоящей статьи. Вместо одной НСМ предлагается построить L параллельных моделей. Предположим, что L^* из них подтверждают друг друга с доверительной вероятностью P^l. Тогда аналогом доверительного интервала i-го наблюдения может служить оценка

$$U_i = \frac{1}{2}\left(\max_{l = \overline{1, L^*}} \delta_i^{(l)} - \min_{l = \overline{1, L^*}} \delta_i^{(l)} \right). \qquad (5)$$

Во взаимодействии НСМ и ВМР образуется гибридная нейросетевая модель (ГНСМ) ранжирования налогоплательщиков, которую можно записать в виде композиции операторов:

$$\Theta = F_{PMR} \circ F_{NNM}(\boldsymbol{X}, Y), \qquad (6)$$

где F_{NNM} — оператор НСМ аппроксимации производственной функции налогоплательщиков; F_{PMR} — оператор ВМР.

Данные для апробации основных идей по построению ГНСМ

Основные идеи по разработке и совершенствованию ГНСМ ранжирования налогоплательщиков, описанные в следующих разделах данной статьи, нами подтверждаются на выборках торговых предприятий, взятых из [1, 3] и обозначенных в [4] как Z^{I} и Z^{II}.

Выборка Z^{I}. Входные факторы: X_1 — сумма основных средств; X_2 — себестоимость; X_3 — среднесписочная численность работающих, чел.; X_4 — сумма оборотных активов; X_5 — среднегодовая стоимость облагаемого налогом имущества; X_6 — коммерческие расходы. Выходная величина Y — выручка. Все величины, кроме X_3, измеряются в тыс. руб. Выборка содержит данные 39 предприятий, каждое из них представлено наблюдениями за девять кварталов. Общее число наблюдений 351. Полностью исходные данные приведены в [1].

Выборка Z^{II}. Входные факторы: X_1 — сумма основных средств; X_2 —амортизационные отчисления; X_3 — оборотные активы; X_4 — запасы; X_5 — среднесписочная численность работающих, чел.; X_6 — дебиторская задолженность; X_7 — коммерческие расходы; X_8 — себестоимость. Выходная величина Y — выручка. Все величины, кроме X_5, измеряются в тыс. руб. Выборка содержит данные 22 налогоплательщиков, у половины из них имеются данные за 11 кварталов, остальные либо представили документацию не в полном объеме, либо начали хозяйственную деятельность позже. Всего имеются 194 наблюдения. Полностью исходные данные приведены в [3].

Оценка и анализ адекватности модели

Вопрос обеспечения адекватности НСМ в рассматриваемом классе задач теоретически не разработан, поскольку при построении НСМ нарушаются практически все постулаты регрессионного анализа [8]. Для обеспечения адекватности НСМ воспользуемся процедурой ОПП — *Generalized Cross Validation* (GCV), первоначально предложенной в [2]. В [4] предлагается следующее понимание ОПП, заключающееся в сравнении D параллельных НСМ по близости результатов расчета по некоторой числовой мере. Если все НСМ указывают большие отклонения вида (2) в одних и тех же наблюдениях, т. е. взаимно подтверждают классификацию соответствующих этим наблюдениям налогоплательщиков как "нарушителей", то оценку можно считать достаточно достоверной в рамках заданной выборки, так как вероятность неверной оценки "по вине" НСМ мала.

Получим для каждой НСМ типа $d = \overline{1, D}$ отклонения δ_i^d вида (2). Введем в качестве эталона сравнения параллельных НСМ среднее значение отклонений по ним: $\bar{\delta}_i = \sum_{d=1}^{D} \delta_i^d / D$, $i = \overline{1, N}$. В каждом i-м наблюдении для НСМ типа d проведем выполнение неравенства допустимого уклонения от "эталона":

$$|(\bar{\delta}_i - \delta_i^d)/\bar{\delta}_i| \leqslant \eta, \qquad (7)$$

где η — экспертно задаваемый уровень ошибки. Рассчитаем величину $P^d = N^d/N$, где N^d — общее число наблюдений i, для которых выполнено (7) для НСМ типа d. Если $P^d < P^{GCV}$, то НСМ типа d удовлетворяет процедуре ОПП. Здесь P^{GCV} — экспертно задаваемый уровень доверительной вероятности для рассматриваемой процедуры.

Усилением процедуры ОПП, повышающим адекватность НСМ, служит ее двухэтапное, "вложенное" использование. Зафиксируем D типов НСМ, различающихся парадигмой нейросетей (НС), числом скрытых слоев, числом нейронов и видом активационных функций в них. Для каждого типа НСМ $d = \overline{1, D}$ (на одном и том же обучающем множестве) построим L параллельных, независимо обученных НСМ. Как отмечалось в предыдущем разделе, они используются при определении доверительного интервала (5), в результате чего получено отклонение $\tilde{\delta}_i = \bar{\delta}_i + U_i$, $\bar{\delta}_i = \sum_{i=1}^{L^*} \delta_i^{(l)} / L^*$, $i = \overline{1, N}$. Каждое из $\tilde{\delta}_i^{(d)}$, полученных для НСМ различных типов, также оценивается по методу ОПП согласно (7).

Для оценки адекватности ВМР и всей ГНСМ (6) служит процедура, названная по аналогии с ОПП модифицированным обобщенным перекре-

Таблица 1

Сравнение ГНСМ по критерию ОПП и МОПП

Тип ГНСМ	P^d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
ГНСМ1	0,95	11	15	16	19	21	1	17	13	18	10	2	4	8	14	20	22	3	12	6	5	9	7
ГНСМ2	0,97	11	15	19	21	16	1	13	17	10	2	18	6	8	4	20	14	3	22	12	5	9	7
ГНСМ3	0,90	11	15	19	21	16	6	1	13	2	8	18	10	17	20	4	14	3	12	22	5	9	7
ГНСМ4	0,93	11	15	16	19	21	17	1	13	10	18	9	8	4	2	20	14	22	3	12	6	5	7
ГНСМ5	0,98	11	15	19	21	16	1	17	13	10	6	2	18	4	8	20	14	22	3	12	9	5	7
ГНСМ6	0,98	11	15	19	21	16	1	13	17	6	2	10	18	8	4	20	14	22	3	12	5	9	7

стным подтверждением (МОПП) — *Modify Generalized Cross Validation* (MGCV) [4]. Она играет роль обобщенной верификации ГНСМ, поскольку оценивает достоверность финишного результата модифицирования — плана отбора налогоплательщиков для проведения выездных проверок. МОПП заключается в сравнении множеств (4) по уже прошедшим ОПП D^* параллельным моделям

$$\Theta^d = \left\{ g: \Psi = \sum_{v=1}^{G^*} \psi_{vg}^d \to \max_g \ ; \ d = \overline{1, D^*} \right\}.$$

Если для двух независимых ГНСМ типов d_α и d_β, $\alpha \neq \beta$ из G^* возможных номеров налогоплательщиков, отобранных в множества Θ^{d_α} и Θ^{d_β}, совпадают G^{**} номеров независимо от порядка их исследования в отрезке $v \in [1; G^*]$, то считается, что процедура МОПП подтверждена с доверительной вероятностью $P^{MGCV} = G^{**}/G^*$, $G^{**} \leqslant G^*$. Процедура МОПП считается выполненной для D^{**} ГНСМ, если $P^{MGCV} \geqslant \eta^{**}$, где η^{**} — заданный уровень доверительной вероятности.

В табл. 1 отражены результаты ОПП и МОПП для шести типов моделей:

1) многослойный персептрон (*Multi Layer Perceptron — MLP*) с одним скрытым слоем и активационной функцией сигмоид (sigm) в нем;

2) MLP с двумя скрытыми слоями, активационной функцией sigm в них;

3) MLP с двумя скрытыми слоями и активационной функцией в первом скрытом слое — sigm, во втором — гиперболический тангенс (th);

4) MLP с одним скрытым слоем и активационной функцией th;

5) MLP с двумя скрытыми слоями и активационной функцией th в них;

6) MLP с двумя скрытыми слоями и активационной функцией в первом скрытом слое — th, во втором — sigm.

В выходных слоях активационная функция — линейная. Отношение $P^d = N^d/N$ представлено в первом столбце таблицы. В остальных указаны коды налогоплательщиков в базе данных в порядке очередности проверок, т. е. множество (4) для каждого типа ГНСМ.

При заданных критических значениях $\eta^* = 1$ и $P^{GVC} = 0{,}9$ ОПП прошли все НСМ. Предположим, что имеется десять проверяющих бригад, т. е. $G^* = 10$. Во всех ГНСМ из десяти налогоплательщиков совпадают семь (обозначены заливкой), т. е. модели подтверждают друг друга с доверительной вероятностью $P^{MGCV} = 0{,}7$. Считаем, что все ГНСМ прошли процедуру МОПП.

Процедуры предпроцессорной обработки данных для повышения качества ГНСМ

Специфика условий моделирования [1] приводит к необходимости разработки специальных процедур предобработки (или "ремонта") данных. Опыт авторов показывает, что построение адекватных моделей с приемлемыми аппроксимативными свойствами в данном классе задач без предобработки данных затруднительно. Нами предложены специальные процедуры предобработки и повышения однородности базы данных (БД), улучшающие качество обучения НСМ. Аналогично [4] представим ГНСМ (6) в виде композиции операторов:

$$\Theta = F_4 \circ [F_3 \circ F_2 \circ F_1](\boldsymbol{X}, Y), \qquad (8)$$

где $F_4 \equiv F_{PMR}$ — оператор ВМР; $F_3 \circ F_2 \circ F_1$ — композиция операторов предобработки данных, составляющая оператор F_{NNM} в (6);

$F_1 : Z \to \bigcup_{q=1}^{Q^*} Z_q$ в (8) — оператор оптимальной кластеризации. Здесь $Z_q = <\boldsymbol{X}, Y>_i$, $i = \overline{1, N_q}$ — БД q-го кластера; Q^* — оптимальное число кластеров, определяемое из условия

$$Q^*: \left[\left(\min_{d_{i,k}} \sum_{q=1}^{Q} \sum_{i,k}^{N_q} d_{i,k}^2 \right) \cap \right.$$

$$\left. \cap \left(\min_{Q} \max_{q} \{\Phi^{(q)}(Q, d_{i,k})\} \right) \right] \qquad (9)$$

при ограничениях на число наблюдений в кластере и критическое значение ошибки обобщения

$$(N_q/n) \geqslant \xi, \quad E_{\max}^{(q)} \geqslant E^*. \qquad (10)$$

Процедура оптимальной кластеризации исходной БД (образования в исходной БД оптимального числа достаточно однородных кластеров Z_q)

(9)—(10), в отличие от традиционных методов кластеризации, увязана с качеством обучения НСМ. В формуле (9) критерий d — евклидовы расстояния между элементами в кластере, $\Phi^{(q)}$ — обобщенный критерий качества вспомогательных НСМ (субмоделей) в q-м кластере на Q-й итерации, он представляет собой произведение трех частных критериев:

$$\Phi^{(q)} = E^{(q)} S^{(q)} R^{(q)}. \qquad (11)$$

Критерий $E^{(q)} = \|\hat{y}^{(q)} - y\|/\|y\|$ — ошибка обобщения — характеризует точность субмоделей. Критерий $S^{(q)} = |\hat{y}_\alpha^{(q)} - \hat{y}_\beta^{(q)}|/\|\boldsymbol{x}_\alpha - \boldsymbol{x}_\beta\|_{\Re^n}$ — аналог константы Липшица — характеризует устойчивость субмоделей. Здесь векторы $\boldsymbol{x}_\alpha$, $\boldsymbol{x}_\beta$ близки по норме в пространстве вещественных чисел $\Re^n$, $\hat{y}_\alpha^{(q)} = F(\boldsymbol{x}_\alpha, W)$ $\hat{y}_\beta^{(q)} = F(\boldsymbol{x}_\beta, W)$ — расчетные значения Y в точках наблюдений α, β тестового множества НС. Критерий $R^{(q)} = 1 - (r_{y,\hat{y}^{(q)}})^2$ определен по аналогии с коэффициентом детерминации, где $r_{y,\hat{y}^{(q)}}$ — коэффициент корреляции между декларированными и расчетными значениями Y.

Проиллюстрируем выполнение процедуры оптимальной кластеризации на примере выборки Z^{I}. НСМ строились на основе MLP с двумя скрытыми слоями. Активационная функция в первом скрытом слое — sigm, во втором — th, в выходном — линейная. Значение критерия (11) на каждой итерации в каждом кластере представлено в табл. 2. На рис. 2 представлен график зависимости максимального значения (11) от номера итерации Q. Из табл. 2 и рис. 2 следует, что найдено оптимальное число кластеров $Q^* = 2$.Таким образом, процедура оптимальной кластеризации состоятельна.

Второй оператор в композиции (8) $F_2: Z_q \to Z_q^*$ — оператор очистки кластера [7], где $Z_q^* = \langle \boldsymbol{X}, Y \rangle_i$, $i = \overline{1, N_{k^*}}$ — БД очищенного кластера; k^* — номер оптимальной итерации, определяемый из условия

$$k^*: ((\min_k \Phi^{(k)}) \cap (\delta_i^{(k)} = |\hat{y}^{(k)}{}_i - y_i)/y_i|100\ \% > \\ > \varepsilon^{(k)},\ i = \overline{1, N_k}))$$

при ограничениях $(N_k/n) \geqslant \xi$, $E^{(k)} \geqslant E^{**}$, аналогичных (10). Критерий $\Phi^{(k)} = E^{(k)} S^{(k)} R^{(k)}$ определяется аналогично (11), $\varepsilon^{(k)}$ — уровень аномальности отклонений (параметр отбора удаляемых аномальных точек на k-й итерации). Итогом является устранение аномальных наблюдений в каждом кластере и определение оптимальной итерации. В отличие от традиционных методов устранения аномальных наблюдений, не связанных с

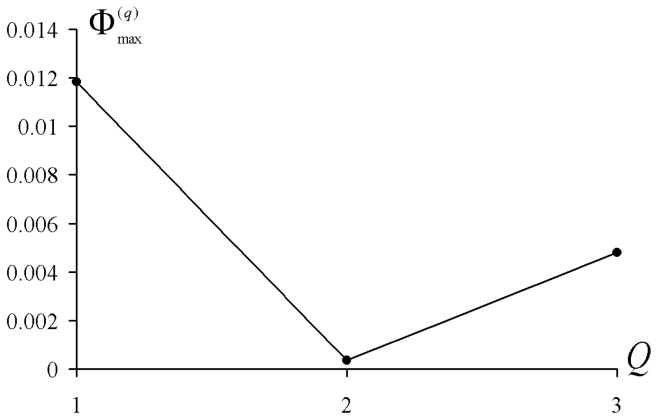

Рис. 2. Зависимость критерия $\Phi^{(q)}_{\max}$ от номера итерации Q

Таблица 2

Обобщенный критерий $\Phi^{(q)}$ для q-го кластера на Q-й итерации процедуры оптимальной кластеризации

Число кластеров, Q	Номер кластера, q		
	1	2	3
	$\Phi^{(q)}_{\max}$		
1	0,0196	—	—
2	0,0022	0,0015	—
3	0,0869	0,1139	0,0354

дальнейшим обучением моделей [8], изложенная выше процедура очистки кластера от аномальных наблюдений увязывает эти процессы, поскольку в силу существенного искажения БД удаляемые аномальные точки имеют разную информативность в аспекте обучения НСМ.

Проиллюстрируем выполнение данной процедуры на примере выборки Z^{II} (табл. 3 и рис. 3). НСМ строились на основе MLP с двумя скрытыми слоями. Активационные функции в скрытых слоях — sigm и th соответственно, в выходном — линейная. В табл. 3 для каждой итерации указаны значения критериев E, S, R и Φ, число наблюдений в кластере N, число аномальных наблюдений A, среднее значение отклонения $\bar{\delta}$ и уровень аномальности ε. Из таблицы и рисунка следует, что найдена оптимальная итерация $k^* = 3$. Процедура очистки кластера состоятельна.

Третий оператор в (8) F_3: $Z_q \to \hat{Y}$ — оператор "рабочей" НСМ, где Z_q, $q = \overline{1, Q^*}$ — БД q-го кластера, $\hat{y} = f^*(\boldsymbol{x}, (W(\boldsymbol{x}, y))$ — расчетное значение выходной величины на основе "рабочей" НСМ, $\hat{Y}$ — множество расчетных значений выходной величины Y.

В качестве одного из способов "ремонта" искаженных данных предлагается метод вложенных математических моделей (ВММ), заключающийся в использовании нескольких НСМ, "вложенных" друг в друга или в общую модель алгоритмически. Разо-

бьем вектор входных факторов $\boldsymbol{X}$ на два вектора: $\boldsymbol{X}^{ddist}$ — трудно искажаемых и $\boldsymbol{X}^{edist}$ — легко искажаемых факторов. Для каждого $\boldsymbol{X}_j^{edist}$, $j = \overline{1, n^{edist}}$, строится субмодель

$$\hat{x}_j^{edist} = f_j(\boldsymbol{X}^{ddist}, W).$$

После этого осуществляется корректировка легко искажаемых факторов через их аппроксимацию на основе трудно искажаемых факторов и

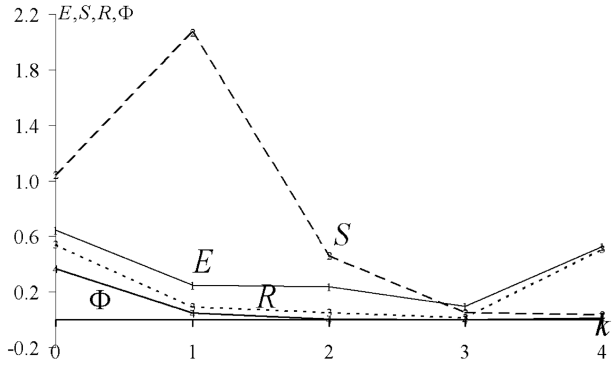

Рис. 3. Зависимость частных критериев *E*, *S* и *R* и обобщенного критерия Ф от номера итерации *k*

Таблица 3

Результаты выполнения процедуры очистки кластера

k	*E*	*S*	*R*	Ф	*A*	*N*	$\bar{\delta}$, %	ε, %
0	0,6476	1,0460	0,5431	0,3679	15	172	39	100
1	0,2500	2,0699	0,0923	0,0478	5	157	30	100
2	0,2384	0,4641	0,0512	0,0057	7	152	32	100
3	0,0961	0,0537	0,0165	0,0001	3	145	25	100
4	0,5258	0,0370	0,5076	0,0099	1	142	18	100

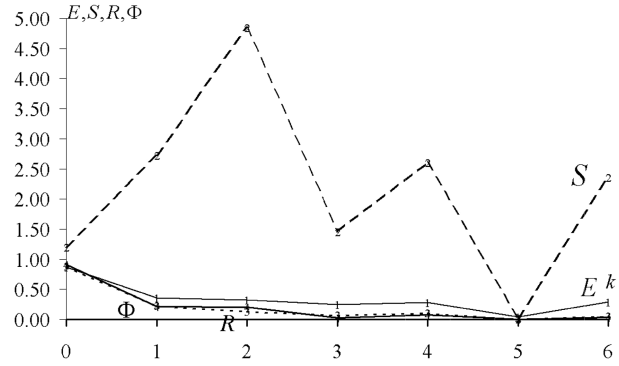

Рис. 4. Зависимость частных критериев *E*, *S* и *R* и обобщенного критерия Ф от номера итерации *k* для субмодели

Таблица 4

Результаты выполнения процедуры очистки кластера для субмодели

k	*E*	*S*	*R*	Ф	*A*	*N*	$\bar{\delta}$, %	ε, %
0	0,8666	1,1977	0,8812	0,9146	24	172	1206	200
1	0,3568	2,7221	0,2235	0,2171	9	148	58	150
2	0,3199	4,8507	0,1298	0,2014	8	139	58	150
3	0,2472	1,4543	0,0647	0,0232	24	131	56	100
4	0,2813	2,6052	0,1036	0,0759	7	107	42	100
5	0,0442	0,0129	0,003	2E-0	3	100	38	100
6	0,2936	2,3599	0,0502	0,0348	5	97	43	100

строится общая НСМ с использованием скорректированных входных факторов:

$$\hat{y} = \varphi(\boldsymbol{X}^{ddist}, \hat{\boldsymbol{X}}^{edist}, W).$$

Покажем, что предлагаемый метод ВММ повышает адекватность НСМ и построенных на их основе ГНСМ. Результаты построения ГНСМ без использования данного метода и оценка ее адекватности были представлены на рис. 3 и в табл. 1 и 3. Перейдем к построению НСМ для легко искажаемого фактора X_8 (себестоимость). В табл. 4 и на рис. 4 приведены результаты выполнения процедуры очистки кластера.

Скорректировав легко искажаемый фактор, построим "отремонтированную" НСМ. Результаты очистки кластера для нее приведены в табл. 5 и на рис. 5. Ошибка обобщения *E* и финишный критерий Ф у "отремонтированной" НСМ меньше, чем у модели, построенной без использования метода ВММ. Для построения "рабочей" модели потребовалась всего одна итерация очистки кластера. Тем самым качество "рабочей" НСМ при применении метода ВММ повышается.

В табл. 6 представлены результаты оценки адекватности. Все НСМ прошли процедуру ОПП с доверительной вероятностью $P^{GCV} = 0{,}91$. Тем самым "отремонтированные" НСМ считаем более адекватными, чем построенные без корректировки легко искажаемого фактора. Во всех ГНСМ из десяти налогоплательщиков совпадают девять (обозначены заливкой), т. е. модели подтверждают друг друга с доверительной вероятностью $P^{MGCV} = 0{,}9$. При этом все модели, кроме ГНСМ5, подтверждают друг друга с доверитель-

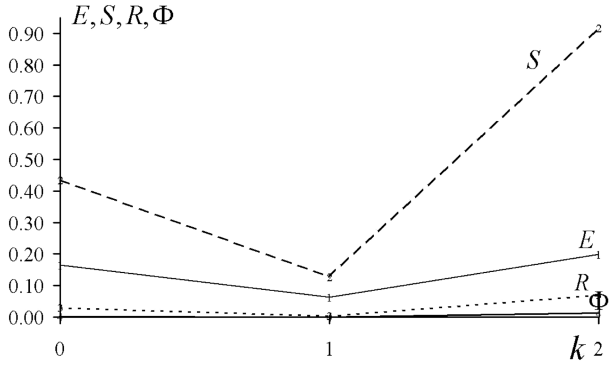

Рис. 5. Зависимость частных критериев *E*, *S* и *R* и обобщенного критерия Ф от номера итерации *k* для "отремонтированной" НСМ

Таблица 5

Результаты выполнения процедуры очистки кластера для "отремонтированной" НСМ

k	*E*	*S*	*R*	Ф	*A*	*N*	$\bar{\delta}$, %	ε, %
0	0,1647	0,4358	0,0285	0,002	34	172	84	100
1	0,0637	0,1279	0,003	0,00002	19	138	46	100
2	0,1992	0,9177	0,0703	0,0129	8	119	36	100

Таблица 6

Сравнение ГНСМ по критерию ОПП и МОПП (с использованием метода ВММ)

Тип ГНСМ	P^d	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
ГНСМ1	0,95	9	10	1	15	21	17	16	6	13	19	4	20	2	11	18	22	12	3	14	8	7	5
ГНСМ2	0,94	9	10	1	15	21	17	16	6	13	19	11	4	20	2	18	22	12	3	14	8	7	5
ГНСМ3	0,91	9	10	1	15	17	21	16	13	6	19	4	2	20	11	18	22	12	3	14	8	7	5
ГНСМ4	0,96	9	10	1	15	21	17	6	16	13	19	11	4	20	2	18	22	12	3	14	8	7	5
ГНСМ5	0,94	9	10	1	15	21	17	16	6	13	11	19	4	20	2	18	22	12	3	14	8	7	5
ГНСМ6	0,92	9	10	15	1	21	17	16	13	6	19	4	20	2	22	18	11	12	3	14	8	7	5

ной вероятностью $P^{MGCV} = 1$. Тем самым численно показано, что использование метода ВММ повышает точность и адекватность ГНСМ ранжирования налогоплательщиков.

Теперь с учетом предложенных методов предобработки данных мы можем дать развернутую формализованную запись ГНСМ:

$$\Theta = F_4 \circ [F_3 \circ F_2 \circ F_1]([F_3 \circ F_2 \circ F_1] \times \times (\boldsymbol{X}^{ddist}, \hat{\boldsymbol{X}}^{edist}), \boldsymbol{X}^{ddist}, Y). \quad (12)$$

Вывод

Разработанная ГНСМ (13) может служить основой информационной технологии предварительных (камеральных) проверок и отбора налогоплательщиков для проведения выездных проверок, обладающей большей степенью объективности и достоверности и меньшей трудоемкостью, чем существующие информационные технологии налогового контроля.

Список литературы

1. **Букаев Г. И., Бублик Н. Д., Горбатков С. А. и др.** Модернизация системы налогового контроля на основе нейросетевых информационных технологий. М.: Наука, 2001.
2. **Горбатков С. А., Габдрахманова Н. Т.** Способы улучшения ассоциативных свойств нейросетевых математических моделей в системе налогового контроля и управления // Информационные технологии. 2001. № 4. С. 7—14.
3. **Бублик И. Д., Голичев И. И., Горбатков С. А. и др.** Теоретические основы разработки технологии налогового контроля и управления. Уфа: РИО БашГУ, 2004.
4. **Полупанов Д. В.** Математические модели ранжирования объектов налогового контроля. Автореф. дисс. ... к.т.н.: 05.13.18. Уфа, Уфим. гос. авиац. техн. ун-т, 2007.
5. **Кремер Н. Ш., Путко Б. А.** Эконометрика. М.: ЮНИТИ ДАНА, 2002.

ПРИКЛАДНЫЕ ИНФОРМАЦИОННЫЕ СИСТЕМЫ

УДК 002.66

М. М. Горбунов-Посадов*, д-р физ.-мат. наук, ст. научн. сотр.,
Н. В. Максимов,** д-р техн. наук, проф.,
Т. А. Полилова*, д-р физ.-мат. наук, ст. научн. сотр.,
В. И. Строгонов*,** д-р техн. наук, зам. директора Центра программных технологий
* Институт прикладной математики им. М. В. Келдыша РАН
** Московский инженерно-физический институт (технический университет)
*** Международный научно-исследовательский институт проблем управления

Об информационном обеспечении деятельности ВАК Минобрнауки России

В настоящее время ВАК Российской Федерации ставит задачу широкого использования информационных технологий для организации взаимодействия с сетью диссертационных советов. Рассматривается сложившаяся ситуация в области документооборота "ВАК —диссертационные советы" и документооборота внутри ВАК. Предлагается использовать web-формы в качестве основного механизма формирования аттестационных дел. Обсуждается сложившаяся практика и перспективы размещения в сети Интернет авторефератов и полных текстов диссертаций. Анализируются структура и интерфейсы баз данных "Ученые России", "Диссертационные советы" и др.

Ключевые слова: *высшая аттестационная комиссия, информационные технологии, процессы аттестации научных кадров, автоматизированные системы управления, информационно-аналитические системы, web-формы.*

17 декабря 2007 г. состоялось заседание Высшей аттестационной комиссии Министерства образования и науки, посвященное состоянию и задачам государственной аттестации научных и научно-педагогических кадров высшей квалификации. В своем докладе председатель ВАК М. П. Кирпичников отметил, что "...основная

задача ВАК — обеспечение единой политики и осуществление контроля и координации всех, кто занимается аттестацией науки и научно-педагогических кадров. Ясно, что через это не прямо, но косвенно мы содействуем улучшению научно-исследовательской работы" [1].

Очевидно, что решение этой задачи сегодня невозможно без применения современных информационных технологий. Это подтвердил председатель ВАК М. П. Кирпичников, сказав о том, что вопрос информационного обеспечения системы аттестации научных кадров выходит на первый план [2].

Управлением государственной аттестации научных и научно-педагогических работников Федеральной службы по надзору в сфере образования и науки в течение ряда последних лет велась разработка вопросов информационного обеспечения и применения современных информационных технологий в системе аттестации научных и научно-педагогических кадров. Учитывая, что эти вопросы имеют важное значение для выполнения основной задачи ВАК, представляется целесообразным дать оценку состояния работ в этой области и наметить направления их дальнейшего развития.

На сегодняшний день в работе ВАК и Управления государственного контроля в сфере аттестации научных и научно-педагогических работников используются программные комплексы "Ученые России" и "Диссертационные советы", а также функционирует сайт ВАК [3].

Программный комплекс "Ученые России" предназначен для сопровождения постоянно пополняемой базы утвержденных ВАК аттестационных дел (АД). Накопление данных по докторским диссертациям ведется с 1993 г., а по кандидатским — с 1997 г. Информация в базу данных вводится с оригиналов решений Президиума ВАК и соответствующих справок к ним. К настоящему времени в базе данных (БД) "Ученые России" содержатся данные о почти 150 тыс. кандидатов наук и 40 тыс. докторов наук.

Программный комплекс "Диссертационные советы", включающий БД "Диссертационные советы", "Отчет" и др., предназначен для сопровождения текущего состояния сети диссертационных советов.

База данных "Диссертационные советы" предназначена для информационного обеспечения работы с сетью диссертационных советов, контроля за выполнением требований Положения о диссертационном совете.

База данных "Диссертационные советы" содержит текущую информацию обо всех действующих советах (свыше 3000 советов), их полномочиях, составах и организациях, в которых они созданы.

Ввод данных в БД осуществляется с оригиналов приказов об утверждении советов и частичных изменениях их составов.

Разработка указанных систем началась более 10 лет назад, и с точки зрения сегодняшних возможностей эффективность их невысока. Во многом это определяется тем, что системы не обладают возможностями автоматизированного обновления данных и требуют ручного ввода информации. Представляемая по запросам информация носит статистический характер, ограничены виды запросов, отсутствует возможность подготовки аналитической информации.

Программный комплекс "Диссертационные советы" не предусматривает автоматизации документооборота рассмотрения материалов диссертационных советов. Информация из БД "Ученые России" недоступна для оперативной работы сотрудникам Управления, экспертным советам ВАК. Отсутствует возможность обращения к информационным ресурсам ВАК соискателей, диссертационных советов и организаций.

Названные недостатки привели к необходимости разработки иного подхода к информационному обеспечению работы ВАК, основанного на организации электронного обмена информацией между ВАК и диссертационными советами.

С этой целью в рамках проектов ФЦПРО, которые выполнялись по предложению Управления государственного контроля в сфере аттестации научных и научно-педагогических работников (Управления) с участием специалистов МГИЭМ, РГУТП, ВГТУ, ИПМ им. Келдыша, МИФИ, ГНИИ Информика, решались задачи:

- разработки технологии сбора информации, поступающей из диссертационных советов и организаций в электронном виде, ее обработки и хранения;
- разработки информационного обеспечения деятельности подразделений, экспертных советов и Президиума ВАК;
- автоматизации и повышение оперативности работы сотрудников Управления.

Последняя задача приобретает особую актуальность в связи с постоянным сокращением численности сотрудников Управления, обеспечивающих работу экспертных советов и президиума ВАК. Так, по словам М. П. Кирпичникова "... в настоящее время численность аппарата (Управления) всего 45 человек. В ВАКе СССР работало 200 штатных сотрудников, в ГВАКе — 90 человек, а нагрузки советского времени и текущей работы ВАК — вещи несопоставимые" [1].

Кратко остановимся на полученных результатах.

Разработана технология обмена информацией о защитах и аттестационных делах между диссертационными советами и информационной системой ВАК в электронной форме.

Подготовка аттестационных документов (данных в электронном виде для передачи в ВАК) выполняется средствами автоматизированных рабочих мест (АРМ) "Диссертационный совет", результатом чего являются как твердая, так и электронная копии данных документов.

Диссертационные советы передают данные в информационную систему посредством заполнения web-форм. При вводе выполняются все необходимые проверки, благодаря чему данные сразу, без вмешательства персонала ВАК могут быть записаны в базу данных, которая работает под управлением СУБД MS SQL Server.

Диссертационный совет, включаемый в систему распределенной подготовки данных, регистрируется в Управлении государственной аттестации. В состав регистрируемых данных входят название организации, шифр диссертационного совета, основной электронный адрес (e-mail) и дополнительный (запасной) электронный адрес.

После успешного приема того или иного электронного документа на экране появляется соответствующее сообщение. Кроме того, на основе данных о регистрации диссертационного совета автоматически генерируется и отправляется электронное письмо в диссертационный совет. Содержание письма имеет стандартный вид:

"Справка к аттестационному делу получена" или "Сведения о членах диссертационного совета получены".

Получаемая в электронном виде информация (данные по аттестационным делам в виде электронной копии справки диссертационного совета о присуждении ученой степени) используется в информационно-технологической системе (ИТС) документооборота аттестационных дел [4]. Прохождение аттестационного дела в ВАК сопровождается дополнением получаемой электронной копии АД информацией о результатах его рассмотрения на различных этапах и завершается компьютерной печатью документов (решений президиума ВАК, дипломов и т. д.).

Опытно-промышленный образец разработанного документооборота аттестационных дел прошел апробацию в двух экспертных советах отдела естественных и технических наук.

Основными результатами по этому направлению являются:

- автоматизированная технология (электронный регламент) управления прохождением аттестационного дела в ВАК и унифицированные формы представления учетных и аналитических данных об авторе и научной работе (на основе XML и онтологий);
- автоматизированная интегральная информационная система, имеющая в своем составе подсистему автоматизированного управления процессом прохождения АД и подсистему справочно-информационного обслуживания;
- информационные ресурсы интегральной АИС, включающие технологическую БД подсистемы управления процессом прохождения АД, электронную картотеку аттестованных кадров высшей квалификации, ретроспективную БД аттестационных дел.

В основу технологии положена концепция интегральной автоматизации подготовки и обработки данных о диссертационных работах (ДР) с однократным контролируемым вводом и многоаспектным использованием информации, поэтапно формируемой на каждой стадии рассмотрения ДР. Использование принципа распределенного поэтапного и поэлементного предоставления информации позволяет выполнить требования оперативности, достоверности и функциональной неизбыточности информации, а также обеспечить эволюционное развитие автоматизированных технологий обработки аттестационных дел и информационно-аналитического обеспечения процессов управления системой аттестации. В частности, это позволит обеспечить корректность данных в БД информационных карт (ИК) ВНТИЦентра (данные по состоянию утверждения, а не защиты, как в настоящее время), а в ВАК сведения о поступлении копии диссертации будут поступать автоматически в электронной форме в виде даты и номера регистрации.

Разработанная информационная форма представления диссертационной работы создает условия для формирования баз знаний, в основе которых будут БД авторефератов диссертаций и заключений диссертационных и экспертных советов, представляющих аналитическую информацию в унифицированной структурированной форме. Это создает возможность автоматизированной обработки запросов, характеризующих актуальность, новизну и значимость работы.

Подсистема автоматизированного управления процессом прохождения АД предназначена для управления обработкой данных, относящихся ко всем этапам прохождения аттестационного дела в ВАК. Функциональные блоки сгруппированы в соответствии с организационной структурой Управления госаттестации и реализованы в виде комплекса автоматизированных рабочих мест. Подсистема реализует технологию, учитывающую все регламентные нормы рассмотрения АД, включая циклы, обусловленные решениями о дооформлении дела или о его дополнительной экспертизе и т. д., а также обеспечивает справочно-информационное обслуживание запросов о прохождении АД, формирование ежемесячных и годовых отчетов о рассмотрении аттестационных дел, доступ к электронным копиям нормативных документов.

Подсистема справочно-информационного обслуживания обеспечивает выборку данных из технологической или ретроспективной баз данных, в том числе для следующих классов информационных задач:

- определение показателей работы подразделений Управления, экспертных советов, Президиума ВАК, диссертационных советов;
- определение тематического, регионального спектров научных исследований (распределение по специальностям, по регионам, организациям и советам, статусу, форме представления и защиты диссертации и т. д.);
- поиск данных, относящихся к личности, выполняющей научные исследования или обеспечивающей руководство, консультирование, оппонирование;
- выявление научных взаимосвязей персон и организаций ("виртуальных коллективов").

Как внешний информационный ресурс подсистема реализована в виде электронной библиотеки, построенной на основе комплекса ретроспективных баз данных, создаваемых на основе технологических БД по диссертационным работам, поступающим в ВАК. Система обеспечивает стандартный и расширенный поиск. Одной из поисковых возможностей является генерация гипертекстовых ссылок для динамического связывания элементов данных в различных, в том числе внешних, ассоциированных информационных ресурсах, что обеспечивает автоматический поиск работ автора в реферативно-библиографических БД и Internet-ресурсах, поиск тематически связанных работ в базах данных описаний диссертаций и НИР (ВНТИЦентра) и т. д.

В мае 2007 г. была предпринята попытка опробовать разработанную технологию обмена информацией между диссертационными советами и информационной системой ВАК в режиме опытной эксплуатации. В соответствии с письмом заместителя руководителя Рособрнадзора Шамхалова Ф. И. "Об опытной эксплуатации системы формирования и передачи в Рособрнадзор электронной справки" диссертационные советы по защите диссертаций на соискание ученой степени доктора и кандидата юридических наук были обязаны при представлении аттестационных дел в ВАК Минобрнауки России также направлять и "Электронную справку диссертационного совета". Однако эта попытка оказалась неудачной, что в значительной мере было обусловлено начавшейся очередной реорганизацией структуры Управления госаттестации.

Неудачной, на наш взгляд, оказалась и следующая, предпринятая в марте 2008 г. попытка наладить электронный документооборот между ВАК и диссертационным советом. Диссертационным советам предписывалось подать электронные отчеты за 2007 г., для чего им было предложено заполнить шаблоны, представленные в формате Microsoft Excel. Работа с такими шаблонами вызывает множество затруднений. Возможности проверки и предварительной обработки вводимых в шаблон Excel значений весьма ограничены, и поэтому такие электронные отчеты нуждаются в последующем дополнительном ручном или автоматическом контроле. Заполненные шаблоны предписывалось пересылать по электронной почте, что также связано с целым рядом потенциально возможных сбоев и ошибок.

Сайт Высшей аттестационной комиссии (http://vak.ed.gov.ru) функционирует с 2001 г. Сайт предназначен для оперативной публикации в Интернете материалов о деятельности ВАК и Управления государственной аттестации научных и научно-педагогических работников Федеральной службы по надзору в сфере образования и науки и организации доступа к нормативным и справочным материалам по вопросам аттестации научных и научно-педагогических кадров.

В целом структура сайта отвечает поставленным задачам. Введение же размещения в Интернете объявления о защите и автореферата диссертации, предусмотренного Положением о порядке присуждения ученых степеней (с изменениями, утвержденными Правительством РФ постановлением от 20.04.2006 № 227), предъявило дополнительные требования к организации сайта. Согласно этому Постановлению, за месяц до защиты авторефераты докторских диссертаций размещаются на официальном сайте ВАК, а авторефераты кандидатских диссертаций — на сайте организации, при которой создан диссертационный совет.

Для размещения объявлений о защите и авторефератов диссертаций на сайте был открыт дополнительный раздел "Объявления о защите докторских диссертаций". Внутри раздела объявления распределены по отраслям наук. Внутри каждой отрасли объявления о защите объединяются по группам (в файлах) по датам размещения на сайте. Каждое объявление содержит гиперссылку на автореферат диссертации.

Материалы (объявления о защите и авторефераты диссертаций) для размещения на сайте ВАК поступают по электронной почте. После обработки (проверка полноты и правильности предоставления материалов, объединения по отраслям наук и датам размещения и т. д.) материалы размещаются на сайте ВАК.

Такой механизм требует значительных трудозатрат и создает проблемы при необходимости устранения возникающих погрешностей в размещаемых материалах. К их числу можно отнести:

- проблемы с процессом пересылки материалов: сбой отсылки со стороны диссертационного совета, обнаружение вируса в присланном сообщении, некорректное разбиение архива автореферата на несколько частей с дальнейшей их пересылкой в разных сообщениях, пересылка сообщений без вложений, многократное дублирование сообщений, размещение автореферата с рисунками и диаграммами прямо в тексте письма и др.;
- проблемы с содержанием файла сведений: отсутствие требуемых сведений о предстоящей защите (предполагаемой даты защиты, телефона и e-mail совета); использование редких шрифтов и кодировок в тексте; указание e-mail соискателя вместо требуемого e-mail совета; не указание (ошибочное указание) отрасли наук; запрет на редактирование файла сведений;
- проблемы с содержанием файла автореферата: авторефераты без титульного листа и его оборотной стороны; отдельно от текста таблицы и приложения; представление автореферата в виде отдельных отсканированных листов; файлы с нарушенной целостностью или с использованием гипертекстовых ссылок к внешним материалам и т. д.

Размещение в сети Интернет указанной информации, осуществляемое без применения средств автоматизации, приводит к задержке размещения этих данных, возникновению ошибок. Аналогичные проблемы существуют и при размещении сведений о защитах кандидатских диссертаций на сайтах организаций, в которых действуют диссертационные советы. Практически нигде не фиксируется дата размещения объявления.

Более рациональной схемой представляется самостоятельная, без участия сотрудников ВАК, публикация диссертационных материалов непосредственно диссертационными советами. Для этого на официальном сайте должны быть предусмотрены электронные формы, заполняя которые представитель диссертационного совета и осуществляет размещение объявления и автореферата. Доступ к экранным формам открывается только по предъявлению пароля, высланного ВАК диссертационному совету [5].

Такие формы, содержащие средства приема, контроля и размещения информации о защитах докторских диссертаций на сайте ВАК были разработаны в 2007 г. Они готовы к использованию, и введение их в действие зависит только от решения Рособрнадзора.

Переход к экранным формам позволит размещать на официальном сайте ВАК также и объявления о защите кандидатских диссертаций и их авторефераты, что даст возможность значительно расширить число участников предварительного обсуждения работ.

Это также позволит решить вопрос о возможности редактирования опубликованных на сайте текстов объявлений о защите. Причины для внесения изменений могут быть различными: опечатка в фамилии соискателя ученой степени, ошибка в названии диссертации, перенос даты и времени защиты и т. д. Возможным решением является протоколирование всех изменений, вносимых диссертационным советом в ранее опубликованное объявление о защите, и размещение протокола на официальном сайте. Такая схема контролируемого внесения изменений может быть реализована через соответствующие экранные формы, размещаемые на официальном сайте и доступные ученому секретарю диссертационного совета.

Подводя итог рассмотрению результатов работ, можно сделать следующие выводы.

Проведенные Управлением государственной аттестации работы позволили создать основу для перехода к качественно новому информационному обеспечению государственной системы аттестации научных и научно-педагогических кадров. Не вызывает сомнения пра-

вильность проектных решений и эффективность перехода к информационному обеспечению ВАК, основанному на технологии электронного обмена информацией с диссертационными советами.

Разработан информационный комплекс ВАК, центральным элементом которого является информационно-технологическая система (ИТС) документооборота аттестационных дел, позволяющая обеспечить:

- получение, хранение, обработку и использование информации, получаемой от диссертационных советов и организаций, автоматизированное формирование базы данных по аттестационным делам;
- оперативную распределенную обработку данных и ввод дополнительной информации на автоматизированных рабочих местах соответствующих структурных подразделений Управления и экспертных советов;
- оперативное предоставление руководству ВАК, Рособрнадзора и Управления, членам ВАК, ЭС информации о прохождении аттестационных дел, о деятельности диссертационных советов, статистические справки о деятельности системы аттестации, аналитические материалы о работе экспертных советов, структурных подразделений Управления и т. д.;
- автоматизированную подготовку документов в Управлении (протоколов и справок, выносимых на рассмотрение Президиума ВАК; запросов и писем; статистические формы отчетности и т. д.);
- учет замечаний по работе диссертационных советов (по результатам рассмотрения аттестационных дел);
- автоматизированное формирование ретроспективной справочно-информационной БД по аттестационным делам и БД "Ученые России";
- автоматизированную компьютерную печать дипломов и аттестатов;
- предоставление информации диссертационным советам и организациям в режиме Интернет-доступа о прохождении аттестационных дел, о принятых Президиумом решениях и т. д.;
- информационное обеспечение работы экспертных советов и Президиума ВАК, включая отображение информации по рассматриваемым аттестационным делам, что позволяет отказаться от "бумажных" копий документов.

Следует особо отметить, что ИТС в полной мере реализует функции электронного административного регламента по присуждению ученых степеней, разработка которого была начата Рособрнадзором, а также функций "Единой информационной системы учета диссертаций", разрабатываемой по заказу Федерального агентства по образованию.

Дальнейшее развитие информационного обеспечения ВАК и применения современных информационных технологий в системе аттестации научных и научно-педагогических кадров следует вести по пути расширения функций и возможностей информационной системы ВАК.

Развитие информационной системы ВАК требует решения следующих задач:

- организационно-правовая поддержка организации электронного документооборота аттестационных дел в системе государственной аттестации научных и научно-педагогических кадров;
- расширение спектра информационных ресурсов, используемых при экспертизе и аттестации, в том числе базы данных ИК диссертаций, электронной библиотеки научных журналов, Российского индекса цитируемости и т. д.;
- создание условий для использования системы "Антиплагиат" — формирование баз данных авторефератов и электронных версий диссертаций;
- разработка БД нормативных документов по вопросам аттестации научных и научно-педагогических кадров и средств доступа к ней через справочники и поиска по содержанию документов;
- завершение формирования "Электронной картотеки специалистов", прошедших государственную аттестацию;
- проведение модернизации БД диссертационных советов;
- разработка программного обеспечения (ПО) электронного документооборота аттестационных дел по присвоению ученых званий (по аналогии с аттестационными делами по присуждению ученых степеней);
- разработка ПО электронного документооборота материалов диссертационных советов.

Для расширения возможностей информационного комплекса ВАК целесообразно рассмотреть вопросы получения необходимой для экспертизы диссертационных работ достоверной информации о результатах в данной области знаний. Развитие информационных сетевых технологий, размещение авторефератов диссертаций на сайте ВАК и организаций, наличие баз данных и электронных библиотек научной информации (электронные каталоги РГБ, ГПНТБ, базы данных ВНТИЦ, ВИНИТИ и т. д.) позволяет организовать оперативный тематически полный поиск аналогичных и близких подходов к решению научных проблем. В настоящее время сложились все условия для комплексной *системной* автоматизации процессов, связанных с аттестацией и подготовкой научных кадров. Формирование информационных ресурсов научной информации делает очевидным необходимость и перспективность более полной интеграции в процесс защиты и экспертизы диссертации технологии подготовки и открытой Интернет-публикации сведений о теоретических и практических результатах научного исследования (НИ), а также о защите автором диссертации. В частности, "новая" *аналитическая* информационная карта диссертации (ИКД) могла бы объединить свойства двух, на сегодня логически и физически отдельных, но имеющих одинаковое "информирующее" назначение (сообщать о наличии работы и об основных ее положениях) вторичных документов: автореферата, представляющего собой аналитический образ исследования, и ИКД — его поискового образа. При этом онлайновый режим формирования такого информационного образа (аналитической ИКД) позволит ввести элементы формализма достаточно естественным путем, а оперативный контент-анализ и автоматическое выделение системой потенциальных ключевых понятий из вводимых автором формулировок автореферата стимулирует автора избегать обтекаемых сложносочиненных фраз и сосредоточиться на существенных и отличительных понятиях и их взаимосвязях. Разработка и использование такой структурированной онтологической формы представления информацион-

но-аналитических материалов (структурированного автореферата, отзывов оппонентов и ведущей организации, заключений диссертационных и экспертных советов) позволит не только облегчить эксперту процесс оценки научного результата, но также создаст информационную (аналитическую) базу знаний для формирования оценок состояния и перспектив соответствующих научных направлений.

В части развития сайта ВАК и его преобразования в портал по вопросам подготовки и аттестации научных кадров ближайшими задачами являются:

- модернизация сайта ВАК;
- пополнение его информационными ресурсами и функциями;
- расширение возможностей по публикации объявлений о защитах кандидатских диссертаций.

Другой полезной функцией официального сайта ВАК могла бы стать организация подписки, предоставляющая посетителям сайта возможность подписаться на объявления о защитах по интересующим их специальностям и/или в интересующих их диссертационных советах. При появлении на официальном сайте объявления о защите по указанной специальности или в указанном совете подписавшемуся будет направляться электронное письмо с краткой информацией о защите и со ссылкой на размещенное на сайте объявление и т. д.

Реализация рассмотренных решений и предложений по их развитию, как нам представляется, могут способствовать решению задач по использованию информационных технологий в работе ВАК, которые были определены в интервью председателя ВАК М. П. Кирпичникова и его заместителя Г. И. Савина газете "Поиск" от 24 мая 2008 г.: "первая —обеспечить качественную оценку квалификации наших научно-педагогических кадров, вторая — сделать так, чтобы оценка происходила в достаточно оперативном режиме, процедура ее была прозрачной, а научные кадры не пребывали долгое время в неведении: что же происходит с оценкой их работы? Третья задача — предоставить возможность научному сообществу оперативно пользоваться материалами уже защищенных диссертационных работ, а не складывать их после защит просто в шкаф".

Использованные источники

1. **Председатель** ВАК Михаил Кирпичников: В 2006 году впервые удалось остановить лавинообразный рост диссертаций. http://www.polit.ru/science/2007/12/17.vak.html

2. **ВАК против** "оборотней со степенью". Интервью с председателем Высшей аттестационной комиссии, академиком Михаилом Кирпичниковым. http://www.polit.ru/science/2007/07/25.

3. **Раздел** "Информационные ресурсы. Управление государственной аттестации научных и научно-педагогических работников" http://www.obrnadzor.gov.ru/inform_resources/.

4. **Голицына О. Л., Максимов Н. В., Сергиевский Г. М., Строгонов В. И.** Об одном подходе к построению интегрированной системы информационной поддержки процессов подготовки и аттестации научных кадров. Системы управления и информационные технологии. 2006. № 2 (24).

5. **Горбунов-Посадов М. М., Полилова Т. А., Строгонов В. И.** Мониторинг научной аттестации в Интернете. Решение проблемы. Информационные технологии и вычислительные системы. 2007. № 1. С. 72—76.

УДК 621.513.6

В. И. Джиган, д-р техн. наук., гл. науч. сотр. ГУП,
"Научно-производственный центр "Электронные вычислительно-информационные системы"
(ГУП НПЦ "ЭЛВИС"), г. Москва
И. Д. Плетнева, ст. преподаватель,
Московский государственный институт электронной техники (технический университет)

Линейно-ограниченный нормализованный алгоритм по критерию наименьшего среднего квадратичного отклонения для цифровой адаптивной антенной решетки

Рассматривается использование линейных ограничений в нормализованном градиентном алгоритме, который применяется для управления адаптивными антенными решетками, функционирующими по критерию постоянства модуля информационных символов. Приводятся вычислительная процедура алгоритма и оценка требуемого числа арифметических операций. С помощью моделирования сравнивается эффективность полученного алгоритма и линейно-ограниченного алгоритма по критерию наименьших квадратов.

Ключевые слова: *адаптивная антенная решетка, линейные ограничения критерия постоянства модуля информационных сигналов, нормализованный алгоритм по критерию наименьшего среднего квадратичного отклонения, созвездие.*

Введение

Трудно представить современные информационные технологии без применения цифровой связи, посредством которой осуществляется передача данных от источника информации к потребителю. Среди различных видов связи все более распространенной становится беспроводная связь. При эксплуатации систем беспроводной связи часто возникает задача борьбы с помехами, находящимися в одной полосе частот с полезным сиг-

налом. В этом случае полезный сигнал может быть отделен от помех только за счет пространственных различий источников сигналов, что приводит к необходимости использования в качестве антенн адаптивных антенных решеток (ААР) [1]. ААР подавляют помехи за счет изменения формы диаграммы направленности (ДН). В направлениях на источники помех в ДН формируются провалы, что обеспечивает ослабление этих помех в выходном сигнале ААР.

В беспроводной связи в основном применяются решетки с небольшим (10—20) числом антенн N. Это позволяет использовать в качестве алгоритмов управления ААР как простые адаптивные алгоритмы с вычислительной сложностью $O(N)$, так и сложные алгоритмы с вычислительной сложностью $O(N^2)$. Такие алгоритмы уже реализуемы на современной цифровой элементной базе [2, 3]. Под вычислительной сложностью понимается число арифметических операций, необходимых для выполнения одной итерации алгоритма.

Использование адаптивных алгоритмов в антенных решетках обычно требует наличия тренировочных последовательностей, т. е. так называемого образцового сигнала адаптивного фильтра. В ряде систем такие последовательности не предусмотрены стандартом связи, а потому отсутствуют. В этом случае в качестве алгоритмов управления ААР могут быть использованы адаптивные алгоритмы на основе минимизации квадратичных функционалов с линейными ограничениями (*Linearly Constrained*, LC) [4] или на основе критерия постоянства модуля информационных сигналов (*Constant Modulus*, CM) [5].

Каждый из этих алгоритмов имеет свои недостатки. В работе [6] было показано, что LC-алгоритмы чувствительны к коррелированным помехам, например, помехам, обусловленным многолучевым распространением полезного сигнала. Эти помехи плохо подавляются с помощью данного алгоритма. В аналогичных условиях CM-алгоритмы могут "хвататься" за коррелированные помехи и подавлять полезный сигнал, формируя основной лепесток ДН в направлении помехи и провал в направлении полезного сигнала [7], т. е. помеха может усиливаться, а полезный сигнал — подавляться. Принудительная ориентация основного лепестка ДН CM ААР в направлении на источник полезного сигнала путем задания начальных значений весовых коэффициентов ААР при этом часто оказывается неэффективной, так как коэффициенты меняются в процессе адаптации.

Если направление на источник полезного сигнала известно, то введение линейных ограничений в адаптивный алгоритм на основе CM-критерия позволяет ААР эффективно функционировать при наличии коррелированных помех. В этом случае ограничения удерживают основной лепесток ДН ААР в направлении на полезный сигнал в течение всего времени адаптивной фильтрации, независимо от значений весовых коэффициентов, вычисляемых с помощью CM-алгоритма в целях подавления помех [8].

Известно, что адаптивные алгоритмы на основе CM-критерия характеризуются многоэкстремальной поверхностью минимизируемых функционалов в пространстве весовых коэффициентов адаптивного фильтра, что вызывает неоднозначность в выборе шага сходимости в градиентных алгоритмах, а также не позволяет использовать более эффективные, но и более сложные, нормализованные градиентные алгоритмы, или алгоритмы на основе критерия наименьших квадратов, поскольку эти алгоритмы предназначены для минимизации квадратичных функционалов.

В работе [9] CM-критерий был сведен к квадратичному функционалу, для минимизации которого использовался рекурсивный алгоритм по критерию наименьших квадратов (*Recursive Least Squares*, RLS) на основе леммы об обращении матриц [10]. В работе [11] было показано, что этот алгоритм может быть неустойчивым при решении задачи управления ААР, и в качестве альтернативных рассматривались другие рекурсивные алгоритмы на основе QR-разложения матриц [12].

LC NLMS-алгоритм

В настоящей работе рассматривается применение LC нормализованного алгоритма по критерию наименьшего среднего квадратичного отклонения (*Normalized Least Means Squares*, NLMS) в задаче управления ААР, функционирующей на основе квадратичного CM-функционала [9], а также сравнивается эффективность этого алгоритма с LC RLS-алгоритмами.

Адаптивные алгоритмы на основе CM-критерия применяются для обработки сигналов, характеризуемых постоянным значением модуля информационных символов, например *Quadrature Phase Shift Keying* (QPSK), (рис. 1). Каждый такой символ a_i обладает свойством $|a_i| = \sqrt{a_i^* a_i} = s = \text{const}$, где * — обозначает операцию комплексного сопряжения. Значение s является известным на приемной стороне. В общем случае CM-критерий формулируется как

$$J(p, q) = E[|s^p - |y(k)|^p|^q], \qquad (1)$$

а соответствующие ему адаптивные алгоритмы обозначаются как CM(p, q). Здесь $E[\cdot]$ — операция усреднения; $y(k) = \boldsymbol{h}_N^H(k-1)\boldsymbol{x}_N(k)$ — выходной сигнал антенной решетки (рис. 2); $\boldsymbol{h}_N(k) = [h_1(k), ..., h_n(k), ..., h_N(k)]^T$ — вектор весовых коэффициентов, $\boldsymbol{x}_N(k) = [x_1(k), ..., x_n(k), ..., x_N(k)]^T$ — вектор пространственно-временных отсчетов сигналов; k — индекс дискретного време-

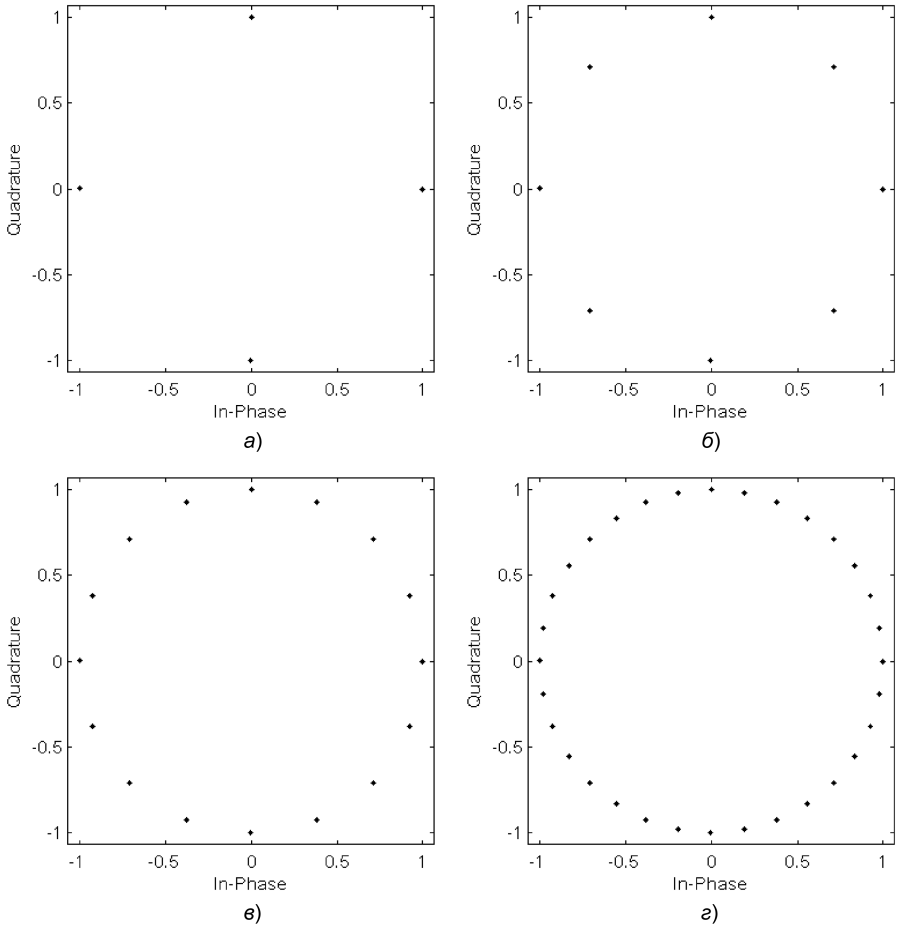

Рис. 1. Созвездия:
а — QPSK-4; *б* — QPSK-8; *в* — QPSK-16; *г* — QPSK-32

ни. Матрицы обозначаются полужирными прописными символами, а векторы — полужирными строчными символами. Нижний индекс N в обозначении векторов и матриц показывает, что рассматривается квадратная матрица с числом элементов $N \times N$ или вектор с числом элементов N, верхние индексы T и H обозначают операции транспонирования и эрмитова сопряжения (транспонирования и комплексного сопряжения) векторов и матриц.

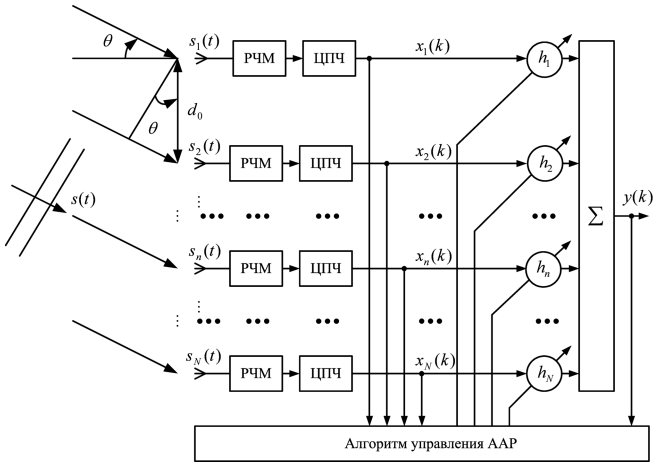

Рис. 2. ААР: РЧМ — радиочастотный модуль; ЦПЧ — цифровой преобразователь частоты

СМ-критерий (1) является многоэкстремальным. Согласно работе [9] при $p = 2$ и $q = 2$ уравнение (1) может быть преобразовано в унимодальный квадратичный функционал в пространстве весовых коэффициентов $\boldsymbol{h}_N(k)$:

$$\boldsymbol{J}'(k) = \sum_{i=1}^{k} \lambda^{k-i} \times \times |s^2 - \boldsymbol{h}_N^H(k)\boldsymbol{z}_N(i)|^2, \qquad (2)$$

где $\boldsymbol{z}_N(k) = \boldsymbol{x}_N(k)y^*(k)$, а $(1 - -0{,}4/N) \leqslant \lambda < 1$ — параметр экспоненциального взвешивания, предназначенный для слежения в небольших пределах за медленно изменяющимися сигналами. Это позволяет использовать любой из адаптивных алгоритмов минимизации квадратичных функционалов в качестве алгоритма минимизации функционала (2). Ниже показано, как для решения этой задачи можно использовать модификацию LC NLMS-алгоритма [13].

При приеме ААР полезного сигнала с известного направления θ_S в алгоритме адаптивной фильтрации можно задать одно линейное ограничение

$$\boldsymbol{c}_N^H \boldsymbol{h}_N(k) = f, \qquad (3)$$

обеспечивающее ориентацию основного лепестка ДН в направлении θ_S и ее значение в этом направлении, равное $F(\theta) = \boldsymbol{h}_N^H \boldsymbol{c}_N(\theta_S) = f = |f|e^{i\varphi}$. Вектор ограничений $\boldsymbol{c}_N$ задается следующим образом. Пусть плоская электромагнитная волна $s(t)$ принимается линейной ААР с направления, определяемого углом θ (рис. 2). Расстояние между антеннами в ААР обычно выбирается равным $d_0 = 0{,}5\lambda_0 = 0{,}5v/f_0$, где λ_0 — длина волны несущего сигнала на частоте f_0; v — скорость распространения электромагнитной волны в свободном пространстве, равная скорости света. Задержка сигнала в n-й антенне относительно опорной антенны, например с номером 1, определяется как $\tau_n = d_0(n-1)\sin(\theta)/v$, фазовый набег — как

$$\omega_0\tau_n = 2\pi d_0 f_0(n-1)\sin(\theta)/v = = 2\pi d_0(n-1)\sin(\theta)/\lambda = \psi_n,$$

а вектор ограничений для направления θ_S — как

$$\boldsymbol{c}_N(\theta_S) = [c_1(\theta_S), ..., c_n(\theta_S), ..., c_N(\theta_S)]^T = \\ = [e^{i\psi_1^{(s)}}, ..., e^{i\psi_n^{(s)}}, ..., e^{i\psi_N^{(s)}}]^T.$$

С учетом (2), (3) и [13], LC CM(2,2) NLMS-алгоритм для управления ААР формулируется как

$$\boldsymbol{h}_N(k) = \mathbf{P}_N[\boldsymbol{h}_N(k-1) + \mu \boldsymbol{z}_N(k)\tilde{\alpha}_N(k)] + \boldsymbol{a}_N; \quad (4)$$

$$\mathbf{P}_N = \mathbf{I}_N - \boldsymbol{c}_N(\boldsymbol{c}_N^H \boldsymbol{c}_N)^{-1}\boldsymbol{c}_N^H, \quad (5)$$

$$\boldsymbol{a}_N = \boldsymbol{c}_N(\boldsymbol{c}_N^H \boldsymbol{c}_N)^{-1} f, \quad (6)$$

$$\tilde{\alpha}_N(k) = [\boldsymbol{z}_N^H(k)\mathbf{P}_N\boldsymbol{z}_N(k) + \delta^2]^{-1}\alpha_N^*(k), \quad (7)$$

$$\alpha_N = s^2 - \boldsymbol{h}_N^H(k-1)\boldsymbol{z}_N(k), \quad (8)$$

где $\mathbf{I}_N$ — единичная матрица; $0 < \mu \leqslant 1$ — масштабирующий множитель динамического шага сходимости, определяемого как $[\boldsymbol{z}_N^H(k)\mathbf{P}_N\boldsymbol{z}_N(k) + \delta^2]^{-1}$ в уравнении (7). Параметр $\delta^2 \geqslant 0{,}01\sigma_z^2$ используется для регуляризации деления (предотвращения деления на ноль) в уравнении (7). Здесь σ_z^2 — дисперсия сигналов $z_n(k) = x_n(k)y^*(k)$, образующих вектор $\boldsymbol{z}_N(k)$.

Вследствие умножений на матрицу $\mathbf{P}_N$ оценка вычислительной сложности LC NLMS-алгоритма (4)—(8) равна $O(N^2)$ арифметическим операциям, в отличие от NLMS-алгоритма без ограничений, вычислительная сложность которого равна $O(N)$ арифметическим операциям.

В то же время уравнение (4), подобно RLS-алгоритмам с линейными ограничениями [12], может быть упрощено как

$$\boldsymbol{h}_N(k) = [\mathbf{I}_N - \boldsymbol{c}_N(\boldsymbol{c}_N^H \boldsymbol{c}_N)^{-1}\boldsymbol{c}_N^H] \times \\ \times [\boldsymbol{h}_N(k-1)\mu\boldsymbol{z}_N(k)\tilde{\alpha}_N(k)] + \boldsymbol{c}_N(\boldsymbol{c}_N^H \boldsymbol{c}_N)^{-1} f = \\ = [\mathbf{I}_N - \boldsymbol{q}_N\boldsymbol{c}_N^H][\boldsymbol{h}_N(k-1) + \mu\boldsymbol{z}_N(k)\tilde{\alpha}_N(k)] + \\ + \boldsymbol{q}_N f = \boldsymbol{h}_N(k-1) + \mu\boldsymbol{z}_N(k)\tilde{\alpha}_N(k) + \\ + [f - \boldsymbol{c}_N^H\{\boldsymbol{h}_N(k-1) + \mu\boldsymbol{z}_N(k)\tilde{\alpha}_N(k)\}] = \\ = \boldsymbol{h}'_N(k) + \boldsymbol{q}_N[f - \boldsymbol{c}_N^H\boldsymbol{h}'_N(k)],$$

где $\boldsymbol{h}'_N(k) = \boldsymbol{h}_N(k-1) + \mu\boldsymbol{z}_N(k)\tilde{\alpha}_N(k)$ и $\boldsymbol{q}_N = \boldsymbol{c}_N(\boldsymbol{c}_N^H\boldsymbol{c}_N)^{-1}$. Таким образом, в полученном уравнении отсутствует операция умножения на матрицу $\mathbf{P}_N$, так как вектор $\boldsymbol{q}_N$ не зависит от k и может быть вычислен заранее. Результирующий многоканальный LC NLMS-алгоритм для ААР на основе CM(2,2)-критерия приведен в таблице.

LC CM (2,2) NLMS-алгоритм

Вычисления	Ссылки
Инициализация: $\boldsymbol{x}_N(0) = \mathbf{0}_N$, $\boldsymbol{q}_N = \boldsymbol{c}_N(\boldsymbol{c}_N^H\boldsymbol{c}_N)^{-1}$, $\boldsymbol{h}_N(0) = \boldsymbol{q}_N f$	(1.0)
For $k = 1, 2, ..., K$	
$\boldsymbol{z}_N(k) = \boldsymbol{x}_N(k)\boldsymbol{x}_N^H(k)\boldsymbol{h}_N(k-1)$	(1.1)
$\alpha_N(k) = s^2 - \boldsymbol{h}_N^H(k-1)\boldsymbol{z}_N(k)$	(1.2)
$\alpha_N(k) = [\boldsymbol{z}_N^H(k)\mathbf{P}_N\boldsymbol{z}_N(k) + \delta^2]^{-1}\alpha_N^*(k)$	(1.3)
$\boldsymbol{h}'_N(k) = \boldsymbol{h}_N(k-1) + \mu\boldsymbol{z}_N(k)\alpha_N(k)$	(1.4)
$\boldsymbol{h}_N(k) = \boldsymbol{h}'_N(k) + \boldsymbol{q}_N[f - \boldsymbol{c}_N^H\boldsymbol{h}'_N(k)]$	(1.5)
End for k	

Вычислительная сложность алгоритма

В уравнении (1.3) алгоритма (см. таблицу) имеется только одна операция с арифметической сложностью $O(N^2)$ вместо двух таких операций в уравнениях (4) и (7) исходного алгоритма. Вычислительная сложность алгоритма (см. таблицу) равна $N^2 + 7N + 1$ операциям умножения, $N^2 + 6N + 1$ операциям сложения и одной операции деления.

Для решения рассмотренной задачи можно также использовать LC RLS-алгоритмы с арифметической сложностью $O(N^2)$ [12]. Среди этих алгоритмов минимальную сложность имеет алгоритм на основе обратного QR-разложения с использованием вращений Гивенса без операций извлечения квадратного корня. Его сложность равна $2{,}5N^2 + 19N + 1$ операциям умножения, $1{,}5N^2 + 12{,}5N + 5$ операциям сложения и $N + 1$ операциям деления. Эта сложность в части операций умножения примерно в 2,5 раза больше сложности алгоритма (см. таблицу), а в части операций сложения и деления в 1,5 и в N раз, соответственно.

Таким образом, представленный в настоящей работе LC CM(2,2) NLMS-алгоритм характеризуется меньшей вычислительной сложностью по сравнению с самым простым LC CM(2,2) RLS-алгоритмом.

Моделирование алгоритмов

Для демонстрации эффективности рассмотренных алгоритмов (LC CM(2,2) NLMS и LC CM(2,2) RLS) проведено компьютерное моделирование процессов подавления помех с помощью ААР. Моделирование проводилось в информационной полосе частот [14]. В качестве полезного сигнала использовались сигналы QPSK-4, QPSK-8, QPSK-16 и QPSK-32 с $|a_i| = 1$. Направление на источник полезного сигнала было выбрано как $\theta_S = 0^\circ$, а направления на источники помех — как $\theta_{J_1} = 21^\circ$ и $\theta_{J_2} = -21^\circ$. Направления на источники помех совпадали с максимумами первых двух боковых лепестков ДН неадаптивной антенной решетки. Коррелированная помеха J_1 представляла собой копию полезного сигнала, задержанного

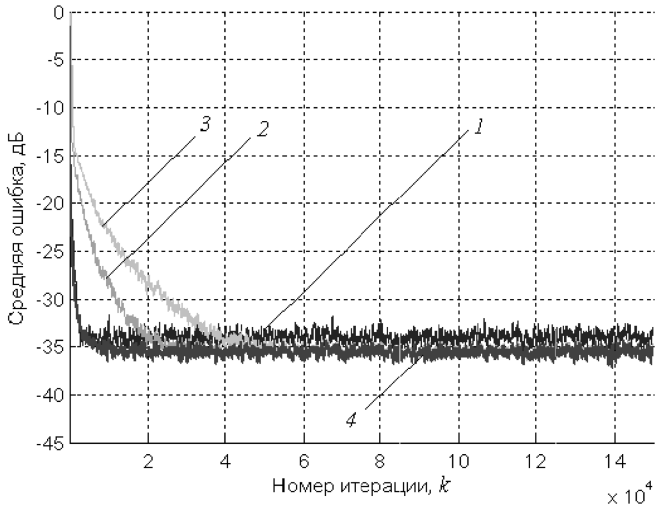

Рис. 3. Переходной процесс при $N = 8$, QPSK-4:
1 — LC CM(2,2) NLMS-алгоритм, $\mu = 0{,}1$; 2 — LC CM(2,2) NLMS-алгоритм, $\mu = 0{,}01$; *3* — LC CM(2,2) NLMS-алгоритм, $\mu = 0{,}005$; *4* — RLS-алгоритм

на половину длительности информационного символа. Уровень этой помехи был на 3 дБ ниже уровня полезного сигнала. Помеха J_2 моделировалась белым шумом. Ее уровень на 20 дБ превышал уровень полезного сигнала. При моделировании были использованы следующие параметры адаптивных алгоритмов: $N = 8$, $\delta^2 = 0{,}01$, $\lambda = 0{,}9999$. Все вычисления проводились в арифметике с плавающей точкой. Для обеспечения задержки коррелированной помехи относительно полезного сигнала на длительности информационного символа производились два отсчета. Отсчеты совпадали с итерациями рассматриваемых алгоритмов k, число которых равнялось $K = 15 \cdot 10^4$.

На рис. 3 показаны переходные процессы LC CM(2,2) NLMS-алгоритма при трех значениях параметра μ и LC CM(2,2) RLS-алгоритма на основе обратного QR-разложения с использованием вращений Гивенса без операций извлечения квадратного корня. Значения, представленные на графиках, определялись путем усреднения параметра $|\alpha_N(k)|$ на скользящем окне в 256 отсчетов. Эти значения в логарифмическом масштабе определялись по отношению к s. Из рис. 3 следует, что с уменьшением μ среднее значение $|\alpha_N(k)|$, достигаемое с помощью LC CM(2,2) NLMS-алгоритма приближается к значению, достигаемому с помощью LC CM(2,2) RLS-алгоритма. Однако длительность переходного процесса в LC CM(2,2) NLMS-алгоритме увеличивается с уменьшением μ. Такое поведение LC CM(2,2) NLMS-алгоритма при минимизации квадратичного функционала согласуется с общей теорией градиентных алгоритмов адаптивной фильтрации.

Значения ДН на каждой итерации рассматриваемых алгоритмов в направлениях источников принимаемых сигналов показаны на рис. 4, *а—г*. Видно, что в LC CM(2,2) NLMS-алгоритме существует большой разброс в мгновенных значениях ДН в направлениях на источники помех. Однако эти значения ДН (−50...−120 дБ) позволяют подавлять некоррелированные и коррелированные помехи до уровня, обеспечивающего различение информационных символов в выходном сигнале ААР. При уменьшении параметра μ провалы в ДН ААР в направлениях помех в среднем становятся глубже и приближаются к значениям, достигаемым с помощью LC CM(2,2) RLS-алгоритма. Ценой этого результата является более длинный переходный процесс в LC CM(2,2) NLMS-алгоритме по сравнению с LC CM(2,2) RLS-алгоритмом. Результаты, аналогичные рис. 4, были получены также и для других видов QPSK-сигналов (см. рис. 1).

Примеры ДН, полученных с помощью LC CM(2,2) NLMS-алгоритма для ААР с $N = 8$ и $N = 16$, приведены на рис. 5, *а*, *б*. ДН неадаптивной антенной решетки показана серой линией. Направления на источники сигналов обозначены

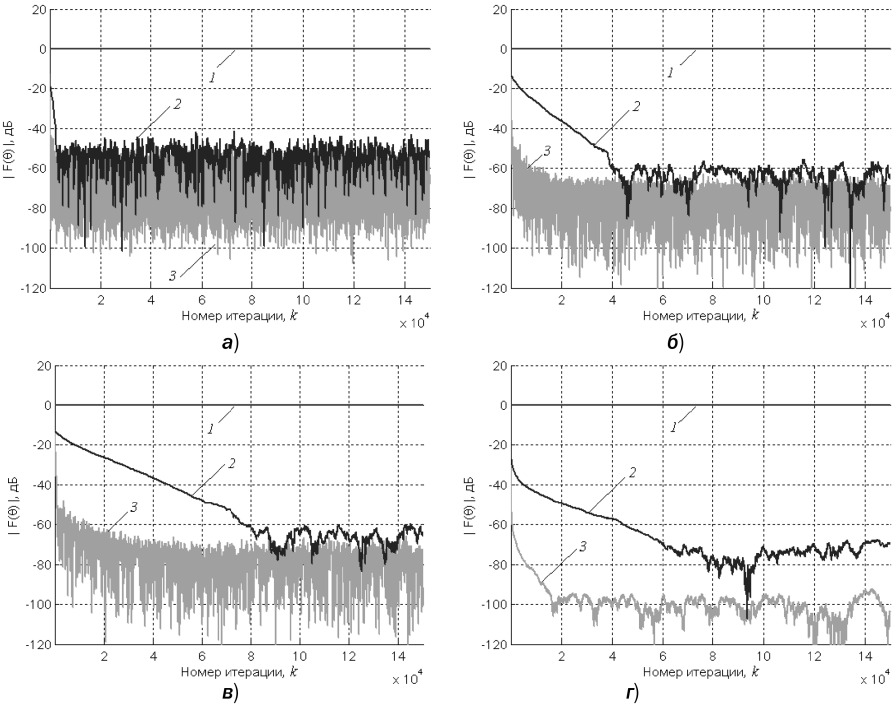

Рис. 4. Переходной процесс при $N = 8$, QPSK-4
($1 - \theta_S = 0°$; $2 - \theta_{J_1} = -21°$; $3 - \theta_{J_2} = 21°$):
а — LC CM(2,2) NLMS-алгоритм, $\mu = 0{,}1$; *б* — LC CM(2,2) NLMS-алгоритм, $\mu = 0{,}01$; *в* — LC CM(2,2) NLMS-алгоритм, $\mu = 0{,}005$; *г* — RLS-алгоритм

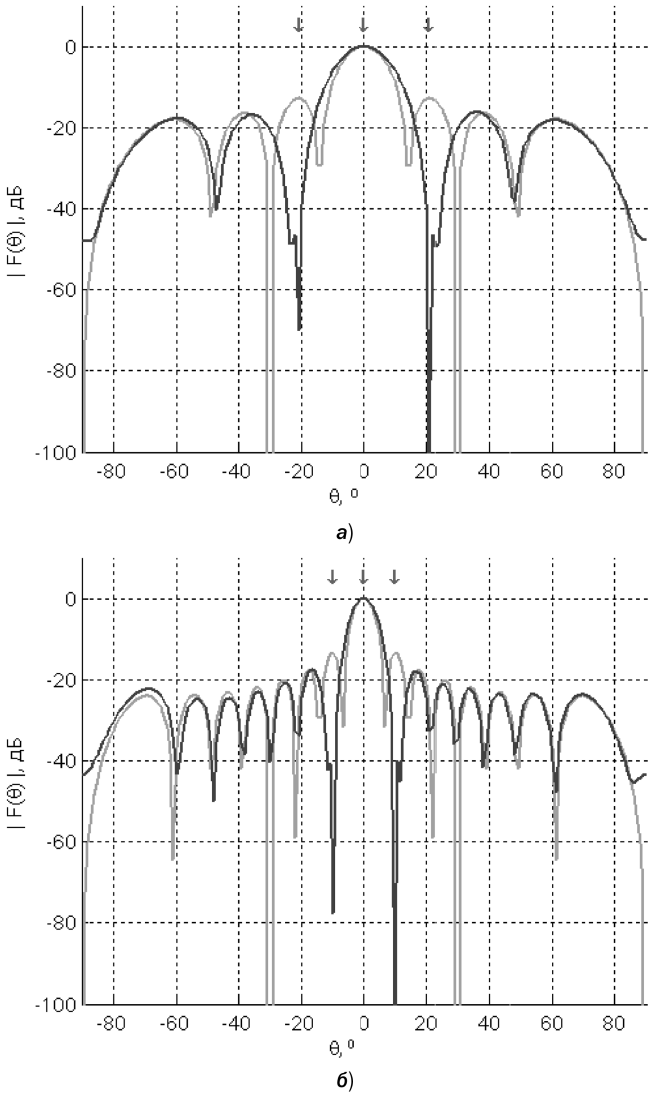

Рис. 5. Диаграмма направленности в установившемся режиме: $N = 16$, QPSK-4:
а — $\theta_S = 0^\circ$, $\theta_{J_1} = -21^\circ$, $-\theta_{J_2} = 21^\circ$; *б* — $\theta_S = 0^\circ$, $\theta_{J_1} = -10^\circ$, $-\theta_{J_2} = 10^\circ$

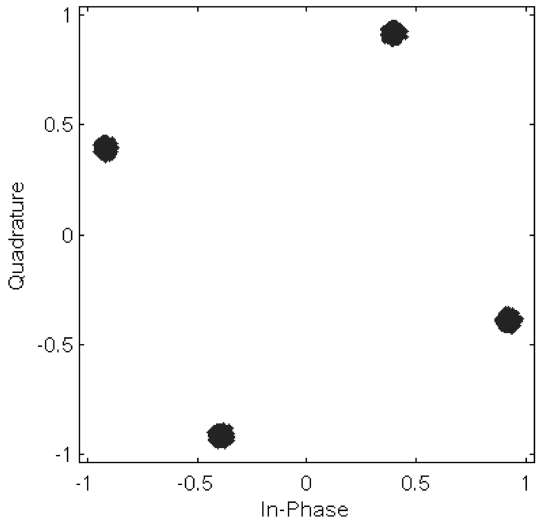

Рис. 6. Созвездие QPSK-4 на выходе ААР

стрелками в верхней части рисунков. В ААР с $N = 16$ помехи располагаются в направлениях $\theta_{J_1} = -10^\circ$, $-\theta_{J_2} = 10^\circ$. Параметры помех были аналогичны помехам, используемым при моделировании ААР с $N = 8$.

Эффективность ААР в терминах достижимых уровней ДН в направлениях источников помех при $N = 16$ аналогична ААР с $N = 8$. ААР с различным числом антенных элементов различаются лишь длительностью переходных процессов. С увеличением N длительность переходного процесса увеличивается.

Таким образом, если длительность переходного процесса ААР не является критичной, то при заданном N и соответствующем выборе параметра μ в среднем можно достигнуть подавления стационарных помех с помощью LC CM(2,2) NLMS-алгоритма, близкого к подавлению помех с помощью LC CM(2,2) RLS-алгоритмов.

Аналогичную эффективность (см. рис. 3—5) также демонстрируют и одноименные алгоритмы без ограничений в случае, если не наблюдается явление "захвата помех". Однако при этом созвездия информационных символов на выходе ААР приобретают фазовый сдвиг (рис. 6), равный $\varphi_S = \text{arctg}\{\text{Im}[F(\theta_S)]/\text{Re}[F(\theta_S)]\}$, что не позволяет правильно различать принимаемые информационные символы. Для компенсации φ_S в ААР и эквалайзерах на основе CM-критерия обычно используются цепи фазовой автоподстройки [5].

При введении линейных ограничений в CM-алгоритм эту компенсацию можно осуществить весьма просто, задав ограничение f, равным действительному числу, т. е. $F(\theta_S) = \boldsymbol{h}_N^H \boldsymbol{c}_N(\theta_S) = |f|$. Результаты моделирования подтверждают, что введение такого ограничения обеспечивает на каждой итерации не только требуемые ориентацию θ_S и уровень f основного лепестка ДН ААР, но и правильную ориентацию созвездия информационных символов, совпадающую с ориентацией символов алфавита передаваемого сообщения (см. рис. 1).

Таким образом, при малых значениях параметра μ LC CM(2,2) NLMS-алгоритм позволяет достигать подавления помех в ААР, близкого к подавлению помех, достигаемому с помощью LC CM(2,2) RLS-алгоритмов. Различие между алгоритмами заключается в длительности переходного процесса и вычислительной сложности.

Материалы статьи могут представлять интерес для специалистов в области цифровой связи, ААР и цифровой обработки сигналов.

Список литературы

1. **Godara L. C.** Application of antenna arrays to mobile communications. II. Beam-forming and direction-of-arrival considerations // Proc. of the IEEE. 1997. Vol. 85. № 8. P. 1195—1245.

2. **Солохина Т., Александров Ю., Петричкович Я.** Сигнальные контроллеры компании "ЭЛВИС": первая линейка отече-

ственных DSP // Электроника: Наука, Технология, Бизнес. 2005. № 7. С. 70—77.
3. **Джиган В. И.** Прикладная библиотека адаптивных алгоритмов // Электроника: Наука, Технологии, Бизнес. 2006. № 1. С. 60—65.
4. **Frost O. L.** An algorithm for linearly constrained adaptive array processing // Proc. of the IEEE. 1972. Vol. 60. № 8. P. 926—935.
5. **Treichler J., Larimore M.** New processing techniques based on the constant modulus adaptive algorithm // IEEE Trans. Acoustics, Speech, and Signal Processing. 1985. Vol. 33. N 2. P. 420—431.
6. **Shan T.-J., Kailath T.** Adaptive beamforming for coherent signals and interference // IEEE Trans. Acoustics, Speech, and Signal Processing. 1985. Vol. 33. № 3. P. 527—536.
7. **Treichler J., Larimore M.** The tone capture properties of CMA-based interference suppressors // IEEE Trans. Acoustics, Speech, and Signal Processing. 1985. Vol. 33. № 4. P. 946—958.
8. **Rude M. J., Griffiths L. J.** Incorporation of linear constraints into the constant modulus algorithm // International Conference on Acoustics, Speech, and Signal Processing. 1989. Vol. 2. P. 968—971.
9. **Chen Y., Le-Ngoc T., Champagne B., Xu C.** Recursive least squares constant modulus algorithm for blind adaptive array // IEEE Trans. Signal Processing. 2004. Vol. 52. № 5. P. 1452—1456.
10. **Giordano A. A., Hsu F. M.** Least square estimation with application to digital signal processing. Canada, Toronto: John Wiley and Sons, Inc., 1985. 412 p.
11. **Джиган В. И., Плетнева И. Д.** Алгоритмы адаптивной фильтрации на основе QR-разложения для антенных решеток систем цифровой связи // Цифровая обработка сигналов. 2007. № 4.
12. **Джиган В. И.** Многоканальные RLS- и быстрые RLS-алгоритмы адаптивной фильтрации // Успехи современной радиоэлектроники. 2004. № 11. С. 48—77.
13. **Apolinario J. A., Werner S., Diniz P. S. R., Laakso T. I.** Constrained normalized adaptive filters for CDMA mobile communication // Proc. of the 9-th European Signal Processing Conference. — Island of Rhodes, Greece, 1998. 4 p.
14. **Плетнева И. Д., Джиган В. И.** Моделирование обработки сигналов в цифровых антенных решетках // Исследования в области цифровых систем связи. М.: Изд. МИЭТ, 2007.

ОБМЕН ОПЫТОМ

УДК 004.9

В. И. Аникин, д-р техн. наук, проф.,
нач. отд. информатизации,
О. В. Аникина, ассистент,
Тольяттинский государственный
университет сервиса
E-mail: anikin@tolgas.ru

Эффективная техника создания табличных моделей в Microsoft Excel

Обсуждается оригинальная техника создания алгоритмических табличных моделей в Microsoft Excel без программирования на языке VBA. Идея техники базируется на аналогии между процессом пересчета содержимого ячеек электронной таблицы и выполнением потока команд компьютерной программы, а также на принципах структурного программирования. Техника моделирования может быть полезной для менеджеров, специалистов в прикладных областях, преподавателей, аспирантов и студентов высших учебных заведений.

Ключевые слова: *алгоритмическое табличное моделирование, табличная модель, Microsoft Excel, программирование без программирования, структурное программирование.*

В данной работе рассматривается эффективная техника создания алгоритмических табличных моделей в Excel по принципу "программирование без программирования", которую мы назвали техникой *алгоритмического табличного моделирования* (АТМ) [1] по двум причинам: табличная модель создается на основе алгоритма обработки данных, который проектируется заранее, на стадии формализации задачи; внутренняя логика работы модели реализуется чисто табличными средствами без программирования на языке VBA.

Идея техники базируется на аналогии между процессом пересчета содержимого ячеек электронной таблицы (ЭТ) и выполнением потока команд обычной компьютерной программы, а также на принципах структурного программирования. Действительно, при любом изменении ячеек ЭТ, Excel автоматически выполняет пересчет содержимого зависимых ячеек, следуя графу связей между ними. Аналогичным образом последовательно выполняются команды обычной компьютерной программы в соответствии с алгоритмом обработки данных. Более того, используя формулы и функции Excel, а также специальным образом организуя связи между ячейками модели, в табличной модели можно реализовать все базовые управляющие конструкции структурного программирования: последовательности, ветвления, циклы и подпрограммы.

Символические переменные модели. Ячейки, составляющие табличную модель, хранят полную "историю" последовательной обработки модельных данных. Каждое входное, промежуточное или выходное значение данных, вводимое пользователем или получаемое в ходе моделирования, хранится в отдельной ячейке табличной модели. Можно считать, что общее число элементов данных модели равно числу задействованных в ней ячеек, при этом роль имени ячейки в табличной модели выполняет ее уникальный адрес.

При создании алгоритмических табличных моделей вместо уникальных адресов ячеек удобно пользоваться символическими именами. Отме-

тим, что символические имена оказываются наиболее полезными на начальной стадии проектирования табличной модели. В это время разработчик еще ничего не знает о том, какие конкретно ячейки ЭТ будут задействованы в создаваемой модели, и возникает насущная потребность в символическом описании ячеек, без привязки к их конкретным адресам. Введение символических переменных способствует также разработке наглядного и удобного пользовательского интерфейса, так как их можно использовать в качестве заголовков ячеек, строк и столбцов таблиц, в комментариях, при документировании модели.

В алгоритмических табличных моделях, как и в обычных программах, следует различать символические переменные двух типов: *скалярные переменные*, которые занимают в табличной модели одну ячейку, и *векторные переменные*, занимающие интервал ячеек. При этом, любое изменение скалярной переменной модели должно фиксироваться в другой переменной, фактически эта скалярная переменная становится векторной, в обычных же программах значение скалярной переменной в ходе вычислений может многократно изменяться, т. е. допускаются инструкции вида $x := x + a$.

Циклы. На графе связей между ячейками табличной модели должны отсутствовать замкнутые контуры, в противном случае циклическая ссылка пересчитывалась бы до бесконечности. Запрет на циклические ссылки означает, что в алгоритмической табличной модели нельзя использовать стандартные вычислительные циклы с возвратом в их начало, как это делается в обычном структурном программировании. Все вычислительные циклы модели должны разворачиваться явно в последовательных строках/столбцах листа Excel подобно тому, как на заре вычислительной техники разворачивались в последовательности инструкций макросы в компьютерных программах.

Ветвления. Для организации пересчета ЭТ, зависимого от условий, в алгоритмической табличной модели следует использовать встроенную функцию Excel ЕСЛИ. Логическое условие, используемое в функции ЕСЛИ, может быть достаточно сложным и, в частности, включать другие функции, а также ссылаться на ячейки табличной модели, содержащие результаты предыдущих вычислений.

Подпрограммы. Из двух видов подпрограмм (процедур и функций) в алгоритмических табличных моделях естественным образом можно пользоваться только функциями Excel, как встроенными, так и созданными пользователем. Известно, что Excel содержит около 350 встроенных функций, и это богатство функций является одним из важнейших преимуществ алгоритмического табличного моделирования в Excel.

Итак, показанная нами возможность использования в табличных моделях базовых управляющих конструкций структурного программирования означает, что таблицы Excel можно создавать обдуманно, на основе блок-схем алгоритмов и графа связей между ячейками, а не по наитию. Можно сказать, что граф связей отражает факт влияния переменных модели друг на друга, а блок-схема алгоритма показывает, как конкретно, посредством каких функциональных или алгоритмических зависимостей осуществляется это влияние.

Алгоритмические табличные модели рекомендуется проектировать и создавать в следующей последовательности.

1. *Изучение бизнес-среды / предметной области*: структуризация словесного описания, постановка задачи, формулировка целей моделирования.

2. *Формализация задачи моделирования*: определение входов, выходов и показателей эффективности, задание символических переменных, выявление связей между ними, построение причинно-следственных диаграмм (диаграмм влияния) [2], установление математических зависимостей между переменными модели, разработка блок-схемы алгоритма обработки данных.

3. *Создание алгоритмической табличной модели*: построение графа связей между ячейками, разработка структуры таблиц модели, оптимизация размещения ячеек на рабочем листе, реализация алгоритма обработки данных посредством формул и функций Excel, создание пользовательского интерфейса, тестирование и отладка табличной модели.

4. *Анализ модели*: разработка плана модельного эксперимента, решение/прогоны табличной модели, анализ и интерпретация полученных результатов.

Все этапы проектирования табличной модели могут итерационно повторяться, пока не будет получен результат, адекватный поставленной задаче и целям моделирования.

Предлагаемая техника АТМ обладает следующими достоинствами:

- проектирование табличной модели становится обдуманным и логически обоснованным;
- разработчик по необходимости значительно больше внимания уделяет созданию концептуальной модели и формализации решаемой задачи, математическим и функциональным зависимостям между переменными, алгоритму обработки данных;
- большая часть времени разработчика затрачивается на работу непосредственно с данными, а не на кодирование в VBA;
- с помощью этой техники многие модели могут быть построены самостоятельно менеджером или сотрудником компании — хорошим специалистом в своей области и посредственным — в программировании;
- техника АТМ особенно полезна в учебных целях, позволяя обучающимся быстро и эффек-

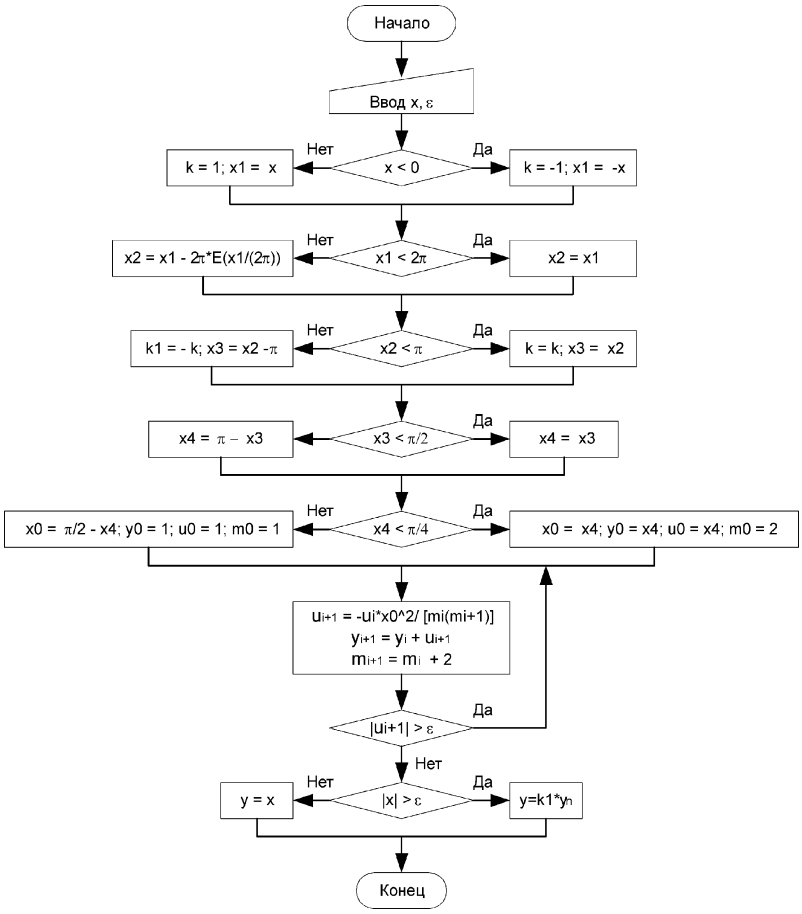

Рис. 1. Блок-схема алгоритма вычисления синуса

тивно создавать логически безупречные модели из разных предметных областей.

Наряду с перечисленными достоинствами, техника АТМ имеет два существенных недостатка:

- алгоритм обработки модельных данных должен быть вычислительным, а не процедурным;
- техника хорошо подходит для создания одномерных, двумерных и в некоторых случаях трехмерных моделей. Для решения задач большей размерности ее использование становится неэффективным из-за плоской структуры электронной таблицы.

Пользуясь техникой АТМ, авторы реализовали в Excel табличные модели большинства алгоритмов, составляющих основу классических численных методов [3]: вычисления по рекуррентным формулам; решение алгебраических уравнений и систем, в том числе итерационным методом Зейделя; расчет значений многочленов по схеме Горнера; нахождение корней трансцендентных уравнений методами деления отрезка пополам, хорд, касательных, простой итерации; одномерная и двумерная оптимизация разными методами; численное дифференцирование, интегрирование, решение дифференциальных уравнений (в том числе в частных производных); решение вероятностных задач методом Монте-Карло; динамическое моделирование работы систем массового обслуживания и др.

В качестве примера создания простой алгоритмической табличной модели рассмотрим решение следующей задачи: используя разложение функции $y = \sin(x)$ в степенной ряд, вычислить значение синуса для произвольного значения аргумента x с заданной точностью ε.

Для вычисления синуса используем разложение функции $y = \sin(x)$ в ряд

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \ldots$$

Чтобы ряд сходился быстро, в алгоритме вычисления синуса (см. рис. 1) величина x сначала приводится к значению $|x| < 1$. Это достигается путем последовательного уменьшения аргумента x до значений $x < 2\pi$, $x < \pi$, $x < \pi/2$, используя функцию выделения целой части числа $E(x)$, а также тригонометрические формулы приведения. Если $\pi/4 < x < \pi/2$, то аргумент уменьшается до величины $\pi/2 - x$, и вычисление синуса сводится к вычислению косинуса, т. е. используется степенной ряд

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \ldots$$

Здесь в цикле суммирования членов степенного ряда используются рекуррентные формулы, *i*-й член ряда обозначен через *ui*, значение функции — через *yi*.

Граф связей между ячейками табличной модели вычисления синуса представлен на рис. 2, а реализация этой модели в Excel — на рис. 3.

На рис. 3 цикл вычислений синуса развернут в интервале ячеек C10:F20. Строка C9 содержит

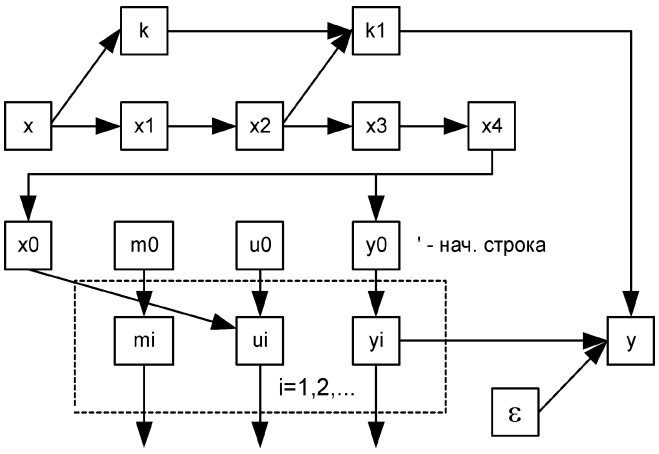

Рис. 2. Граф связей между ячейками табличной модели вычисления синуса

	A	B	C	D	E	F	G	H
1								
2		x	2,617994	' x = 5/6·π	y	0,500000		
3		ε	0,00001					
4								
5		x1	x2	x3	x4	k	k1	
6		2,617994	2,617994	2,617994	0,523599	1	1	
7								
8		x0	mi	ui	yi	A(ε)		
9		0,523599	2	0,523599	0,523599	0	' init	
10			4	-0,02392	0,499674	0		
11			6	0,000328	0,500002	0		
12			8	-2,1E-06	0,500000	1		
…			…	…	…	…		
20			24	-1,3E-29	0,500000	1		
21								

Рис. 3. Алгоритмическая табличная модель вычисления синуса

данные инициализации цикла. В модели используются следующие формулы:

```
[x1] : = ЕСЛИ($C$2 < 0;-$C$2;$C$2)
[x2] : = ЕСЛИ($B$6 < 2*ПИ(); $B$6;
$B$6-2*ПИ()*0КРУГЛВНИЗ($B$6/2/ПИ();
0))
[x3] : = ЕСЛИ($C$6 < ПИ();$C$6;$C$6-ПИ())
[x4] : = ЕСЛИ($D$6 < ПИ()/2;$D$6;ПИ()-$D$6)
[k] : = ЕСЛИ($C$2 < 0;-1;1)
[k1] : = ЕСЛИ($C$6 < ПИ() ;1;-1)
[y] : = ЕСЛИ(ABS($C$2) < $C$3; $C$2;
$G$6*СМЕЩ($F$8; ПОИСКПОЗ(1; $F$9:$F$20;
0) ; -1))
[x0] : = ЕСЛИ($E$6 < ПИ()/4; $E$6;
ПИ()/2-$E$6)
[m0] : = ЕСЛИ($E$6 < ПИ()/4; 2; 1)
[u0] : = ЕСЛИ($E$6 < ПИ()/4; $E$6; 1)
[y0] : = ЕСЛИ($E$6 < ПИ()/4; $E$6; 1)
[A(ε)] : = ЕСЛИ(ABS($D9) < $C$3; 1; 0)
```

Формулы тела цикла (строка 10)

```
[C10] : = $C9 + 2
[D10] : = -$D9*$B$9^2/$C9/($C9+1)
[E10] : = $E9 + D$10
[F10] : = ЕСЛИ(ABS($D10) < $C$3; 1; 0)
```

копируются в другие строки интервала C10:F20 с помощью маркера заполнения.

Список литературы

1. **Аникин В. И., Аникина О. В.** Техника алгоритмического табличного моделирования в Excel // Сб. тез. докл. научно-методической конференции "Научно-методическое обеспечение инновационного развития образовательного учреждения". — Тольятти, 2007. С. 211—214.
2. **Мур Дж., Уэдерфорд Л. Р. и др.** Экономическое моделирование в Microsoft Excel. 6-е изд.: Пер. с англ. — М.: Издательский дом "Вильямс", 2004. 1024 с.
3. **Киреев В. И. Пантелеев А. В.** Численные методы в примерах и задачах: Учеб. пособие — 2-е изд. стер. — М.: Высшая школа, 2006. 480 с.

ИНФОРМАЦИЯ

Перспективы развития высокопроизводительных вычислительных архитектур

Всероссийская научно-практическая конференция, приуроченная к 60-летию ИТМиВТ им. С. А. Лебедева РАН

26 июня 2008 г.

Более 40 лет (начиная с 1950-х годов) отечественная вычислительная техника практически не отставала от зарубежных аналогов, преимущественно в области создания мощных вычислительных комплексов для решения научных и оборонных задач. Особую роль в этом вопросе сыграл ИТМиВТ, разработав более 20 уникальных типов ЭВМ общего и специального назначения. События 1990-х годов подорвали отрасль. Сегодня Россия переживает экономический и технический подъем, а возможность создания и производства собственных вычислительных комплексов соответствует статусу мировой державы. Необходимо возродить научную школу разработки вычислительной техники и воссоздать технологическую базу для их производства. Для решения такой глобальной задачи требуется поддержка государства и кооперация сохранившихся проектных центров. Именно этим вопросам была посвящена конференция, организованная ИТМиВТ.

Объединение проектных центров

В преддверии 60-летнего юбилея, 26 июня 2008 г., Институт точной механики и вычислительной техники им. С. А. Лебедева РАН при поддержке Федерального агентства по промышленности РФ (Роспром), Российской академии наук и участии ведущих отраслевых организаций провел Всероссийскую научно-практическую конференцию "Перспективы развития высокопроизводительных вычислительных архитектур. История, современность и будущее отечественного компьютеростроения". Тема мероприятия неотделимо связана с деятельностью ИТМиВТ. Благодаря достижениям ученых тех лет в кратчайшие сроки фактически с нуля была создана новая отрасль вычислительной техники.

В задачу конференции входили определение проблем в построении производительных вычислительных архитектур, нахождение возможных точек прорыва, осмысление путей интеграции российских разработчиков в мировое разделение труда. Данная конференция была призвана объединить усилия коллективов, привлечь внимание специалистов и государственных руководителей к поиску новых интересных архитектурных подходов, что в целом будет способствовать возвращению России в мировую индустрию компьютеростроения.

В настоящее время к предприятиям отрасли электроники и вычислительной техники поступил мощный инновационный импульс со стороны государства, заинтересованного в преодолении отечественным компьютеро-

строением периода упадка. В свете наметившегося прогресса по данному вопросу вырабатывается механизм поддержки проведения работ по созданию отечественных суперкомпьютеров и объединения усилий сохранившихся и недавно образованных коллективов разработчиков. Сейчас отрасль практически преодолела состояние кризиса, а специалисты доказали свою способность вести передовые разработки. На этой основе сформированы зоны ответственности компьютеростроения в сфере национальной безопасности и запланировано их расширение.

По словам **А. Е. Суворова,** начальника управления радиоэлектронной промышленности и систем управления Федерального агентства по промышленности РФ (Роспром): "Особое значение имеет усовершенствование и развитие системы проектных центров, в которых создаются новые модели отечественных компьютеров. В этой связи очень важно освоение опыта проектной деятельности советского периода и роли в нем ИТМиВТ им. С. А. Лебедева, как флагмана отечественного компьютеростроения тех лет".

К настоящему моменту имеются все необходимые предпосылки к выполнению национального проекта по разработке отечественного супервычислителя. Помимо ИТМиВТ им. С. А. Лебедева РАН в той или иной степени сохранили свой потенциал ведущие проектные центры в области вычислительной техники: НИЦЭВТ, ИНЭУМ, НИИСИ РАН и ИПМ им. М. В. Келдыша РАН. При объединении компетенций перечисленных научных организаций возможно восстановить между ними отраслевую кооперацию, в которую могут войти еще не менее 15 научных центров и предприятий России, работающих в области микроэлектроники и вычислительной техники. Общими усилиями задачу по созданию отечественной суперЭВМ решить реально в разумные сроки — в пределах 5 лет. На возрождение полноценной научной школы в области вычислительной техники, аналогичной той, которая была создана С. А. Лебедевым, потребуется не менее 10—15 лет.

Директор ИТМиВТ **С. В. Калин** отмечает актуальность обращения ряда отраслевых организаций к вопросу разработки российского суперкомпьютера: "России нужен проект по созданию отечественной суперЭВМ, а также восстановлению промышленно-технологической инфраструктуры — экосистемы электроники. Организации, принимающие участие в конференции, по факту и составляют костяк сети проектных центров, на которых может основываться разработка перспективных вычислительных машин".

Программа конференции была составлена таким образом, чтобы в полной мере показать существующие препятствия для реализации глобального проекта создания российской суперЭВМ и наметить возможные пути решения стратегически важной задачи. В программных выступлениях отражены вопросы, связанные с мировыми тенденциями развития вычислительных архитектур, представлены достижения отечественных коллективов разработчиков, работающих над созданием аппаратно-программных платформ, суперкомпьютерных технологий, высокопроизводительных процессоров.

Тему национальной, в том числе и информационной, безопасности поднял **Л. К. Эйсымонт,** заместитель главного конструктора суперкомпьютера стратегического назначения (СКСН) "Ангара", ОАО "НИЦЭВТ", который акцентировал внимание на следующем вопросе: "Для обеспечения национальной безопасности и решения наиболее важных научно-технических проблем России необходимо иметь мощные вычислительные центры, оснащенные СКСН. Исследования и разработки, расчеты производительности позволяют утверждать, что после 2011 г. в России реально изготовить образцы такой СКСН с характеристиками, близкими к перспективным американским High-End Computers. Залогом успеха решения этой задачи станет восстановление инфраструктуры на основе кооперации организаций научных институтов и производственных центров".

В докладе **А. К. Кима,** генерального директора ОАО "ИНУЭМ", в первую очередь была освещена работа по созданию МВК "Эльбрус-3", в основе которой заложены знаменитые разработки ИТМиВТ "Эльбрус-1,2". Докладчик раскрыл достигнутые результаты: "Архитектура "Эльбрус" многократно повышает производительность за счет возможности одновременного исполнения операций, заложенной в аппаратную реализацию широких команд, и автоматического распараллеливания программ оптимизирующим компилятором, использующим аппаратную поддержку. Данный проект станет основой для реализации стратегического плана по созданию суперкомпьютера петафлопной производительности".

Разработкой многопроцессорных вычислительных структур с динамически реконфигурируемой архитектурой на основе ПЛИС занимается НИИ МВС ЮФУ, о чем рассказал в своем докладе заместитель директора по научной работе НИИ МВС ЮФУ **И. И. Левин.** Основная мысль его выступления состояла в следующем: "При решении многих практических задач реальная производительность многопроцессорных вычислительных систем (МВС) резко падает и не превышает 5—10 % от пиковой. Основная причина этого заключается в несоответствии "жесткой" архитектуры МВС и информационной структуры широкого класса решаемых задач. Данную проблему можно устранить с помощью концепции МВС с "гибкой", динамически реконфигурируемой (программируемой) архитектурой, подстраиваемой под информационную структуру каждой конкретной, решаемой в текущий момент времени задачи".

Еще одной перспективной отечественной платформе "Мультикор" был посвящен доклад **Я. Я. Петричковича,** директора ГУП НПЦ "Элвис": "Реализация принципа сквозного иерархического параллелизма в СБИС серии "Мультикор" и в многокристальных системах на ее основе обеспечивает масштабируемую пиковую производительность при обработке сигналов и изображений. Сравнительный анализ показывает, что отечественный сигнальный микропроцессор не уступает лучшим известным DSP- процессорам даже при реализации по худшим проектным нормам".

В докладе **С. В. Калина** был сделан обзор разработок ИТМиВТ прошлых лет с описанием уникальных технических решений, реализованных впервые советскими инженерами и только потом подхваченными зарубежными коллегами. Также рассказано о современных результатах, достигнутых Институтом за последние с начала реформирования три года.

* * *

На рубеже своего юбилея ИТМиВТ продолжает работу в области разработки перспективных информационно-коммуникационных технологий. В качестве стратегической цели было определено формирование условий для разработки в нашей стране собственных высокопроизводительных вычислительных архитектур и суперкомпьютеров. Но решить такую задачу в одиночку и без поддержки государства невозможно. Это привело Институт к пониманию необходимости формирования отраслевой кооперации и наращиванию компетенций, которые в итоге позволили бы принять участие в масштабном проекте по суперЭВМ.

В качестве важного достижения Института следует отметить создание Дизайн-центра микроэлектроники и радиоэлектронной аппаратуры численностью 70 высококвалифицированных специалистов, ведущих работы по созданию цифроаналоговой техники, сложнофункциональных блоков, "систем-на-кристалле" и т. д.

Активная работа ведется в области построения перспективных цифровых систем управления авиационными двигателями — разработано семейство бортовых процессорных модулей, являющихся звеньями систем автоматического управления газотурбинным двигателем (САУ ГТД).

Инновационным направлением Института является создание беспроводных систем мониторинга широкого применения на основе реализации собственного стека ZigBee. К настоящему моменту произведены и прошли сертификацию модули для построения систем промышленного мониторинга и охранно-пожарной сигнализации.

Важными проектами, выполняемыми ИТМиВТ, являются государственные заказы в области: внедрения системы биопаспортов; создания системы информационно-навигационного обеспечения железнодорожного транспорта с использованием ГЛОНАСС/GPS/Galileo; разработки видеоконтроллера высокого разрешения для мультимедийных систем цифрового ТВ и др.

В ИТМиВТ проводятся научные исследования в целях поиска принципиально новых решений в области построения архитектур вычислительной техники (об этом на конференции рассказал руководитель направления **А. М. Степанов**). Развитие получила не фон-неймановская модель вычислений, которая является обобщением принципа динамического dataflow. Сформулированы архитектурные принципы и требования к структуре перспективного многоядерного процессора на кристалле.

Перспективным признано направление по развитию технологий оптимизирующей компиляции, которая многократно сокращает трудозатраты на создание эффективного компилятора, позволяет оптимизировать компиляцию любым существующим компилятором, автоматизировать распараллеливание на произвольное число ядер и получить высоконадежный и эффективный машинный код для любой существующей или новой вычислительной архитектуры (доклад руководителя направления **А. Ю. Дроздова**).

По итогам конференции **С. В. Калин** отмечает: "Основой современных далеко идущих планов по созданию отечественной суперЭВМ является славная более чем полувековая история и большое число разработок. ИТМиВТ уверенно восстанавливает былой потенциал и готов к участию в серьезном проекте национального масштаба. Это позволяет надеяться, что общее дело по восстановлению прежних позиций России в сфере разработки супервычислителей получит достойную поддержку, задуманное будет реализовано, а конференция станет проводиться на регулярной основе".

CONTENTS

The solution of the problem of automatic structure of production rules on the base of application of genetic programming is suggested. Geneticed rules are destined for formation of ntologies structure library.
Keywords: production rules, genetic programming, formation of ontologies, chromosome structure.

In the article a number of directions in software development are discussed where using ontologies can lead to a considerable progress. In the first part basic concepts related to ontologies and possibilities for using ontologies to analyze domains and to build their mathematical models are considered.
Keywords: ontology, conceptualization, knowledge base, software development, analytics, domain mathematical model, metaontology, ontological analysis.

The GOST-28147 encryption implementation on graphics processing units for DirectX and OpenGL graphics API is suggested. The approach for organizing the computation process and some realizations of base operation are described. The results of computational experiments on different video cards are given. The results confirm the possibility for applying this technology for organization of high speed stream encryption system.
Keywords: symmetric ciphering, parallel processing, graphics processing units, pixel shaders.

Some settings of the targets which could be appear during the process of the education system perfections are considering. These targets are brings to the multiple-vector task of ranking. The solution method is offering for the example task of the selection the postulants to the posts of leading and academic staff. The numerical example is considering.
Keywords: The education system perfections, the miltiple-vector task of ranking, keading and academic staff.

An information taxation control technology is developed, it is based on a hybrid neural network model. The mentioned model consists of neural network approximation model of taxpayers cluster production function and also of a probabilistic model of taxpayers ranging, taking into account report documentation, prehistory and scale of activities. Original methods of data pre-processing are suggested and confirmed by computing experiments. These methods enhance the quality and adequacy of the developed models.
Keywords: taxation control technology, hybrid neural network model, network approximation model of taxpayers cluster production function, probabilistic model of taxpayers ranging, data pre-processing, method of inserted mathematical models.

Today Supreme Attestation Committee of MES of Russia poses a problem to use systematically information technologies for interaction between dissertation council and official. The current situation in document circulation inside Committee is considered. Web forms as a main mechanism for e-documents creation are proposed. A practice and perspectives of Internet publication of theses and full texts of dissertations is discussed. The structure and interfaces of databases "Scientists of Russia", "Dissertation Councils", etc. is analyzed.
Keywords: supreme attestation committee, information technologies, certifying processes of scientists, automated control systems, informationally-analysis systems, web-forms.

The paper considers the application of the linear constraints in normalized gradient algorithm, used to control of adaptive arrays based on constant modulus criterion. Computational procedure and arithmetic complexity of the algorithm are presented. Simulation compares the efficiency of the presented adaptive algorithm and the linearly constrained least squares algorithm.
Keywords: adaptive arrays, linear constraints, constant modulus criterion, normalized least means squares algorithm, constellation.

Original technique of creating table models in Microsoft Excel without programming on VBA language is discussed. The idea of technique is based on analogy between cells content recalculating process in electronic table and command stream executing in computer program, and on principles of structural programming. The modeling technique may be useful for managers, applied specialists, high school teachers, aspirants and students.
Keywords: algorithmic table modeling, Microsoft Excel, table model, programming without coding, structural programming.

Адрес редакции:

107076, Москва, Стромынский пер., 4/1

Телефон редакции журнала **(495) 269-5510**

E-mail: it@novtex.ru

Дизайнер *Т.Н. Погорелова*. Художник *В.Н. Погорелов*.
Технический редактор *О. А. Ефремова*. Корректор *Е. В. Комиссарова*

Сдано в набор 08.08.2008. Подписано в печать 19.09.2008. Формат 60×88 1/8. Бумага офсетная. Печать офсетная.
Усл. печ. л. 9,8. Уч.-изд. л. 11,65. Заказ 1058. Цена договорная.

Журнал зарегистрирован в Министерстве Российской Федерации по делам печати,
телерадиовещания и средств массовых коммуникаций.
Свидетельство о регистрации ПИ № 77-15565 от 02 июня 2003 г.

Отпечатано в ООО "Подольская Периодика"
142110, Московская обл., г. Подольск, ул. Кирова, 15